Median based approaches for noise suppression and interest point detection.

Miao, Zhenwei.

2013

Miao, Z. (2013). Median based approaches for noise suppression and interest point detection. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/55053>

<https://doi.org/10.32657/10356/55053>

Median Based Approaches for Noise Suppression and Interest Point Detection

MIAO Zhenwei

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2013

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

.....

Date

.....

MIAO Zhenwei

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my supervisor, Associate Professor JIANG Xudong, for his professional guidance, unconditional support and continuous encouragement through my Ph.D. study and research. It has been an honor for me to be one of his Ph.D. students. His immense knowledge, inspiring nature and enthusiastic assistance have been great source of motivation throughout my doctoral studies. Without his numerous help, suggestions and discussions, I would not have completed this research work. His sharp insights, criticism and strict attitude towards the research work have been a resource and model for me to complete this thesis and they will continuously benefit me throughout my lifetime.

I would like to thank Professor Alex C. KOT and Associate Professor YAP Kim Hui from Nanyang Technological University, Professor YANG Jun and Associate Professor XU Yong from Chinese Academy of Sciences, and Professor PENG Linhui from Ocean University of China for their encouragements, supports and helps. I also would like to thank Dr. JI Peifeng, Dr. YANG Huijuan, Dr. CAO Hong, Dr. GENG Cong, Dr. LEI Baiying, Dr. LIU Siyuan, Dr. CHEN Changsheng, Dr. LI Sheng, Ms. WANG Yan, Ms. FAN Jiayuan, Ms. ZHAN Huijing, Mr. JI Wei, Mr. LAI Jian, Mr. Amit SATPATHY and Mr. REN Jianfeng for their encouragements and helps. I am grateful to Mr. Stewart CHU for his kind technical supports.

I sincerely thank the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore, for providing me the full research scholarship and support to complete my Ph.D. study.

Finally, I must thank my family and my friends. I would like to dedicate this work to my parents and my sisters for their persistent love, continuous encouragement, endless support and sacrifice.

Abstract

Mean and median are two basic operations in signal/image processing. They are widely used for their easy implementation and soundly mathematical analysis tools. The mean filter achieves the best performance in attenuating Gaussian noise. However, it cannot effectively suppress the long-tailed noise and it blurs image structures. On the contrary, the median filter has the advantages in suppressing the long-tailed noise and preserving image structures. These advantages motivate us to develop the median based approaches in both noise suppression and interest point detection.

Noise suppression is a fundamental and important research topic in signal/image processing. The recently proposed iterative truncated arithmetic mean (ITM) filter provided an effective way to suppress the long- and short-tailed noise. By iteratively truncating the extreme samples, the ITM filter's output starts from the mean and approaches the median. The termination condition enables the ITM filter owning merits of these two operations. The filter's output can be used as an approximation of the sample median without using the time consuming data sorting algorithm. The merits of the ITM filter inspire part of the work in this thesis.

We firstly analyze the ITM filter and verify that the ITM filter is more effective than the median filter in suppressing both Gaussian and Laplacian noise. Furthermore, we propose a fast implementation named fast ITM (FITM) filter. Mathematical analysis of the computational complexity is given. The ITM and FITM filters are of order $\mathcal{O}(n\sqrt{n})$ and $\mathcal{O}(n \log n)$, respectively. It is seen that the FITM filter has a lower computational complexity than the ITM filter.

As band- and high-pass characteristics are expected in many applications, we propose a rich class of filters named weighted ITM (WITM) filters in this thesis. By iteratively truncating the extreme samples, the output of the WITM filter moves from the weighted mean towards the weighted median. Proper stopping criterion makes the WITM filters own some merits of both the weighted median and mean filters and, therefore, outperform the both in some applications. Three structures are designed to enable the WITM filters being low-, band- and high-pass filters. Properties of these filters are presented and analyzed in this work.

A more practical noise model for real images is a mixed-type noise which contains the additive and exclusive noise. Although the ITM filter can effectively deal with the additive noise, its result is not optimal in case the exclusive noise exists. As samples corrupted by the exclusive noise do not contain the information of the signal, the best way is to remove (trim) such samples from the filter inputs. This inspires the proposed iteratively trimmed and truncated mean (ITTM) filter. The proposed ITTM filter outperforms the mean, median and ITM filters in many cases. It has a linear computational complexity with order of $\mathcal{O}(n)$ which is smaller than that of the ITM filter.

Another important research topic in image processing is the interest point detection. The interest points refer to the image patterns which are different from their immediate neighborhoods. As such image patterns can be used to represent the image robustly and sparsely, the interest point detection has drawn great attentions and been widely used in computer vision in the last two decades. However, designing a robust interest point detector to deal with the complex image structures and various variations is still a challenging task. Many leading detectors, such as SIFT and SURF detectors, employ the linear LoG filter to detect the blob structures from images. The inherent shortcoming of the linear filter that cannot effectively deal with abrupt variations limits the performance of the corresponding detectors. Such detectors cannot stably extract the interest points in case impulsive noise or abrupt variations exist. The merit of the median filter that can effectively deal with such kind of variations inspires the proposed

detectors in this thesis. A rank order filter named rank order Laplacian of Gaussian (ROLG) filter is designed in this work. A novel interest point detector is designed based on this filter to detect the image local structures where a significant majority of pixels are brighter or darker than a significant majority of pixels in their corresponding surroundings. Compared to the linear filter based detectors, e.g. SIFT detector, the proposed ROLG detector is more robust to abrupt variations of images.

Another issue of the linear filter based detectors is that their responses are proportional to the local image contrast. This makes such detectors prefer the local structures with high contrast. Low contrast structures will not be easily detected even if they are stable under different variations. The ROLG detector uses the rank order filter instead of the linear filter to reduce the influence of noise and the nearby structures. However, it still prefers the structures which have high contrast. In this thesis, a vote of confidence (VC) based detector is proposed to detect bright and dark regions from images. Whether a local region is bright or dark is voted by all the pixels in this region. The proposed VC detector is robust to illumination changes and effective to cluttered structures.

In general, our work is based on the median operation but not restricted on it. The proposed FITM, WITM, and ITTM filters own some merits of both the median and mean operations. These filters have better performance in suppressing noise and preserving structures. The median based interest point detectors inherit the merits of the median filter which is insensitive to the abrupt structures. Therefore, the proposed ROLG and VC detectors can effectively deal with the abrupt variations caused by illumination and geometric changes.

The contributions of this thesis are summarized as follows:

1. Studied the properties of the ITM filter and proposed a fast realization named FITM filter. Compared to the median filter, the FITM filter has a better performance in suppressing both Gaussian and Laplacian noise with a faster speed.

2. Proposed a rich class of weighted ITM (WITM) filter. By assigning the proper weights, the WITM filter can be designed as low-, band- and high-pass filters.
3. Proposed the ITTM filter. By iteratively trimming and truncating the extreme samples, the ITTM filter outperforms the median and ITM filters in some cases, and has a lower computational complexity.
4. Proposed the ROLG filter for interest point detection and designed the ROLG detector. The ROLG detector is more robust to the abrupt variation than the linear filter based detector.
5. Proposed a vote of confidence based filter for interest point detection and designed the VC detector. Compared to the leading interest point detectors, the VC detector is more robust to illumination changes.

Contents

Acknowledgements	i
Abstract	iii
List of Figures	xv
List of Tables	xxv
1 Introduction	1
1.1 Motivation	1
1.2 Contributions	5
1.3 Organization of the Thesis	7
2 Literature Review	8
2.1 Filters in Noise Suppression	8
2.1.1 Median Based Filters	8
2.1.1.1 Mean and Median Filters	9
2.1.1.2 Alpha-trimmed Mean Filter	10
2.1.1.3 Modified Trimmed Mean Filter	10

2.1.1.4	Mean-Median Filter	11
2.1.1.5	Median Affine Filter	11
2.1.1.6	Optimal Myriad Filter	12
2.1.1.7	Mean-LogCauchy Filter	13
2.1.1.8	Iterative Truncated Arithmetic Mean Filter	13
2.1.1.9	Discussion	15
2.1.2	Soft-limiting Filters	16
2.1.2.1	Anisotropic Filter	17
2.1.2.2	Total Variation Minimization Filter	17
2.1.2.3	SUSAN Filter	17
2.1.2.4	Bilateral Filter	18
2.1.2.5	Non-local Mean Filter	18
2.1.2.6	Discussion	19
2.2	Interest Point Detection	19
2.2.1	Corner Detectors	20
2.2.1.1	Moravec Detector	20
2.2.1.2	Harris Detector	21
2.2.1.3	Harris-Laplace/Affine Detector	22
2.2.1.4	SUSAN Corner Detector	23
2.2.1.5	FAST: Features from Accelerated Segment Test	24
2.2.1.6	Paler Detector	26

2.2.1.7	Discussion	26
2.2.2	Blob Detectors	27
2.2.2.1	Hessian Detector	27
2.2.2.2	Hessian Laplace/Affine Detector	29
2.2.2.3	SIFT Detector	29
2.2.2.4	SURF: Speeded-Up Robust Features Detector	31
2.2.2.5	Salient Region Detector	32
2.2.2.6	Self-similarity Detector	33
2.2.2.7	Discussion	35
2.2.3	Region Detectors	36
2.2.3.1	Maximally Stable Extremal Region Detector	36
2.2.3.2	Intensity-based Region Detector	38
2.2.3.3	Principal Curvature-Based Region Detector	39
2.2.3.4	Discussion	40
2.2.4	Evaluation Criterion	40
2.3	Summary	42
3	Further Properties and a Fast Realization of the ITM Filter	44
3.1	Introduction	44
3.2	Noise Suppression Properties of the ITM Filter	45
3.2.1	Gaussian Noise	47
3.2.2	Laplacian Noise	49

3.3	The Proposed Fast ITM Filter	52
3.4	Computational Complexity	55
3.5	Summary	62
4	Weighted Iterative Truncated Mean Filter	63
4.1	Introduction	63
4.2	Weighted ITM Filters with Positive Weights	66
4.2.1	Weighted Mean and Median Filters	66
4.2.2	Iterative Truncated Arithmetic Mean Filter	68
4.2.3	The Proposed Weighted ITM Filter with Positive Weights	69
4.2.4	Stopping Criterion	74
4.2.5	Properties of the WITM Filter with Positive Weights	79
4.3	Weighted ITM Filter Admitting Negative Weights	80
4.3.1	General WITM Filter with Negative Weights	81
4.3.2	Linear Combined WITM Filter with Negative Weights	82
4.3.3	The Proposed Dual WITM Filter with Negative Weights	83
4.4	Experiments	86
4.4.1	Attenuation of the Short- and Long-tailed Noise	86
4.4.2	Frequency Selective WITM Filters	91
4.4.3	Design of High-pass WITM Filters	94
4.4.4	Image Denoising	98
4.5	Summary	102

5	Iterative Trimmed and Truncated Arithmetic Mean Filter	103
5.1	Introduction	104
5.2	The Proposed ITTM Filter	106
5.2.1	Iterative Truncated Arithmetic Mean Filter	106
5.2.2	The Proposed ITTM Filters	107
5.3	Properties of the ITTM Filters	115
5.4	Computational Complexity	122
5.5	Experiments	131
5.5.1	Single Type of Noise in Constant Signal	132
5.5.2	Mixed Types of Noise in Constant Signal	133
5.5.3	Noise Step Edge	134
5.5.4	Real Images	136
5.5.5	α -Stable Noise Corrupted Homogeneous Images	137
5.5.6	Gaussian and α -Stable Noise Corrupted Real Image	138
5.6	Summary	140
6	Rank Order Laplacian of Gaussian Filter and ROLG Interest Point Detector	141
6.1	Introduction	141
6.2	The Proposed ROLG Filter	143
6.2.1	LoG Filter	143
6.2.2	Weighted Rank Order Filter	144
6.2.3	The Proposed ROLG Filter	146

6.3	Analyses of the ROLG Filter	147
6.3.1	Responses of the ROLG Filter on Blobs	148
6.3.2	Responses of the ROLG Filter on Edges and Corners	149
6.4	Interest Point Detection by the ROLG Filter	150
6.4.1	Interest Point Detection in a Single Scale	150
6.4.2	Eliminating Ridge Responses	153
6.4.3	Algorithm for the ROLG Detector	154
6.5	Experiments	155
6.5.1	Repeatability and Discrimination Tests	156
6.5.2	Application to Face Recognition	162
6.5.2.1	Results on AR Database	164
6.5.2.2	Results on ORL Database	165
6.5.2.3	Results on GT Database	166
6.5.2.4	Results on FERET Database	167
6.5.2.5	Results on LFW Database	168
6.6	Summary	169
7	Interest Point Detection Based on Vote of Confidence	170
7.1	Introduction	170
7.2	Vote of Confidence Based Detector	172
7.2.1	Voting Algorithm	173
7.2.2	Ridge Suppression	176

7.2.3	VC Detector in Multiple Scales	177
7.3	Experiments	180
7.3.1	Visual Inspection	180
7.3.2	Intraclass Variations	181
7.3.3	Repeatability and Discrimination Tests	185
7.3.4	Application to Face Recognition	188
7.4	Summary	190
8	Conclusions and Future Work	191
8.1	Contributions in Noise Suppressing	191
8.1.1	Fast ITM Filter	192
8.1.2	Weighted Iterative Truncated Mean Filter	192
8.1.3	Iteratively Trimmed and Truncated Arithmetic Mean Filter	193
8.2	Contributions in Interest Point Detection	193
8.2.1	Interest Point Detection Using Rank Order LoG Filter	194
8.2.2	Interest Point Detection Based on Vote of Confidence	194
8.3	Future Work	194
8.3.1	Probability Analysis of the Proposed Filters	194
8.3.2	Iterative Truncated Mean Filters in Multiple Dimensions	195
8.3.3	Trimming and Truncating Algorithm for Robust Statistic	195
8.3.4	Affine Invariant Interest Point Detector	195
	Author's Publications	196

Bibliography**197**

List of Figures

2.1	(a) Input image. (b) Response of the Movervec corner detector.	20
2.2	Image structures and their corresponding gradient fields, from (a) to (d) are corner, textured structure, gradient field of the corner, and gradient field of the textured structure, respectively.	21
2.3	USAN area of the SUSAN detector: (a) original image; (b) USAN areas of 5 circular masks, in which the white area indicate the USAN area.	24
2.4	Illustration for the FAST detector. Three Bresenham's circles are shown here: nucleus A is on the corner, nucleus B is on the flat area, and nucleus C is on the edge.	25
2.5	Corners detected by the Paler detector: (a) corners detected on an artificial image; (b) corners detected on this image polluted by pepper&salt noise.	26
2.6	Different order derivatives of Gaussian curves: g represents the Gaussian curve, gx represents the first order derivative of the Gaussian curve, $\text{abs}(gxx)$ represents absolute value of the second order derivative of the Gaussian curve.	28
2.7	Overview of the SIFT detector scheme [1].	30

2.8	From left to right: the (discretised and cropped) second order Gaussian derivative $\frac{\partial^2}{\partial y^2}G(\sigma)$ and $\frac{\partial^2}{\partial xy}G(\sigma)$, and the box filters approximating for previous Gaussian derivatives respectively. The grey regions are equal to zero [2].	32
2.9	Image patterns and their polar representations: column (a) have high radial saliency; column (b) have high tangential saliency; and column (c) have high residual saliency [3].	35
2.10	Example of the extremum region in the 1D curve: extremum region R_l is obtained with threshold l and $R_{l+\Delta}$ is obtained with threshold $l + \Delta$ [4].	37
2.11	Flowchart of the intensity-based region detector [5].	38
2.12	Flowchart of the PCBR detector: (a) input image; (b) principal curvature image; (c) filtered binary image; (d) watershed regions; (e) detected interest regions fitted by ellipses [6].	39
3.1	Normalized MSE against (a) the number of iterations k , and (b) the filter size n for Gaussian noise.	48
3.2	Normalized MSE against (a) the number of iterations k , and (b) the filter size n for Laplacian noise.	51
3.3	Average number of iterations determined by the stopping criterion in [7] against the filter size n	56
3.4	Average visiting times of a sample against the filter size n . The x -axis is in log scale of n	58
3.5	Average visiting times of a sample against the filter size n . The x -axis is in linear scale of n	59

3.6	Normalized time consumption against the number of iterations k . The time consumption is normalized by that of the median filter.	60
3.7	Normalized time consumption against the filter size n . The time consumption is normalized by that of the median filter.	61
4.1	Ascending sorted data $x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(n)}$, their corresponding weights $\{w_{(1)}, w_{(2)}, \dots, w_{(n)}\}$ and one possible location of the truncated weighted mean μ_w . The weighted median ϕ_w is denoted by $x_{(m)}$	75
4.2	Outputs of frequency selective filters on chirp signal. (a) Chirp signal. (b) Output of linear FIR filter. (c) Output of GWM filter. (d) Output of GWITM filter.	83
4.3	Outputs of frequency selective filters on chirp signal with a constant offset. (a) Chirp signal with a constant offset. (b) Output of linear FIR filter. (c) Output of GWM filter. (d) Output of GWITM filter.	84
4.4	Mean absolute error (MAE) normalized by that of the weighted median filter in suppressing non-identical distributed Gaussian noise against (a) the number of iterations with fixed filter size $n = 25$, and (b) the filter size n . The average numbers of iterations for the ITM filter are 1.71, 3.44, 5.42, 7.50 for the filter size from 9 to 81, respectively, and those for the WITM filter are 1.69, 2.51, 3.82, 5.40.	87

- 4.5 Mean absolute error (MAE) normalized by that of the weighted median filter in suppressing non-identical distributed Laplacian noise against (a) the number of iterations with fixed filter size $n = 25$, and (b) the filter size n . The average numbers of iterations for the ITM filter are 2.01, 3.98, 6.05, 8.17 for the filter size from 9 to 81, respectively, and those for the WITM filter are 2.27, 3.99, 5.96, 8.04. The MAE of the mean filter is drastically larger than those of other filters and hence not plotted. “F WITM” represents the WITM filter with fixed numbers of iterations of 7, 12, 17, 22 for the filter size from 9 to 81. 88
- 4.6 Normalized running time against filter size n . The running time is normalized by that of the weighted median filter. The y -axis is in log scale. The weighted myriad filters with $L = 20$ and $L = 5$ iterations are both plotted. 90
- 4.7 Frequency selective filter outputs. (a)-top: Chirp test signal, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 3.21, and those for each sub-filter of the LCWITM and DWITM filters are 1.01 and 2.08, respectively. 92
- 4.8 Frequency selective filter outputs in noise. (a)-top: Chirp test signal in α -stable noise with $\alpha = 1.2$ and $\gamma = 0.1$, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 3.32, and those for each sub-filter of the LCWITM and DWITM filters are 1.06 and 2.87, respectively. 93

- 4.9 Frequency selective filter outputs. (a)-top: two-tune signal, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 1.09, and those for each sub-filter of the LCWITM and DWITM filters are 1.00 and 2.18, respectively. 95
- 4.10 Frequency selective filter outputs in noise. (a)-top: two-tune signal in α -stable noise ($\alpha = 1.2, \gamma = 0.1$), (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 1.36, and those for each sub-filter of the LCWITM and DWITM filters are 1.02 and 1.89, respectively. 97
- 4.11 Normalized mean absolute error (MAE) against the signal to noise ratio (SNR) of input signal. The MAE is normalized by that of the weighted median filter. The MAEs of the LCWM and LCWITM filters are drastically larger than those of other filters and hence not plotted. 99
- 4.12 "Lena" of size 512×512 : (a) original image and (b) corrupted image by ϵ -contaminated ($\epsilon = 0.5$) Gaussian ($\sigma_n^2 = 100$) and α -stable ($\alpha = 1.4, \gamma = 10$) noise. 100
- 4.13 PSNR of the filtered image against the noise level σ of input image. 101
- 5.1 MAD of the samples from the mean against the number of iterations. The input data set is $\mathbf{x}_0 = \{-7, -6, -6, -2, -1, 0, 1, 3, 6, 8, 24\}$ 108

5.2	Average MAD of the samples from the mean against the number of iterations. The input data sets are Laplacian noise. The filter size is $n = 49$	119
5.3	Profile outputs of the mean, median, ITTM1, ITTM2 and ITTM3 filters of size $n = 11 \times 11$ after 3 iterations for a step edge.	121
5.4	Average number of iterations determined by the default stopping criterion against the filter size n . The x -axis is in log scale of n	126
5.5	Average visiting times of a sample against the filter size n	128
5.6	Normalized time consumption against the number of iterations k . The time consumption is normalized by that of the median filter. The y -axis is in log scale.	129
5.7	Normalized time consumption against the filter size n . The time consumption is normalized by that of the median filter. The y -axis is in log scale.	130
5.8	MAE against the filter size n in suppressing (a) Gaussian and (b) Laplacian noise. The MAE is normalized by that of the median filter. The average iteration numbers of the ITM filter are 1.5, 3.3, 5.3, 7.5 in (a) and 1.8, 3.6, 5.7, 7.7 in (b) for the filter size from 9 to 81, respectively. The average iteration numbers of the ITTM filter are 1.4, 2.1, 2.6, 3.1 in (a) and 1.5, 2.3, 2.9, 3.4 in (b).	132
5.9	MAE against the filter size n in suppressing (a) Laplacian ϵ -contaminated Gaus- sian noise, and (b) Laplacian and impulsive ϵ -contaminated Gaussian noise. The MAE is normalized by that of the median filter. The average iteration numbers of the ITM filter are 1.6, 3.4, 5.4, 7.5 in (a) and 1.6, 3.4, 5.5, 7.6 in (b) for the filter size from 9 to 81, respectively. The average iteration numbers of the ITTM filter are 1.4, 2.2, 2.8, 3.2 in (a) and 1.4, 2.2, 2.8, 3.2 in (b).	133

5.10	MAE normalized by that of the median filter for the filter size 3×3 in (a) and 11×11 in (b) against the noise level σ_n . The average iteration numbers of the ITM filter are 1, 1, 1, 1, 2.7, 3.7, 2 in (a) and 1.8, 3.8, 4.7, 11.6, 13.5, 14.7, 12.2 in (b). The average numbers of ITTM iterations are 1, 1, 1, 1, 2.3, 2.8, 1.6 in (a) and 1.8, 2.6, 3.0, 6.7, 6.6, 5.2, 4.1 in (b).	135
5.11	Real images in testing: Crowd, Bank, Girl of size 512×512	135
5.12	Average MSEs over 10 runs for the 3 images at 5 different noise levels of (a) Crowd, (b) Bank and (c) Girl. The average number of ITM iterations is closely around 3.4. The average number of ITTM iterations is closely around 2.1. . . .	136
5.13	Real image “Lena” of size 512×512 tested for the mixed α -stable noise. . . .	139
5.14	Normalized MSEs for real image “Lena” corrupted by ϵ -contaminated ($\epsilon = 0.5$) Gaussian ($\sigma_n^2 = 100$) and α -stable ($\gamma = 10$) Noise.	139
6.1	LoG filter. (a) Shape of the LoG filter. (b) Two parts of the LoG filter. S_1 corresponds to the surrounding ring containing positive weights. S_2 corresponds to the inner circular disk containing all negative weights.	143
6.2	Response on a 1D blob. (a) A 1D blob. (b) Response of the LoG filter. (c) Response of the ROLG filter.	148
6.3	Response on a 2D blob. (a) A 2D blob. (b) Absolute value of the LoG response. (c) Absolute value of the ROLG response.	149
6.4	A 1D blob and two 1D-ROLG-filter masks on the blob. In each mask, the white region corresponds to the inner part of the ROLG filter mask, and the gray regions correspond to its surrounding parts.	149
6.5	Response on a 1D edge. (a) A 1D edge. (b) Response of the LoG filter. (c) Response of the ROLG filter.	150

-
- 6.6 Response on a corner. (a) A 2D corner. (b) Absolute value of the LoG response. (c) Absolute value of the ROLG response. 151
- 6.7 A 1D edge and a 1D-ROLG-filter mask on the edge. In this mask, the white region corresponds to the inner part of the ROLG filter mask, and the gray regions correspond to its surrounding parts. 151
- 6.8 Points detected on a single scale. ‘*’ denotes desired point and ‘+’ denotes undesired point. From left to right of (a), (b), (c) and (d) are input images, absolute value of the LoG responses, local extrema of the LoG responses, absolute value of the ROLG responses, and local extrema of the ROLG responses, respectively. 152
- 6.9 Interest points detected by two detectors. (a) SIFT detector, (b) ROLG detector. Local maxima are indicated in blue, and local minima are indicated in red. The radius of the yellow circle is 2 times the scale σ of the interest point at its center. 155
- 6.10 Sensitivity of the offset parameter δ with respect to the detection results. (a) The average number of repeated points (top line) and matched points (lower line). (b) The average value of repeatability (top line) and matching score (lower line). 157
- 6.11 Samples from the 8 data sets of the Oxford database. ‘boat’ and ‘bark’: scale and rotation change. ‘graf’ and ‘wall’: viewpoint change. ‘ubc’: JPEG compression. ‘bikes’ and ‘trees’: image blur. ‘leuven’: lighting change. Two (the 1st and the 6th) of the six images are shown for each data set. The top image in each set is used as the reference image in the experiments. 158

6.12	Number of repeated interest points (a) and number of correct matched interest points (b) on the Oxford database. Each figure contains 8 columns corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.	160
6.13	Repeatability score (a) and matching score (b) on the Oxford database. Each figure contains 8 columns corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.	161
6.14	Sample images in AR, ORL, GT, FERET and LFW databases. They show the typical image variations of the same persons in each database.	163
6.15	Cumulative matching curves on AR database.	164
6.16	Cumulative matching curves on ORL database.	165
6.17	Cumulative matching curves on GT database.	166
6.18	Cumulative matching curves on FERET database.	167
6.19	Cumulative matching curves on LFW database.	168
7.1	(a) input image. (b) an enlarged image patch. (c)&(d) voting for brightness: (c) voting results by the pixels in the surrounding part and (d) voting results by the pixels in the inner part. (e)&(f) voting for darkness: (e) voting results by the pixels in the surrounding part and (f) voting results by the pixels in the inner part. For each voting pixel in (c) to (f), green color represents voting with 1 while blue color means voting with 0. Best viewed in color.	172

7.2	Voting maps of (a) bright regions and (b) dark regions. Best viewed in color. . .	173
7.3	Interest points detected by MSER detector. The circles indicate the scales of the detected local structures.	178
7.4	The VCB against the scale dimension at the center of a blob.	179
7.5	(a)&(e) Input images. (b)&(f) Interest points detected by SIFT detector. (c)&(g) Interest points detected by MSER detector. (d)&(h) Interest points detected by VC detector.	181
7.6	(a) Repeatability results for the Caltech Motorbikes images with a clear background. (b) Repeatability results for the Caltech Motorbikes images with clutter background	182
7.7	Circles represent 10 regions of the highest local maximum of the VC detector detected on images in the Motorbikes database: left are the images clear background and right are the images with clutter in the background.	184
7.8	(a) Number of repeated interest points and (b) number of corrected matched interest points on the Oxford database. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.	185
7.9	(a) Repeatability score and (b) matching score on the Oxford database. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively. .	187

List of Tables

3.1	CRLB and MSE of Sample Mean μ and Median ϕ	47
4.1	MAEs for the Filtered Two-tune Signal Contaminated by α -stable Noise. . . .	96
4.2	MAEs, MSEs, PSNRs and Running Time of the Noise Contaminated “Lena” Image. For the Myriad and Weighted Myriad Filters, the Running Time with $L = 5$ Iterations is Shown in Brackets.	100
5.1	MSEs for Homogeneous Image corrupted by α -Stable Noise.	137
6.1	Rank one recognition rate on AR database.	164
6.2	Rank one recognition rate on ORL database.	165
6.3	Rank one recognition rate on GT database.	166
6.4	Rank one recognition rate on FERET database.	167
6.5	Rank one recognition rate on LFW database.	168
7.1	Recognition Rate on AR, ORL, GT, and FERET Databases.	190

Chapter 1

Introduction

This thesis focuses on developing the median based approaches for noise suppression and interest point detection. The motivations of proposing such methods in these research areas are explained in Section 1.1. Our major contributions of this thesis are listed in Section 1.2. Finally, the organization of this work is described in Section 1.3.

1.1 Motivation

Noise suppression is a fundamental research topic in signal/image processing [8, 9]. In last decades, great effort has been devoted to attenuating noise of images by developing linear and nonlinear filters. The linear filters are designed based on the arithmetic mean. They are widely used because of their easy implementation and the availability of rigorous mathematical tools to analyze them. However, the linear filters cannot cope with impulsive noise, and tend to destroy image details. The abrupt changes of images, such as the edges and boundaries, which are important for machine vision and human perception, are blurred by the linear filters. Therefore, nonlinear filters were developed to preserve the image structures.

The median filter [9] is the most widely used one among the nonlinear filters. It is originated from the robust estimation theory, and provides a powerful tool for signal/image processing because of its good properties in structure preservation and long-tailed noise suppression. How-

ever, it destructs fine signal/image details and cannot effectively suppress additive Gaussian and other short-tailed noise. In order to alleviate these problems, various modifications and extensions of the median filter are developed [8]. Some extensions of the median filter are based on the sorted data to provide more flexibility [10–21]. As the median filter is less effective than the mean filter in attenuating the short-tailed noise, another branch of filters were developed by making compromises between these two filters. Such filters include the STM filter [8], the L filter [8], the α -trimmed mean (α T) filter [22,23], the modified trimmed mean (MTM) filter [24], the mean-median (MEM) filter [25] and the median affine (MA) filter [26]. The commonality of these filters is that their outputs vary smoothly between the mean and median by adjusting some free parameters. However, it is still a challenging task to choose the optimal parameters for different images though efforts were made [26–28]. The α T filter discards some samples evenly from the both sides of the sorted data. This may not be efficient as it does not consider the dispersion of the data [26]. The MTM filter discards the samples outside a neighborhood of the median. This causes this filter sensitive to the slight disturbance of samples near the threshold [26]. The MEM filter combines the mean and median filters linearly, which may not be optimal to attenuate the short- and long-tailed mixed noise. The MA filter's output is a dynamic weighted mean of the input samples. As the MA filter uses a soft-limiting method, it has limitation in suppressing strong impulsive noise. All these filters require both data sorting and arithmetic computing. Compared to the arithmetic computing, the data sorting is much more time consuming [29].

Therefore, nonlinear filters which own the merits of the median filter while do not require data sorting were desirable and proposed in [7, 30]. The Myriad (LogCauchy) filter [30] was designed for a specific distributed noise model, α -stable noise. Its performance highly depends on the tunable “linear parameter” [30] computed from the priori knowledge of the noise distribution. The iterative truncated arithmetic mean (ITM) filter [7] iteratively truncates extreme samples to a dynamic threshold that ensures the filter's output converges to the median. Proper

termination of the iterations makes the ITM filter own merits of the mean and median operations in attenuating the short- and long-tailed noise. However, in case the exclusive noise, such as the impulsive noise, and the additive noise coexists, the performance of the ITM filter decreases. It cannot effectively suppress the mixed short-, long-tailed noise and impulsive noise. Moreover, further analysis shows that the computational complexity of the ITM filter is of order $n\sqrt{n}$. It is not acceptable for a large filter size as it causes heavy computational burden. The merits and the deficiencies of the ITM filter compel us to devote more effort to designing filters based on the ITM filter, and Chapters 3-5 are contributed in developing these filters.

Besides the noise suppression, filters are also widely used in the interest point detection, which is an active research topic in image processing and computer vision. The interest points are used to anchor the local features that are different from their immediate neighborhoods [1]. This research topic has drawn great attentions in the last two decades [1, 5, 31, 32]. Compared with the global features, local features can sparsely represent the image and be robust to occlusion and noisy background. Interest point detectors have been widely used in panoramic image stitching [33], image retrieval [34], image registration [35], texture classification [36], object categorization [37], object recognition [38, 39], 3D object modeling [40], video shot retrieval [41], and face recognition [42]. Many interest point detectors have been proposed in the past few years to detect local structures of images [3, 43–54]. They can be categorized into three types: corner-based, blob-based and region detectors.

Corners correspond to points in the 2D images with high curvature [1]. Harris corner detector [43] uses the second moment matrix to detect the image local structures where the principal intensity changes in two orthogonal directions are both large. However, this type of structures includes not only corners, but also textured patterns and noise [55]. Harris-Laplace/affine detectors [44] were proposed to alleviate the problems caused by scale and affine changes. It uses the Harris corner detector to detect corners in multi-scales, and uses the Laplacian operator to estimate their characteristic scales. As the shape of a corner does not match the shape of the

Laplacian operator, scale estimation for corners is often unstable [56]. SUSAN detector [45] defines a corner as the smallest USAN (univalue segment assimilating nucleus) point, which is dissimilar from a majority of pixels within a neighborhood of it. This detector is sensitive to impulsive noise and blur, and it fails to deal with scale changes.

Blobs refer to bright regions on dark backgrounds or vice versa [57]. Hessian detector [47] employs the Hessian matrix to detect blobs in a single scale. Hessian-Laplace/affine detectors [44] were developed to detect blobs in multiple scales based on the Hessian detector and the Laplacian operator. These detectors are stable in estimating the characteristic scales of blobs, of which the shapes are similar to that of the Laplacian operator. SIFT detector [48] employs the difference of Gaussian (DoG) filter to approximate the normalized Laplacian of Gaussian (LoG) filter. The DoG filter significantly accelerates the computation process. SURF detector [49] employs the box filters and the integral images to further speed up the Hessian-Laplace detector. The box filters are approximations of the second order Gaussian derivative filters. The integral images allow for the fast convolutions of the box filters with the input image. Different approaches, which are not based on the second order derivative of image intensity, were proposed in [3, 50, 51] to detect blobs. Salient region detector [50] employs the image local complexity to detect blobs. The characteristic scales are determined by the entropy extrema of the local descriptors. A common computational concept is proposed in [3] to detect local structures of different types. The intensity variance in a local circular region is divided into three components, which are used to detect corners, blobs and high textured structures. A histogram-based similarity measure is introduced in [51] to bridge the gap between interest point detectors and descriptors.

Region detectors extract regions with similar image structures and properties [1]. Edge-based region detector [52] uses the Harris points as the initial points, and uses the two nearby edges to determine a parallelogram region. Intensity-based region detector [52] employs the local intensity extrema as the initial points. The affine invariant regions are determined by the

significant changes of the intensity profiles, which are along rays going out of the extrema. MSER detector [53] starts from the local intensity extrema. The maximally stable extremal regions are extracted using a watershed like segmentation algorithm. The MSER detector works well for structured images, which have strong intensity changes on region boundaries. But it is sensitive to image blur which undermines the stability criterion [31].

Among these detectors, the most famous one is the SIFT detector [38,48] which had gotten 22602 citations till January 2013 according to Google scholar. This detector is based on the difference of Gaussian (DoG) filter which approximates to the linear LoG filter. The drawback that linear filters cannot effectively suppress the impulsive noise causes the DoG filter sensitive to the abrupt variations of the image. The response of the DoG filter can be easily affected by the strong and abrupt structures near the structure to be detected. This makes the SIFT detector unstable to detect the low contrast image structures, where other strong and abrupt structures caused by illumination or geometric changes partially fall in the detection window. Moreover, due to the second order derivative nature of the SIFT detector, many unstable and spurious points are often detected around the structures. Contrary to the linear filters, the median based filters can effectively suppress the impulsive noise. This enables the median based approaches robust to the abrupt variations. The advantage of the median filter motivates us to develop the interest point detectors based on the median operation, and Chapters 6-7 are devoted in designing these detectors.

1.2 Contributions

The major contributions of this thesis are summarized as follows.

- We further analyze the properties of the recently proposed ITM filter. Both theoretical and experimental results show that this filter suppresses both Gaussian and Laplacian noise more effectively than the median filter. We propose a fast realization of the ITM filter

based on the theoretical analysis, and study its computational complexity. The proposed realization is faster than the median filter.

- A rich class of filters named weighted ITM (WITM) filters are proposed. By iteratively truncating the extreme samples, the output of the WITM filter converges to the weighted median. Proper stopping criterion makes the WITM filters own merits of both the weighted mean and median filters and hence outperforms the both in some applications. Three structures are designed to enable the WITM filters being low-, band- and high-pass filters. Properties of these filters are presented and analyzed.
- An iterative trimmed and truncated arithmetic mean (ITTM) algorithm is proposed. Here, trimming a sample means removing it, while truncating a sample is to replace its value by a threshold. The proposed trimming and truncating rules ensure that the ITTM filter's output moves from the sample mean and toward the sample median by increasing the number of iterations. Proper termination of the iteration makes the ITTM filter own some merits of both the mean and median filters. Compared to the median and ITM filters, the ITTM filter suppresses noise more effectively in some cases and has lower computational complexity.
- A novel nonlinear filter, named rank order Laplacian of Gaussian (ROLG) filter, is proposed. An interest point detector is developed based on this filter. The ROLG filter is a weighted rank order filter. It is used to detect the image local structures where a significant majority of pixels are brighter or darker than a significant majority of pixels in their corresponding surroundings. Compared to linear filter based detectors, e.g. SIFT detector, the proposed rank order filter based detector is more robust to abrupt variations of images caused by illumination and geometric changes.
- A vote of confidence (VC) based detector is proposed to detect the interest points. Whether a local region is bright or dark is voted by all the pixels in this region and its surround-

ings. Compared to the intensity based detectors, the VC detector is robust to illumination changes.

1.3 Organization of the Thesis

The remainder of this thesis is organized as follows.

1. In Chapter 2, the representative approaches of the median based filters are reviewed. Their advantages and disadvantages are presented. Besides, a detailed review of the interest point detectors is presented.
2. In Chapter 3, further properties of the ITM filter are analyzed and a fast implementation is proposed.
3. In Chapter 4, a rich class of filters named weighted ITM (WITM) filters are proposed.
4. In Chapter 5, an iterative trimmed and truncated arithmetic mean (ITTM) filter is proposed.
5. In Chapter 6, the rank order Laplacian of Gaussian (ROLG) filter is proposed. The detector based on the proposed filter is developed.
6. In Chapter 7, we propose a vote of confidence (VC) based interest point detector.
7. In Chapter 8, we conclude this thesis by highlight the contributions of this work. The possible future work is discussed following.

Chapter 2

Literature Review

In this chapter, a detailed literature review of the median based approaches in noise suppression is presented. Besides, a comprehensive literature review of the interest point detectors is given.

2.1 Filters in Noise Suppression

Vast amounts of filters have been developed to suppress noise in last decades. As this thesis focuses on the median based approach, in this section we mainly review the filters based on the median operation. Besides, the filters based on soft limitation are also reviewed and discussed here for the great attention paid by the researchers.

2.1.1 Median Based Filters

In this section, the two fundamental filters (mean and median filters) are reviewed firstly. The filters which are built on these two operations, including the α -trimmed mean (α T) filter [22, 23], the modified trimmed mean (MTM) filter [24], the mean-median (MEM) filter [25] and the median affine (MA) [26] filter are reviewed following. Besides, the optimal myriad (OM) filter [30], the mean-LogCauchy (MLC) filter [25] and the iterative truncated arithmetic mean (ITM) filter [7], which own the merits of the median filter but not rely on the data sorting, are also reviewed.

2.1.1.1 Mean and Median Filters

The sample mean is the maximum likelihood (ML) estimate of location for data sets with Gaussian distribution. Assume a filter window contains n independent Gaussian distributed samples as $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ with unknown constant mean μ_o and variance σ^2 . The ML estimate of location μ_o is to find the value of μ_l that maximizes the likelihood function

$$\begin{aligned} L(\mu_l | x_1, x_2, \dots, x_n) &= \prod_{i=1}^n f(x_i | \mu_l) \\ &= \left(\prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \right) \exp \left(- \sum_{i=1}^n \frac{(x_i - \mu_l)^2}{2\sigma^2} \right). \end{aligned} \quad (2.1)$$

It is equivalent to minimizing the squares sum

$$G_2(\mu_l) = \sum_{i=1}^n \frac{(x_i - \mu_l)^2}{\sigma^2}. \quad (2.2)$$

The value of μ_l minimizing (2.2) is the mean μ

$$\begin{aligned} \mu &= \arg \min_{\mu_l} G_2(\mu_l) \\ &= \text{mean}(\mathbf{x}_0). \end{aligned} \quad (2.3)$$

μ is the optimal estimate of μ_o because its variance equals to the Cramer-Rao lower bound (CRLB) [58]. Similarly, the ML estimate of location μ_o under Laplacian distribution is equivalent to minimizing

$$G_1(\mu_l) = \sum_{i=1}^n \frac{|x_i - \mu_l|}{\sigma}. \quad (2.4)$$

The value of μ_l that minimizes (2.4) is the median ϕ

$$\begin{aligned} \phi &= \arg \min_{\mu_l} G_1(\mu_l) \\ &= \text{median}(\mathbf{x}_0). \end{aligned} \quad (2.5)$$

As the mean square error (MSE) of the median is larger than the CRLB under Laplacian distribution, it does not achieve the minimum MSE though it is the ML estimate [59].

2.1.1.2 Alpha-trimmed Mean Filter

Although the mean and median filters are effective in suppressing Gaussian and Laplacian noise, respectively, none of them can effectively attenuate these two types of noise simultaneously. Filters that can effectively suppress both the short- and long-tailed noise are desirable. The representative one that makes a compromise between the mean and median filters is the alpha-trimmed mean (α T) filter [22, 23]. It is summarized as follows.

The α T filter is operated on the sorted data $\mathbf{x} = \{x_{(1)}, x_{(2)}, \dots, x_{(n)}\}$ with $x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(n)}$. This filter removes a fixed fraction α ($0 \leq \alpha \leq 0.5$) of samples from the two ends of the ascending sorted data and uses the mean of the remaining samples as the filter output, i.e.

$$y_\alpha = \frac{1}{n - 2[\alpha n]} \sum_{i=[\alpha n]+1}^{n-[\alpha n]} x_{(i)}, \quad (2.6)$$

where $[x]$ returns the smallest integer which is larger than or equal to x . The output of the α T filter equals to the mean if $\alpha = 0$ and equals to the median if $\alpha = 0.5$. As the α T filter removes samples strictly based on their orders without considering its dispersion, this leads to loss of fidelity and inefficiency in some cases [26].

2.1.1.3 Modified Trimmed Mean Filter

Different from the α T filter removing samples strictly relying on their orders, the modified trimmed mean (MTM) filter [24] provides more flexibility by removing samples with a fixed threshold λ . Let ϕ be the median of the input data set $\mathbf{x}_0$, and $\mathbf{x}_r = \{x_i | x_i \in \mathbf{x}_0, -\lambda \leq x_i - \phi \leq \lambda\}$. The MTM filter is defined as the mean of the samples in $\mathbf{x}_r$, i.e.

$$y_{\text{MTM}} = \text{mean}(\mathbf{x}_r). \quad (2.7)$$

As the parameter λ is a predetermined threshold, the filter's performance is highly depended on the chosen of λ [26]. Moreover, the MTM filter is serious sensitive to the slight disturbance of the samples near the threshold [7].

2.1.1.4 Mean-Median Filter

As the median filter is a selected type filter, its output is always a sample from the input data set. This makes the median filter more limited to suppress the impulsive noise. In order to overcome this limitation, the mean-median (MEM) filter [25,27] is proposed to make a compromise between these two filters. This filter is developed to deal with the ϵ -contaminated normal distribution given by Huber [60]

$$P_\epsilon = \{(1 - \epsilon)\Phi - \epsilon H\}, \quad (2.8)$$

where Φ is Gaussian distribution and H is Laplacian distribution. It is defined as

$$y_{\text{MEM}} = (1 - \lambda)\text{mean}(\mathbf{x}_0) + \lambda\text{median}(\mathbf{x}_0), \lambda \in [0, 1]. \quad (2.9)$$

The parameter λ is optimized by minimizing the asymptotic variance [25]. Linear combination of the median and mean filters may make the MEM filter not optimal in suppressing noise with different degree of impulsiveness.

2.1.1.5 Median Affine Filter

Instead of trimming the samples according to their orders (in α T filter), the median affine (MA) filter [26] employs a soft-limiting affinity measurement to control the influence of individual samples. Its output varies from the linear FIR filter to the median filter by adjusting a simple affinity function. The filter output is defined as

$$y_{\text{MA}} = \sum_{i=1}^n \mathcal{A}_{(i)}^{\phi, \gamma} x_{(i)} / \sum_{i=1}^n \mathcal{A}_{(i)}^{\phi, \gamma}, \quad (2.10)$$

where $\mathcal{A}_{(i)}^{\phi, \gamma} = e^{-((x_{(i)} - \phi)^2 / \gamma)}$ is the Gaussian affine function used to control the influence of each sample. Setting the parameter γ is important in balancing the filter's behavior of a linear FIR filter and a robust median filter. An adaptive optimization was given in [26] under the mean square error (MSE) criterion. Compared to the median and α T filters, the MA filter has limitations in attenuating impulsive noise because it employs a soft-limiting instead of removing algorithm.

2.1.1.6 Optimal Myriad Filter

The optimal myriad (OM) filter [30] was proposed to suppress the non-Gaussian impulsive noise that can be represented by the α -stable noise. The characteristic function of the symmetric α -stable noise is

$$\phi(\omega) = e^{-\gamma|\omega|^\alpha}, \quad (2.11)$$

where $\alpha \in (0, 2]$ determines the tail heaviness or impulsiveness of the distribution and γ is the dispersion proportional to the scale of the distribution. The closed-form expressions for the density functions of the α -stable noise are known only for $\alpha = 1$ and $\alpha = 2$, which are corresponding to the Cauchy and Gaussian distribution, respectively. The density function of the Cauchy distribution is

$$f(x|\mu_l, \gamma) = \frac{\gamma}{\pi} \frac{1}{\gamma^2 + (x - \mu_l)^2}, \quad (2.12)$$

where μ_l is the location to be estimated from the input data $\mathbf{x}_0$. The myriad estimate is to find μ_l which maximizes the likelihood function

$$\begin{aligned} y_{\text{OM}} &= \text{myriad}\{k; x_1, \dots, x_n\} \\ &= \arg \max_{\mu_l} \prod_{i=1}^n \frac{1}{k^2 + (x_i - \mu_l)^2}, \end{aligned} \quad (2.13)$$

where $k = \gamma$ for the Cauchy distribution. As there is no closed-form solution for (2.13), an iterative algorithm is given in [30] to find the solution. For the α -stable noise with the parameter (α, γ) , the tunable “linear parameter” $k(\alpha) = \sqrt{\alpha/(2 - \alpha)}\gamma^{1/\alpha}$ is used to achieve a consistently

efficient result for all α values. The performance of the OM filter is highly dependent on the linear parameter $k(\alpha)$. Moreover, solving the equation (2.13) is time consuming with the order of $\mathcal{O}(n^2)$ [61].

2.1.1.7 Mean-LogCauchy Filter

As the mean and LogCauchy (also named myriad) filters are the optimal filters in suppressing the Gaussian and Cauchy distributed noise, respectively. The linear combination of these two filters, named mean-LogCauchy (MLC) filter [25], is developed to deal with the following noise probability distribution

$$P_\epsilon = \{(1 - \epsilon)\Phi - \epsilon S\}, \quad (2.14)$$

where Φ is Gaussian distribution and S is the Cauchy distribution with dispersion γ . The corresponding MLC filter is defined as

$$y_{\text{MLC}} = (1 - \lambda)\text{mean}(\mathbf{x}_0) + \lambda\text{LC}_\gamma(\mathbf{x}_0), \lambda \in [0, 1], \quad (2.15)$$

where $\text{LC}_\gamma(\mathbf{x}_0)$ is the LogCauchy (LC) filter defined as

$$\text{LC}_\gamma(\mathbf{x}_0) = \arg \max_{\mu_l} \prod_{i=1}^n \frac{1}{\gamma^2 + (x_i - \mu_l)^2}. \quad (2.16)$$

The MLC filter is generalized in [25] to suppress the α -stable noise with dispersion γ and impulsiveness α .

2.1.1.8 Iterative Truncated Arithmetic Mean Filter

The iterative truncated arithmetic mean (ITM) filter [7] iteratively truncates the extreme samples in the input data set to a dynamic threshold that ensures the filter's output converges to the median. The proposed stopping criterion of the ITM filter makes it own merits of both the arithmetic mean and order statistic median operations in attenuating the short- and long-tailed noise. The ITM algorithm, starting from $\mathbf{x} = \mathbf{x}_0$, iteratively truncates the extreme samples in $\mathbf{x}$

to a dynamic threshold. This algorithm includes the following steps.

1. Compute the sample mean: $\mu = \text{mean}(\mathbf{x})$;
2. Compute the dynamic threshold: $\tau = \text{mean}(|\mathbf{x} - \mu|)$;
3. $b_l = \mu - \tau$, $b_u = \mu + \tau$, and truncate $\mathbf{x}$ by:

$$x_i = \begin{cases} b_u, & \text{if } x_i > b_u \\ b_l, & \text{if } x_i < b_l ; \\ x_i, & \text{otherwise} \end{cases}$$

4. Terminate the iteration if the stopping criterion S is met. Otherwise, return to Step 1.

The ITM filter has two types of outputs [7]. The type I output ITM1 is

$$y_{t1} = \text{mean}(\mathbf{x}). \quad (2.17)$$

Let $\mathbf{x}_r = \{x_i | b_l < x_i < b_u\}$ and n_r be the number of samples in $\mathbf{x}_r$. The type II output ITM2 is

$$y_{t2} = \begin{cases} \text{mean}(\mathbf{x}_r), & \text{if } n_r > \xi \\ \text{mean}(\mathbf{x}), & \text{otherwise} \end{cases}. \quad (2.18)$$

The parameter ξ is used to avoid an unreliable mean in case too few samples remain in $\mathbf{x}_r$. It is set to $\xi = n/4$ in [7].

The ITM filter's output moves from the mean towards the median by increasing the number of iterations. Proper stopping criterion is required to terminate the iterations. The stopping criterion S proposed in [7] is designed for general cases, and includes 4 parts. The first one terminates the iteration if the truncated mean is close to the median. It is

$$S_1(\varepsilon_1) : \Delta n \triangleq |n_h - n_l| \leq \varepsilon_1. \quad (2.19)$$

If $S_1(1)$ is met, no sample exists between the truncated mean and the median. The second part S_2 uses a predefined maximum number ε_2 to limit the number of iterations k

$$S_2(\varepsilon_2) : k \geq \varepsilon_2. \quad (2.20)$$

The third part S_3 is to handle edges

$$S_3(\varepsilon_3) : n_\tau \triangleq |n_{\tau u} - n_{\tau l}| \geq \varepsilon_3. \quad (2.21)$$

An auxiliary condition S_4 is used to avoid an immature stop

$$S_4 : n_\tau(k) = n_\tau(k-1). \quad (2.22)$$

A sophisticated stopping criterion S chosen in [7] is a combination of the above, that is

$$S = S_1(1) \vee S_2(2\sqrt{n}) \vee S_3[(n - \sqrt{n})/2] \vee [S_3(\sqrt{n}) \wedge S_4]. \quad (2.23)$$

2.1.1.9 Discussion

The mean and median are the maximum likelihood estimates of the location under Gaussian and Laplacian distributed noise, respectively. However, neither the mean nor median filters can effectively suppress both Gaussian and Laplacian noise. The median filter loses 40% efficiency in suppressing Gaussian noise compared to the mean filter, while the performance of the mean filter is poorer than the median filter in suppressing Laplacian noise. The performance of the mean filter becomes even poorer if the strong impulsive noise exists. Therefore, filters which can effectively suppress these two types of noise were proposed by making a compromise between the mean and median filters. Some of these filters firstly trim some extreme samples, and then use the average of the remaining ones as the filter's output. Such filters include the α T and MTM filters. As both the α T and MTM filters trim the samples without consider the dispersion of the data, they may lose efficiency in some cases. Moreover, as the MTM filter uses a fixed threshold to remove the extreme samples, slight variation of the samples close to the threshold

may greatly affect the filter's output.

Instead of directly removing samples according to their orders, the MA filter employs a soft-limiting affinity measurement to control the influence of samples around the median. Choosing the parameter that controls the affinity measurement is important in balancing the filter's behavior from a linear filter to a median filter. Soft-limiting algorithm may reduce the filter's capability in suppressing the strong impulsive noise. The OM filter's output is the maximum likelihood estimate under the Cauchy distribution. There is no closed-form expression to calculate the filter's output. The iterative algorithm given in [61] is of order $\mathcal{O}(n^2)$. It is drastically larger than the order of the median filter, which is $\mathcal{O}(n \log n)$. The OM filter employs a tunable "linear parameter" $k(\alpha)$ to make this filter efficient for the α -stable noise with different α values. This causes the performance of this filter highly dependent on the parameter $k(\alpha)$. The MEM and MLC filters were developed to deal with the ϵ -contaminated normal distribution. The MEM filter was designed to suppress the Laplacian contaminated Gaussian noise, and the MLC filter was developed to attenuate the α -stable contaminated Gaussian noise. As the MEM filter linearly combines the mean and median, this reduces the filter's capability in suppressing the impulsive noise. Combining the mean filter with the LC filter also makes the MLC filter less effective in suppressing the impulsive noise.

Instead of using the data sorting, the ITM filter [7] employs an iterative truncating algorithm to approximate the median. By iteratively truncating the extreme samples in the input data set to a dynamic threshold, the filter's output starts from the mean toward the median. The proposed stopping criterion of the ITM filter makes it own merits of both the arithmetic mean and order statistic median operations in attenuating the short- and long-tailed noise.

2.1.2 Soft-limiting Filters

Some state-of-art filters which do not rely on data sorting, including the anisotropic filter [62], the total variation minimization filter [63], the SUSAN filter [45], the bilateral filter [64] and

the non-local mean filter [65], are reviewed in this section.

2.1.2.1 Anisotropic Filter

The anisotropic filter (AF) [62] reduces the blurring effect by only convolving the images along the direction orthogonal to the gradient. It is defined as

$$y_{AFh}(\mathbf{x}) = \int G(t, h) I(\mathbf{x} + t \frac{DI(\mathbf{x})^\perp}{|DI(\mathbf{x})|}) dt, \quad (2.24)$$

where $G(t, h)$ is the one-dimensional Gauss function with variance h^2 , $\mathbf{x}$ is coordinate of the corresponding pixel, $I(\mathbf{x})$ is the pixel intensity, $DI(\mathbf{x})$ is the gradient and $DI(\mathbf{x})^\perp$ is orthogonal to $DI(\mathbf{x})$. The advantage is that the straight edge is well preserved by this filter. However, textures and flat regions are still degraded [65].

2.1.2.2 Total Variation Minimization Filter

Instead of minimizing the variation of local region, the total variation minimization filter [63] intends to minimize the total variation using the following objective function

$$\text{TVF}_\lambda(v) = \arg \min_u \text{TV}(u) + \lambda \int |v(\mathbf{x}) - u(\mathbf{x})|^2 d\mathbf{x} \quad (2.25)$$

where v is the observed value, u is the “true” value, λ is the corresponding Lagrange multiplier and $\text{TV}(u)$ represents the total variation of u . There exists one unique solution for the objection function (2.25). The parameter λ is related to the noise statistics, and is used to control the degree of smoothness [65].

2.1.2.3 SUSAN Filter

In order to avoid blurring the image structures and edges, the SUSAN filter [45] averages the pixels which have a similar gray value and a close distance to the reference pixel at the location

$\mathbf{x}$. It is defined as follows

$$y_{\text{SUSAN}} = \frac{1}{C(\mathbf{x})} \int_S I(\mathbf{y}) e^{-\frac{|\mathbf{y}-\mathbf{x}|^2}{\rho^2}} e^{-\frac{|I(\mathbf{y})-I(\mathbf{x})|^2}{h^2}} d\mathbf{y}, \quad (2.26)$$

where S is the local region, $C(\mathbf{x}) = \int_S e^{-\frac{|\mathbf{y}-\mathbf{x}|^2}{\rho^2}} e^{-\frac{|I(\mathbf{y})-I(\mathbf{x})|^2}{h^2}} d\mathbf{y}$ is the normalization factor, ρ is a spatial filtering parameter and h is a intensity filtering parameter. As this filter only averages the pixels which are similar to the pixel in the local region center, it keeps the image structures and does not blur the edges. The deficiency of the SUSAN filter is that it is not robust to the long-tailed noise. Artificial shocks can also be generated by this filter [65].

2.1.2.4 Bilateral Filter

The bilateral filter [64] processes the images using the information in both the spatial and intensity domains. It is defined as

$$y_{\text{BL}}(\mathbf{x}) = k^{-1}(\mathbf{x}) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} I(\mathbf{y}) c(\mathbf{y}, \mathbf{x}) s(I(\mathbf{y}), I(\mathbf{x})), \quad (2.27)$$

where $k(\mathbf{x}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} c(\mathbf{y}, \mathbf{x}) s(I(\mathbf{y}), I(\mathbf{x}))$ is the normalization function, $c(\mathbf{y}, \mathbf{x})$ is used to measure the spatial closeness, and $s(I(\mathbf{y}), I(\mathbf{x}))$ is used to measure the intensity similarity between the pixels at $\mathbf{y}$ and $\mathbf{x}$. The bilateral filter turns to the SUSAN filter if choosing the Gaussian function to measure the similarities in both the spatial and intensity domains.

2.1.2.5 Non-local Mean Filter

Instead of only using the samples in a local region, the non-local (NL) mean filter [65] uses the similar pixels in the whole image to suppress noise. It is defined as

$$y_{\text{NL}}(\mathbf{x}) = \sum_{\mathbf{y} \in \Omega} w(\mathbf{x}, \mathbf{y}) I(\mathbf{y}), \quad (2.28)$$

where Ω represents the whole spatial domain of the image, and $w(\mathbf{x}, \mathbf{y})$ is the similarity of two pixels at $\mathbf{x}$ and $\mathbf{y}$. This similarity $w(\mathbf{x}, \mathbf{y})$ is measured by the similarity of the pixel vectors $I(\mathcal{S}_{\mathbf{x}})$

and $I(\mathcal{S}_y)$ in their corresponding local regions, where $\mathcal{S}_x$ and $\mathcal{S}_y$ represent a fix-sized square neighborhood with the central pixel $\mathbf{x}$ and $\mathbf{y}$, respectively. $w(\mathbf{x}, \mathbf{y})$ satisfies that $\sum_{\mathbf{y}} w(\mathbf{x}, \mathbf{y}) = 1$ and $0 \leq w(\mathbf{x}, \mathbf{y}) \leq 1$.

By searching over the whole image and using the weighted average of all similar pixels as the filter's output, the NL-mean filter has a better performance in keeping image structures and suppressing short-tailed noise. However, it cannot effectively suppress the long-tailed noise and has a high computational complexity.

2.1.2.6 Discussion

The soft-limiting filters reviewed in this part don't require data sorting. Most of them are faster than the median filter. As the non-local mean filter needs search the similar image patches over the whole image, its speed is slower than the median filter. The anisotropic filter convolves the image only along the edges or structure boundaries. It can keep the edges well. However, for the texture patterns and the flat regions, the performance of this filter degrades as these regions do not have explicit orientations. Both the SUSAN and bilateral filters use the average of the pixels, which are similar and close to the reference pixel, as the filter outputs. They can keep the details and structures well. The negative effect of relying on the reference pixel is that these filters are not robust to the long-tailed and impulsive noise. The non-local mean filter inherits this deficiency as it also uses the pixels which are similar to the reference pixel to estimate the image.

2.2 Interest Point Detection

Based on the detected local image patterns, the interest point detectors can be grouped into three categories: corner, blob and region detectors. In the following, typical detectors in each category are described and their properties are analyzed.

Figure 2.1: (a) Input image. (b) Response of the Moravec corner detector.

2.2.1 Corner Detectors

In this part, the following detectors are reviewed: (1) the Moravec detector [66], (2) the Harris detector [43], (3) the Harris Laplace/affine detector [44], (4) the smallest univalue segment assimilating nucleus (SUSAN) detector [45], (5) the features from accelerated segment test (FAST) detector [67, 68], and (6) the Pader detector [69].

2.2.1.1 Moravec Detector

The Moravec detector [66] was developed in 1977. It is one of the earliest corner detectors. This detector employs the local intensity changes in eight directions (horizontal, vertical, and four diagonals) to measure the dissimilarity between the nearby image patches. The intensity change has the following regularity: (1) if the pixel is in a uniform region, the intensity changes in all directions are small; (2) if the pixel is on an edge, the intensity change along the edge is small and that across the edge is large; (3) if the pixel is on a corner, the intensity change in any direction is large. Therefore, the minimum intensity change among those in the eight directions is chosen to form the corner map. The local maxima are detected from the corner map, and used as the interest point candidates. Fig. 2.1(b) shows that large responses have been generated at the corner by the Moravec detector. However, as this detector only considers the

Figure 2.2: Image structures and their corresponding gradient fields, from (a) to (d) are corner, textured structure, gradient field of the corner, and gradient field of the textured structure, respectively.

intensity changes at every 45° , the detector is anisotropic. The response of the edge is large if the angle of the edge direction is not an integer multiple of 45° . Using the rectangular window to crop the image patch also causes the response anisotropic.

2.2.1.2 Harris Detector

The Harris detector [43] is developed based on Movarvec detector [66]. In order to overcome the weakness of anisotropic, intensity changes in all directions are taken into consideration. With the help of Taylor expansion, the structure tensor matrix is derived from the local intensity

changes. The two eigenvalues of the tensor matrix, λ_{max} and λ_{min} , are proportional to the principal curvatures of the local structures. They have the following properties: (1) if the local region is flat, both eigenvalues are small; (2) if the local region is on the edge, λ_{max} is drastically larger than λ_{min} ; (3) if the local region is on the corner, both eigenvalues are large. The structure tensor matrix is defined as

$$M(\mathbf{x}) = \sigma_D^2 G(\mathbf{x}, \sigma_I) * \begin{bmatrix} I_x^2(\mathbf{x}, \sigma_D) & I_x(\mathbf{x}, \sigma_D)I_y(\mathbf{x}, \sigma_D) \\ I_x(\mathbf{x}, \sigma_D)I_y(\mathbf{x}, \sigma_D) & I_y^2(\mathbf{x}, \sigma_D) \end{bmatrix}, \quad (2.29)$$

where $*$ is the convolution operator. The corner map is generated by

$$cornerness(\mathbf{x}) = \det(M(\mathbf{x})) - \kappa \text{trace}^2(M(\mathbf{x})), \quad (2.30)$$

where κ is used to suppress the response of the edge.

The Harris detector is the most repeatable one [32], and is optimal for the L junction. However, it is not optimal for other junctions, such as the T junction of which the principal curvature in one direction is larger than that in the orthogonal direction [70]. Moreover, as the related position information of the gradient is ignored in creating the tensor matrix (2.29), the Harris detector is sensitive to texture structure. Fig. 2.2 shows two image patterns, one is a corner and the other is a textured pattern. Although they have significantly different gradient fields, as shown in Fig. 2.2(c) and Fig. 2.2(d), their tensor matrices are almost the same.

2.2.1.3 Harris-Laplace/Affine Detector

Harris-Laplace detector [44] detects corners in both spatial and scale dimensions. The corner map is generated by

$$cornerness(\mathbf{x}, \sigma_I, \sigma_D) = \det(M(\mathbf{x}, \sigma_I, \sigma_D)) - \kappa \text{trace}^2(M(\mathbf{x}, \sigma_I, \sigma_D)), \quad (2.31)$$

where σ_D is the differential scale and σ_I is the integral scale. It is used to detect corners in the spatial domain. By adjusting σ_D and σ_I , corners in multi-scales are detected. The characteristic

scale is selected with the help of the scale-normalized Laplacian operator, defined as

$$|LoG(\mathbf{x}, \sigma_I)| = \sigma_I^2 |\Delta L(\mathbf{x}, \sigma_I)| = \sigma_I^2 |L_{xx}(\mathbf{x}, \sigma_I) + L_{yy}(\mathbf{x}, \sigma_I)|, \quad (2.32)$$

where L_{xx} and L_{yy} represent the second order derivatives of the intensity along x direction and y direction, respectively. For a point $\mathbf{x}$ at the scale σ_I , if $cornerness(\mathbf{x}, \sigma_I, \sigma_D)$ is the local maximum in the spatial domain and $|LoG(\mathbf{x}, \sigma_I)|$ is the local maximum in the scale domain, $\mathbf{x}$ is detected as a corner point with the characteristic scale σ_I . The scale selected by the Laplace maximum is not well defined for the corner because the corner structure changes very small over a large range along the scale dimension [1].

Harris affine detector is proposed to enhance the performance of the Harris Laplace detector in dealing with affine variation. The affine invariant corners are obtained by iteratively normalizing the affine regions until the corresponding Harris matrix converges to an identity matrix [71]. The procedures are as follows: (1) detect the initial regions by the Harris Laplace detector; (2) estimate the affine shape based on the Harris matrix, and normalize the affine shape to a circular one; (3) re-detect the new location and scale in the normalized image; (4) go to step 2 if the eigenvalues of the Harris matrix are not equal [1, 44].

2.2.1.4 SUSAN Corner Detector

The SUSAN detector was proposed by Smith and Brady [45]. Instead of using the local gradient, this method uses the number of pixels, which are similar to the nucleus, to generate the corner map. The similar pixels are counted in each local region, and the segment consisted by these similar pixels is named "univalue segment assimilating nucleus (USAN)", as shown in Fig. 2.3. This similarity, which refers to the number of similar pixels, has the following properties: (1) if the local region is flat, the similarity is high; (2) if the local region is on the edge, the similarity is half of the mask's area size; (3) if the local region is on the corner, the similarity is small. Thus, if a local minimum of the similarity map is below a certain threshold, it implies

Figure 2.3: USAN area of the SUSAN detector: (a) original image; (b) USAN areas of 5 circular masks, in which the white area indicate the USAN area.

that a corner exists. The function to calculate the similarity is

$$m(\mathbf{x}) = \sum_{\mathbf{y}} \exp\left(-\left(\frac{I(\mathbf{y}) - I(\mathbf{x})}{t}\right)^6\right), \quad (2.33)$$

where $\mathbf{y}$ belongs to the neighborhood with the center of $\mathbf{x}$, and t is a brightness difference threshold. This similarity has low response on thin lines. In order to reject this false positive, the centroid and the contiguity of the USAN are utilized [45]. The performance of this detector becomes poor in dealing with blurred images, in which the similar and dissimilar pixels are difficult to judge [1].

2.2.1.5 FAST: Features from Accelerated Segment Test

The FAST detector is proposed by Rosten et. al. [67, 68]. This detector is a relaxed version of the SUSAN detector. Instead of comparing all the pixels in the circular disk with the center pixel (nucleus), only the pixels on a Bresenham's circle of radius 3.4 pixels are compared with it. In this way, 16 pixels are needed to compare with the nucleus for a full segment test, as shown in Fig. 2.4. The length S of the connected pixels on the circle, which are all darker or brighter than the nucleus, is used to detect corners. The FAST detector is based on the following properties: (1) if the local region is flat, almost no pixels on the Bresenham's circle are dissimilar to the nucleus, thus S is small, e.g., $S = 0$ for the mask B in Fig. 2.4; (2) if the local region is on

Figure 2.4: Illustration for the FAST detector. Three Bresenham's circles are shown here: nucleus A is on the corner, nucleus B is on the flat area, and nucleus C is on the edge.

the edge, nearly half of the pixels on the Bresenham's circle are dissimilar to the nucleus, thus S is near half of the circular length, e.g., $S = 7$ for the mask C in Fig. 2.4; (3) if the local region is on the corner, more than half of the pixels on the Bresenham's circle are dissimilar to the nucleus, thus S is large, e.g., $S = 11$ for the mask A in Fig. 2.4. If S is larger than a threshold S_r , then a corner is considered to be occurred. S_r is used to constrain the maximum angle of the detected corner. ID3, a machine learning method, is employed to accelerate the comparison process. Non-maximum suppression method is used to remove points which have adjacent corners. This method can achieve a high computational speed. However, this method is sensitive to noise because only a few pixels on a Bresenham's circle are used to make the decision. It is difficult for the FAST detector to detect complex structure for it needs a long dissimilar chain. The training process makes the detector highly dependent on the train data, and small rotation of the camera may destroy the performance [72].

Figure 2.5: Corners detected by the Paler detector: (a) corners detected on an artificial image; (b) corners detected on this image polluted by pepper&salt noise.

2.2.1.6 Paler Detector

The Paler detector [69] is proposed based on the fact that the median filter cannot keep the image details. The vertex of the corner will be modified if the median filter is used to process the image. Therefore, it is reasonable to use the difference I_d of the median filtered image I_m and the original image I , $I_d = I_m - I$, as the corner map. As the difference of these two images functions as a high pass filter, the Paler detector is seriously sensitive to noise, e.g. in Fig. 2.5.

2.2.1.7 Discussion

Several typical corner detectors are described in this part. Corner has the advantage of high localization accuracy. This makes it suitable for camera calibration or 3D reconstruction. However, its scale is difficult to define because corner structures have little change over a wide range of scale.

The Moravec, Harris and Harris-Laplace/Affine detector are based on the intensity variation

calculated by linear filters. The SUSAN and FAST detectors are based on the similarity between the central pixel and pixels in its neighborhoods. The Paler detector is based on the median filter. The Harris detector is identified as the most stable one and especially has high response on the L junction. Harris Laplace/affine detectors are scale and affine invariant extensions of the Harris detector. Commonly, the affine transformation holds for viewpoint changes if the local region is a planar region and relatively far from the camera. However, corners are often detected on the boundary of the object where the image patch is not planar region, and hence the viewpoint invariance is limited.

The SUSAN detector has a lower computational complexity than the Harris detector. However, it cannot effectively deal with the blurred structures and it is sensitive to noise. The FAST detector is an optimized version of the SUSAN detector. This detector is sensitive to noise and it is difficult to detect the intricate image patterns. The Paler detector employs a high-pass filter to detect corners. This makes it more sensitive to noise.

2.2.2 Blob Detectors

Blob is another important structure in images and some work has done in this area. The following blob detectors are reviewed in this section: (1) the Hessian detector [47], (2) the Hessian Laplace/Affine detector [44], (3) the SIFT detector [38,48], (4) the speeded-up robust features (SURF) detector [49], (5) the salient region detector [50,73], and (6) the self-similarity detector [3].

2.2.2.1 Hessian Detector

Hessian detector [47], which is used to detect blobs, was proposed by Beaudet in 1978. As shown in Fig. 2.6, the Gaussian curve is a 1D blob structure. The first order derivative equals to zero while the second order derivative reaches the maximum at the peak of the blob. Hence, the second order derivative is used to detect the blob structures. The Hessian matrix, formed by

Figure 2.6: Different order derivatives of Gaussian curves: g represents the Gaussian curve, gx represents the first order derivative of the Gaussian curve, $\text{abs}(gxx)$ represents absolute value of the second order derivative of the Gaussian curve.

the second order derivatives of the intensity, is defined as

$$H(\mathbf{x}, \sigma_D) = \begin{bmatrix} I_{xx}(\mathbf{x}, \sigma_D) & I_{xy}(\mathbf{x}, \sigma_D) \\ I_{xy}(\mathbf{x}, \sigma_D) & I_{yy}(\mathbf{x}, \sigma_D) \end{bmatrix} \quad (2.34)$$

where $I(\mathbf{x})$ represents the intensity at location $\mathbf{x}$, and $I_{mn}(\mathbf{x}, \sigma_D)$, $m, n \in \{x, y\}$, represents the second-order derivative of the image smoothed by a Gaussian filter with standard deviation σ_D .

The trace of the Hessian matrix is used to generate the blob map $K_{\text{Beaudet}}(\mathbf{x}) = \text{trace}(H)$. Utilizing the second order derivative of the intensity causes the Hessian detector sensitive to noise. Moreover, the trace of the Hessian matrix cannot effectively suppress the response of the edge.

2.2.2.2 Hessian Laplace/Affine Detector

Hessian Laplace detector was proposed by Mikolajczyk et al. [44] to detect the blob structures in multi-scales. The blob map is generated by

$$\text{cornerness}(\mathbf{x}) = \det(H(\mathbf{x}, \sigma_D)) - \kappa \text{trace}^2(H(\mathbf{x}, \sigma_D)), \quad (2.35)$$

where σ_D is the differential scale. By adjusting σ_D , blob detection in multi-scales is achieved. Similar to that of the Harris Laplace detector, the characteristic scale is determined with the help of the absolute value of the scale-normalized Laplacian operator, i.e.

$$|LoG(\mathbf{x}, \sigma_n)| = \sigma_n^2 |\Delta L(\mathbf{x}, \sigma_n)| = \sigma_n^2 |L_{xx}(\mathbf{x}, \sigma_n) + L_{yy}(\mathbf{x}, \sigma_n)|, \quad (2.36)$$

where σ_n is proportional to σ_D . The Hessian detector is stable in estimating the characteristic scales of blobs, of which the shapes are similar to that of the Laplacian operator. However, it is sensitive to noise because it employs the second order derivative. The blob-like interest points tend to have less accuracy in location [1].

Hessian-affine detector was proposed to enhance the performance of the Hessian-Laplace detector in dealing with the affine variations. In order to obtain the affine invariant interest points, this detector iteratively normalizes the affine regions until the absolute value of the ratio of the two eigenvalues of the Hessian matrix converges to 1. The procedures are the same as that of Harris-affine detector described in Section 2.2.1.3 [1, 44].

2.2.2.3 SIFT Detector

The SIFT detector was proposed by Lowe [38, 48]. This detector is the most famous one for its high computational efficiency and high location accuracy. Based on the diffusion equation in scale-space theory $\sigma \nabla^2 G = \partial G / \partial \sigma$ [74], Laplacian in the spatial dimensions is proportional to the derivative in the scale dimension. This shows that it is reasonable to approximate the

Figure 2.7: Overview of the SIFT detector scheme [1].

Laplacian of Gaussian (LoG) by the different of Gaussian (DoG) in the spatial dimension as

$$\sigma \nabla^2 G = \partial G / \partial \sigma \approx \frac{G(\mathbf{x}, k\sigma) - G(\mathbf{x}, \sigma)}{k\sigma - \sigma}, \quad (2.37)$$

which can be reformulated as

$$G(\mathbf{x}, k\sigma) - G(\mathbf{x}, \sigma) \approx (k - 1)\sigma^2 \nabla^2 G. \quad (2.38)$$

Therefore, the SIFT detector employs the DoG, which does not need the second order derivatives, to generate the blob map.

The SIFT detector mainly contains two steps: scale-space extremum detection and interest point localization. In the first step, as shown in Fig. 2.7, images in multi-scales are generated by continuously smoothing it with a Gaussian kernel. The multi-scale images are grouped into several octaves to improve the computational efficiency. Difference of Gaussian is produced by subtracting the adjacent Gaussian images. Local extremum of the DoG images in the spatial and scale dimensions are detected as the interest points candidates. In the second step, a detailed fit is performed for each interest point candidate to the nearby data in both the spatial and scale dimensions. The discrete DoG image is interpolated based on the Taylor expansion

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}} \mathbf{x} + \frac{1}{2} \mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x}. \quad (2.39)$$

The refined location $\hat{\mathbf{x}}$ of the interest point is the nearest extremum of the interpolated DoG image. The detailed fit is determined by taking the derivative of interpolation function, and setting it to zero, giving

$$\hat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2} \frac{\partial D}{\partial \mathbf{x}}. \quad (2.40)$$

The contrast of this extremum is estimated by

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2} \frac{\partial D^T}{\partial \mathbf{x}} \hat{\mathbf{x}}. \quad (2.41)$$

Low contrast points are considered as noise and rejected by a threshold. The principal curvatures at each interest point are computed from the Hessian matrix, defined as

$$H = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix}. \quad (2.42)$$

The ratio of the principal curvatures is used to remove the points on the edge. In order to avoid computing the two eigenvalues, this ratio is measured by $\text{trace}^2(H)/\det(H)$.

The SIFT detector can localize the interest point well in the spatial and scale dimensions. However, it still have some drawbacks. It is sensitive to noise. Has high response on the edges and slight vibrations may cause false detection. Moreover, this detector can't effectively suppress the effect of small but strong abrupt structures on the detection of large scale structures.

2.2.2.4 SURF: Speeded-Up Robust Features Detector

The SURF detector [2, 49] (also named fast-Hessian detector) was proposed to speed up the Hessian-Laplace detector. It uses the determinant of the Hessian matrix to detect the interest points in both spatial and scale dimensions. The second order derivatives in the Hessian matrix are achieved by convolving the image with the second order derivatives of the Gaussian filter. In order to reduce the computational complexity, the second order derivatives of the Gaussian filter are approximated by a set of box filters, as shown in Fig. 2.8. The convolution of the image

Figure 2.8: From left to right: the (discretised and cropped) second order Gaussian derivative $\frac{\partial^2}{\partial y^2}G(\sigma)$ and $\frac{\partial^2}{\partial xy}G(\sigma)$, and the box filters approximating for previous Gaussian derivatives respectively. The grey regions are equal to zero [2].

with the box filters can be executed very fast by using the integral image, which is defined as

$$I_{\Sigma}(\mathbf{x}) = \sum_{i=0}^{x} \sum_{j=0}^{y} I(i, j). \quad (2.43)$$

The box filters can produce a sufficient approximation of the Gaussian derivatives as the generated noise in the approximating process is not obvious compared with other noise. A weighting coefficient is employed to alleviate the influence of the box-filter approximation in evaluating the determinant of the Hessian matrix

$$\det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2. \quad (2.44)$$

Local extremum is selected as the interest point candidate by the non-maximum suppression in a $3 \times 3 \times 3$ neighborhood. This method has been reported to be more than five times faster than the SIFT detector [49].

2.2.2.5 Salient Region Detector

The salient region detector was proposed by Kadir et al. [50, 73]. Instead of using the intensity derivatives, which magnify the influence of the noise, this detector utilizes the entropy to measure the complexity of the local structures. High complexity of the local structure leads to high entropy. Salient regions with the local maximum entropy are selected as the interest point can-

didates. In order to select the characteristic scale, the self-dissimilarity in the scale dimension is used as a weighting function. This detector contains the following stages:

1. The probability distribution function of intensity values within a local image region $p(I, \mathbf{x}, s)$ is computed over the parameter of scale s and location $\mathbf{x}$;
2. Entropy of each pixel at scale s is computed by $\mathcal{H}(\mathbf{x}, s) = -\sum_I p(I, \mathbf{x}, s) \log p(I, \mathbf{x}, s)$;
3. Local maxima of entropy at each scale are selected as the interest point candidates. The derivative of the entropy against scale at each candidate is computed by $\mathcal{W}(\mathbf{x}, s) = s^2/(2s-1) \sum_I |\partial p(I, \mathbf{x}, s)/\partial s|$;
4. Weight $\mathcal{H}(\mathbf{x}, s)$ with $\mathcal{W}(\mathbf{x})$ to create the weighted entropy $\mathcal{V}(\mathbf{x}, s) = \mathcal{W}(\mathbf{x}, s)\mathcal{H}(\mathbf{x}, s)$. The weighted entropy values are sorted and one simple cluster algorithm is used to find the salient region.

The number of interest points detected by this method is relatively low. The rank of the extracted points is meaningful due to the entropy based criteria: higher entropy means more stable. The affine invariant version of the detector was proposed in [73]. Local maximum over the scale s and the shape parameters (orientation θ and ratio λ of the major to minor axis of the elliptical region) is considered as the interest points candidate. However, this process significantly slows down this detector [1].

2.2.2.6 Self-similarity Detector

Self-similarity detector was designed by Maver [3] recently. This detector proposed a uniform computational concept to detect different image structures, such as: corner-like structures, blob-like structures and highly texture structures. This detector is motivated by the fact that local image patches with self-similar structures are distinguishable from their vicinity, and hence can be used for image matching.

The self-similarity of an image patch P is first modeled by a linear relationship $I(T(\mathbf{x})) = a + bI(\mathbf{x}), \forall \mathbf{x} \in P$, where T is limited to the reflection and rotation functions. For an image patch, if one of the two solutions $a = 0, b = 1$ and $a = I(x) + I(T(x)), b = -1$ is suitable for the above model, the image patch is considered as self-similar. In order to extend the linear relationship to real image data, the self-similarity of the image patch P is measured by the normalized correlation coefficient:

$$ncc(P, T) = \frac{\sum_i (I(\mathbf{x}_i) - \bar{I})(I(T(\mathbf{x}_i)) - \bar{I})}{\sum_i (I(\mathbf{x}_i) - \bar{I})^2}. \quad (2.45)$$

The value $ncc = 1$ for symmetric patch, while $ncc = -1$ for antisymmetric patch.

In order to avoid computing the ncc for every translating function T , the average ncc is used to measure the local saliency. If T is a reflection or rotation function, the local saliency, named as radial saliency, can be simplified as

$$S_r(P) = \frac{1}{N} \frac{1}{V_P} \sum_{m=0}^{M-1} (C_m - \bar{C})^2, \quad (2.46)$$

where M and N are constants and related to the sampling intervals for the radius and angle, respectively, V_P is the total variance of the local image patch, C_m is the variance on the circumference with radius r_m and $\bar{C} = \frac{1}{M} \sum_{m=0}^{M-1} C_m$. This measurement is used to detect blobs. Similarly, other two measurements, tangential saliency and residual saliency, are also defined under the uniform self-similarity measurement. The measurement of tangential saliency is

$$S_t(P) = \frac{1}{M} \frac{1}{V_P} \sum_{n=0}^{N-1} (R_n - \bar{R})^2, \quad (2.47)$$

where R_n demotes the sum of pixels on the n^{th} radius and $\bar{R} = \frac{1}{N} \sum_{n=0}^{N-1} R_n$. The residual saliency is defined as

$$S_{res} = 1 - S_r - S_t. \quad (2.48)$$

The tangential saliency and residual saliency are used to detect the corner and the textured patterns, respectively. Fig 2.9 shows three sets of image patterns for the proposed three mea-

Figure 2.9: Image patterns and their polar representations: column (a) have high radial saliency; column (b) have high tangential saliency; and column (c) have high residual saliency [3].

surements.

This detector uses the local variance to normalize the saliency. If the image patch is small, the local variance is small and its reciprocal is unstable. This makes the interest points sensitive to noise. The edge has high tangential saliency. This makes many points be detected on the edge if the tangential saliency is used to detect the interest points.

2.2.2.7 Discussion

Blob detectors have been widely used in many applications. Compared with the corner-like structures, it is easy to define the scale and shape of the blob-like structures, but difficult to localize them accurately in the image plane. This makes the blobs less suitable for some applications that need high location accuracy, e.g., camera calibration and 3D reconstruction. On the other hand, the blob detectors have been widely used in object recognitions, in which precise location is not needed since the entire recognition process is very noisy. The well-defined scales of the blob structures facilitate the extraction of the descriptors, such as SIFT descriptor, and thus useful in recognition application.

The Hessian, Hessian-Laplace/Affine, SIFT and SURF detectors are based on the second

order derivative of intensity. Using the second order derivative makes these detectors sensitive to noise. Hessian-Laplace detector is more stable in scale selection compared to Harris-Laplace detector. The SIFT detector is a speed-up version of Hessian detector. It employs the different of Gaussian to approximate the Laplace of Gaussian. A detailed fit is performed on each interest point candidate to refine its location and reject the unstable interest points. However, as linear filters cannot efficiently suppress impulsive noise, The SIFT detector is sensitive to impulsive noise and abrupt variations. The SURF detector employs the box filters to approximate the second order derivatives of Gaussian filters. This detector is reported to be five times faster than the SIFT detector. The salient region detector employs the local entropy to detect blobs in multi-scales. Comparing with other detectors, the number of the interest points detected by the salient region detector is smaller because it employs a greedy cluster method to group the nearby interest points. The self-similarity detector employs the local variance to detect corners, blobs and textured patterns. It uses the reciprocal of the local variance to normalize the saliency. If the image is flat or the scale is small, the local variance is small and its reciprocal is unstable. In this case, this detector is sensitive to noise.

2.2.3 Region Detectors

The following region detectors are reviewed: (1) the maximally stable extremal region (MSER) detector [53, 75], (2) the intensity-based region (IBR) detector [52, 76], and (3) the principal curvature-based region (PCBR) detector [6].

2.2.3.1 Maximally Stable Extremal Region Detector

Matas et al. [53, 75] proposed the maximally stable extremal region (MSER) detector to alleviate the affine influence in the interest point detection. Extremal regions refer to the regions in which the intensity of pixels are either lower (dark extremal regions) or higher (bright extremal regions) than the pixels on their outer boundary. The set of extremal regions are obtained by

Figure 2.10: Example of the extremum region in the 1D curve: extremum region R_l is obtained with threshold l and $R_{l+\Delta}$ is obtained with threshold $l + \Delta$ [4].

binarizing. Extremal regions along scales are generated by increasing the binarization threshold. The maximally stable extremal region, whose sizes are stable over a large range of the binarizing threshold, is selected to describe the local structures. Fig. 2.10 gives a 1D example of the extremum regions. The area of the extremum region is a function of the binarization threshold. The extremum regions have the following advantages: (1) it is closed under monotonic change of the intensity; (2) it is closed under continuous transformation of image coordinates. The procedures of this detector are as follows:

1. Segment the image into a series of binary images by a series of thresholds with ascending order;
2. The closed sets in the binary image are detected out and named extremal regions;
3. Extremal regions which are stable over a large range of threshold is selected as the MSER.

The detected regions of the MSER detector are irregular. In order to facilitate the extraction of the descriptor, an ellipse region, which has the same first and second moment matrices [31] as that of the irregular region, is used instead. This detector works well for structured images which can be segmented well, especially for those images with strong intensity changes on the

Figure 2.11: Flowchart of the intensity-based region detector [5].

edges. Its performance decreases if the edge is blurred.

2.2.3.2 Intensity-based Region Detector

The intensity-based region (IBR) detector was proposed by Tuytelaars et al. [52, 76]. Similar to the MSER detector, this detector starts from the local extreme of the image intensity. The boundary of the invariant region is decided based on the intensity changes along rays emanating from the local extreme. The irregular shape is then replaced by the ellipse region with the same first and second moment matrices to facilitate the extraction of the descriptor. The procedures are as follows:

1. Search the local extreme of the image, which are considered as anchor points;
2. At each anchor point, the intensity change function along rays emanating from this point, as shown in Fig. 2.11(a), is calculated as

$$f(t) = \frac{\text{abs}(I(t) - I_0)}{\max\left(\frac{\int_0^t \text{abs}(I(t) - I_0) dt}{t}, d\right)}, \quad (2.49)$$

where t is the Euclidean arc-length along the ray, $I(t)$ and I_0 are the intensities at point t and the anchor point respectively, d is a small value to avoid a division by zero;

3. The maxima along each ray, as shown in Fig. 2.11(b), are selected and linked one by one to generate the affine covariant region, as shown in Fig. 2.11(c);

Figure 2.12: Flowchart of the PCBR detector: (a) input image; (b) principal curvature image; (c) filtered binary image; (d) watershed regions; (e) detected interest regions fitted by ellipses [6].

4. An ellipse is used to replace the above region, as shown in Fig. 2.11(d). The ellipse region has the same first and second moment matrices as the original region;
5. Enlarge the ellipse region with a ratio 2 to make it more distinctive, as shown in Fig. 2.11(e).

2.2.3.3 Principal Curvature-Based Region Detector

The principal curvature-based region (PCBR) detector was proposed by Deng et al. [6] for object class recognition. The PCBR detector first segments the image based on its principal curvature images in multiple scales. The maximally stable regions in consecutive scales are extracted to describe the image. One example is shown in Fig. 2.12. The procedures of this method are as follows:

1. Create the set of smoothed images by continuously convolving with a Gaussian kernel as done in the SIFT detector;
2. Generate the principal curvature images using the minimum (or maximum) eigenvalues of the Hessian matrix;
3. Generate the maximum principal curvature images by selecting the maximum over each triplet consecutive principal curvature images;

4. Denoise using the grayscale morphological closing operation and binarize the image with the eigenvector-flow hysteresis thresholding method;
5. Perform the watershed transform on the above binarized image. The interest regions are generated by fitting the resulting segmented regions with ellipses, which have the same first and second moment matrices.

2.2.3.4 Discussion

The local regions extracted by the region detectors are often homogeneous regions. This may cause the descriptors drawn from these regions have less distinctiveness. In order to alleviate this problem, some detectors [52] enlarge the extracted support regions to include more complex image structures. The IBR detector is stable to small gap but will break down if the local image structure doesn't have local extreme. The MSER detector has high repeatability of image structures which have strong edges. However, it is sensitive to image blur, which makes it difficult to detect the maximum stable regions. Similarly, the PCBR detector could work well if the local region has strong boundary but cannot effectively extract the structures of which the boundary is not clear.

2.2.4 Evaluation Criterion

Evaluation is an important task in designing the interest point detectors. Some criteria were given to evaluate the performance [31, 32, 77]. In the following, the related criteria are reviewed briefly.

Ground truth verification Ground truth is firstly generated by human experts. Then the interest points are extracted by the detectors from the same image. Finally the comparison between the ground truth and the extracted interest points is used to evaluate the detector's performance. This evaluation method highly depends on the experts, and it is difficult to

perform on a large database.

Visual inspection The performance of the detectors is evaluated based on a set of visual criterion. These evaluation methods are more subjective due to they directly depend on human to evaluate the results. It is very time consuming to use these evaluation methods on a large database.

Consistency Consistency of corner number (CCN) [77] is used to evaluate the stability of the number of corners detected under various variations, and it is defined as

$$\text{CCN} = 100 \times 1.1^{-|n_t - n_o|}, \quad (2.50)$$

where n_o is the number of corners in the original image, and n_t is the number of corners in the transformed image. n_o and n_t should be detected in the same scene. The CCN measurement only considers the number of detected corners and does not care whether they are correct or not.

Repeatability The repeatability [32] is defined as: $r = m / (\min(n_o, n_t))$, where m , n_o and n_t are the number of repeated interest points, the number of interest points in the original image, and the number of interest points in the transformed image, respectively.

Two regions x_o and x_t is deemed to repeat if the overlap error ϵ is sufficiently small, such as $\epsilon < \epsilon_o$. The overlap error ϵ , defined in [31], is as follow:

$$\epsilon = 1 - \frac{R_{\mu_a} \cap R_{(H^T \mu_b H)}}{R_{\mu_a} \cup R_{(H^T \mu_b H)}}, \quad (2.51)$$

where μ is the second moment matrix of the corresponding support region, R_μ represents the elliptical region defined by $\mathbf{x}^T \mu \mathbf{x} = 1$. H is the homography of the two images. $R_{\mu_a} \cup R_{(H^T \mu_b H)}$ and $R_{\mu_a} \cap R_{(H^T \mu_b H)}$ are the union and intersection of the regions, respectively.

Matching score The matching score is used to measure the discriminative ability. The rules to compute the matching score given in [31] are as follows:

1. The overlap error ϵ of a matching region pair should be smaller than a threshold, such as $\epsilon \leq 40\%$. This provides the ground truth for correct matches.
2. The matching score is defined as the ratio between the number of correct matches and the smaller number of the detected regions in the same scene of the image pair. A match is the nearest neighbor in the descriptor space, which is measured by the Euclidean distance.

Among the above criteria, the repeatability and matching score are widely used because they can be easily carried out on a large database, and can work under large affine variations [31].

2.3 Summary

In this chapter we give a review of the leading approaches in noise suppression and interest point detection. Linear filters are widely used in these two research areas because of their easy implementation, low computational complexity and the soundly mathematical analysis tools. The linear filters are optimal in suppressing the short-tailed noise. However, their performance decreases in dealing with the long-tailed noise and abrupt structures which are frequently encountered in image processing. They also blur image structures and destruct image details. Although the soft-limiting filters can effectively preserve image details, their performance becomes poorer in case the long-tailed noise or impulsive noise exists. In contrast to the linear filters, the median based nonlinear filters can effectively attenuate the long-tailed noise and impulsive noise. By making a compromise between the mean and median filters, various filters are designed to effectively suppress both the short- and long-tailed noise. However, optimizing this compromise for different images is still not an easy task. Moreover, most of these filters are designed based on data sorting. Compared to the arithmetic computing, the data sorting is time consuming and this becomes a barrier for it being widely used in image processing.

Filters which own merits of both the mean and median filters while do not require data sorting are desirable. The ITM filter proposed a novel way to approximate the median with arithmetic computing. However, its computational complexity is of order $\mathcal{O}(n\sqrt{n})$ as it uses an iterative algorithm. This makes it slower than the median filter especially for larger filter size. Compared to trimming the extreme samples, truncating the extreme samples employed by the ITM filter still cannot effectively suppress the impulsive noise. Moreover, the flexibility of the ITM filter is limited as it can only play as a low-pass filter. In cases that requiring band- and high-pass characteristics, the ITM filter is not workable.

The merits of the linear filters also induce the development of interest point detectors based on such filters. In this chapter, a detailed review of the detectors, including corner detectors, blob detectors and region detectors, is given. Among these detectors, the linear filter based detectors, especially the SIFT detector, is more prevalent according to their citations from Google scholar. However, the deficiencies of the linear filters are also inherited by the corresponding detectors. As the linear filters are sensitive to impulsive noise and cannot effectively handle the image structures, the linear filter based detectors are not robust to the impulsive noise and abrupt variations. Most of the linear filter based detectors use the first or second order derivatives to enhance the response of the local features. As the derivative also has high response on edges and boundaries, unstable points are detected along the image structures. Compared to the linear filter, the median based filter can easily handle the impulsive noise and abrupt variation. However, there are still many issues in designing the interest point detectors with the median filter, which we will solve in the corresponding chapters.

Chapter 3

Further Properties and a Fast Realization of the ITM Filter

In this chapter¹, we analyze the ITM filter [7] in noise suppression based on the Cramer-Rao lower bound (CRLB). It is verified that the ITM filter outperforms the median filter in attenuating not only the short-tailed Gaussian noise but also the long-tailed Laplacian noise. The truncation algorithm of the ITM filter is analyzed and its regularity is unveiled. A fast realization of the ITM filter is proposed based on this analysis. Mathematical studies on the computational complexity of both the original and the fast ITM (FITM) filter are given. It shows that the proposed FITM filter is of order $\mathcal{O}(n \log(n))$. It is faster than the original ITM filter whose order is $\mathcal{O}(n\sqrt{n})$. Experimental results corroborate the reduction of the computational complexity, and it shows that the proposed algorithm is faster than the standard median filter.

3.1 Introduction

The iterative truncated arithmetic mean (ITM) filter [7] was proposed recently. It iteratively truncates the extreme values of samples in the filter window to a dynamic threshold. This threshold guarantees that the filter output converges to the median of the input samples. Proper

---

¹This work is partially published in IEEE Transactions on Circuits and System-II (journal paper [1] in author's publication list).

stop criterion enables the ITM filter owning merits of both the arithmetic mean and the order-statistical median operations. Both edge preservation and noise attenuation can be achieved within just a few iterations. The ITM filter outperforms the median filter in attenuating the single type of noise, such as Gaussian and Laplacian noise, and the mixed type of noise, such as the mixed Gaussian and impulsive noise. It also offers a way to estimate the median by a simple arithmetic computing algorithm.

Using the mean absolute error as a criterion, it is shown in [7] that the ITM filter outperforms the median filter in attenuating Laplacian noise. This opens to doubt because the median is the optimum location estimator of Laplacian noise in the sense of maximum likelihood estimation (MLE). Therefore, in this chapter, we use the Cramer-Rao lower bound (CRLB) [58] and the mean square error (MSE) to put it beyond doubt that the ITM filter outperforms the median filter in attenuating Laplacian noise. As the ITM filter employs an iterative algorithm in the filtering process, people tend to think its computational complexity must be much higher than the median filter. This chapter demonstrates that it is not the case. We show that, compared to the standard median filter, the ITM filter is faster for small filter size but slower for large filter size. Therefore, we propose a fast ITM filter (FITM filter) which is faster than the standard median filter for all filter sizes.

3.2 Noise Suppression Properties of the ITM Filter

Given a set of n samples $\mathbf{x}_0 = \{x_i\}$ in the filter window, the ITM algorithm [7], starting from $\mathbf{x} = \mathbf{x}_0$, iteratively truncates the extreme samples in $\mathbf{x}$ to a dynamic threshold τ as shown in Algorithm 1.

The stopping criterion S proposed in [7] is composed of four termination rules. In general, it terminates the iteration if the truncated mean is close to the median or has little change. It utilizes some relationships of the numbers of samples in different conditions to terminate the

Algorithm 1: Truncation Procedure of the ITM Filter**Input:** $\mathbf{x}_0 \Rightarrow \mathbf{x}$; **Output:** Truncated $\mathbf{x}$;1 **do**2   Compute the sample mean: $\mu = \text{mean}(\mathbf{x})$;3   Compute the dynamic threshold: $\tau = \text{mean}(|\mathbf{x} - \mu|)$;4 $b_l = \mu - \tau$, $b_u = \mu + \tau$, and truncate $\mathbf{x}$ by:

$$x_i = \begin{cases} b_u, & \text{if } x_i > b_u \\ b_l, & \text{if } x_i < b_l ; \\ x_i, & \text{otherwise} \end{cases}$$

5 **while** the stopping criterion S is violated ;

iteration.

The ITM filter has two types of outputs. In this chapter, we focus on the type I output ITM1 that has a better performance in attenuating the short- and long-tailed noise than the other type of ITM output. The ITM1 output [7] is

$$\text{ITM1 : } y_{t1} = \text{mean}(\mathbf{x}),$$

where $\mathbf{x}$ is the truncated data set output from Algorithm 1. As a necessary preliminary of the study, two properties of the ITM filter are presented as follows.

Property 1: The distribution of the ITM output is symmetric, if the samples of the input data set $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are independent and identical distributed (i.i.d.) samples of the random variable X with a symmetric distribution around the symmetry center c .

Proof: If x_i is symmetric around c , $2c - x_i$ has the same distribution as x_i . Thus, the ITM output $y_{t1}(x_1, x_2, \dots, x_n)$ has the same distribution as $y_{t1}(2c - x_1, 2c - x_2, \dots, 2c - x_n)$, which equals $2c - y_{t1}(x_1, x_2, \dots, x_n)$ because the ITM filter is invariant to scale and shift [7]. It follows that the distribution of y_{t1} is symmetric around c . ■

Property 2: The output of the ITM filter is an unbiased estimate of the population mean of

X , if the samples in $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are i.i.d. samples of the random variable X with a symmetric distribution around the symmetry center c .

Proof: As x_i is symmetrically distributed around c , $E\{X\} = c$. Similarly, $E\{y_{t1}\} = c$ according to Property 1. Therefore, $E\{y_{t1}\} = E\{X\}$. This completes the proof of Property 2. ■

In this section, we analyze why the ITM1 filter (a) cannot be better than the mean filter for Gaussian noise, and (b) can be superior to the median filter for Laplacian noise even though the median is the optimum location estimator of Laplacian noise in the sense of MLE. As Property 2 shows that the output of the ITM filter is unbiased, the analysis is based on Cramer-Rao lower bound (CRLB) [58], which provides a lower bound of the mean square error (MSE) of unbiased estimators.

Table 3.1: CRLB and MSE of Sample Mean μ and Median ϕ .

	CRLB	MSE(μ)	MSE(ϕ)
Gaussian noise	σ^2/n	σ^2/n	$\frac{\pi}{2(n+2)}\sigma^2$
Laplacian noise	$\sigma^2/(2n)$	σ^2/n	$\sigma^2 \sum_{r=0}^m \frac{K(m,r)}{(m+r+1)^2}$

3.2.1 Gaussian Noise

The probability density function (pdf) of Gaussian noise X with the population mean μ_o and the standard deviation σ is

$$f_G(X = x|\mu_o, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x - \mu_o)^2}{2\sigma^2}\right). \quad (3.1)$$

As shown in Table. 3.1, the CRLB for the estimate of μ_o is σ^2/n . The MSE of the sample mean μ is σ^2/n [78], which equals to the CRLB. Thus, μ is the minimum MSE estimator. It means no estimator can outperform μ in estimating signals in Gaussian noise in terms of MSE. For large

Figure 3.1: Normalized MSE against (a) the number of iterations k , and (b) the filter size n for Gaussian noise.

n , the MSE of the sample median ϕ approximates $\text{MSE}(\phi) = \frac{\pi}{2(n+2)}\sigma^2$ [78]. It is about $\pi/2$ times of the CRLB.

It is difficult to get a mathematical expression of the MSE of the ITM1 output. Here we use the Monte Carlo simulations [78] to evaluate the ITM1 filter. 10^6 independent input data sets are used in the simulations. As the α T filter approaches the mean if $\alpha \rightarrow 0$ and approaches the median if $\alpha \rightarrow 0.5$, $\alpha = 0.25$ is chosen in this chapter. Fig. 3.1(a) shows the normalized MSE of the ITM1 output against the number of iterations. The filter size is $n = 49$. When the number of iterations is zero, the ITM1 output equals the mean. Its MSE approximately equals the CRLB. By increasing the number of iterations, the ITM1 output approaches the median, and its MSE increases and approaches that of the median. Fig. 3.1(b) shows the normalized MSE against the filter size. The ITM1 filter employs the default stopping criterion in [7]. We see that both the ITM1 and α T filters significantly outperform the median filter.

3.2.2 Laplacian Noise

The pdf of Laplacian noise X is

$$f_L(X = x|\mu_o, b) = \frac{1}{2b} \exp\left(-\frac{|x - \mu_o|}{b}\right), \quad (3.2)$$

where μ_o is the population mean of X and b is a scale parameter. Its variance is $\sigma^2 = 2b^2$.

Table. 3.1 shows its CRLB is $\sigma^2/(2n)$. The derivation of the CRLB is as follows.

The first order partial derivative of $\ln f_L(x_i|\mu_o, b)$ against μ_o is

$$\begin{aligned} \frac{\partial \ln f_L(x_i|\mu_o, b)}{\partial \mu_o} &= -\frac{1}{b} \frac{\partial ((x_i - \mu_o)^2)^{1/2}}{\partial \mu_o} \\ &= \frac{1}{b} ((x_i - \mu_o)^2)^{-1/2} (x_i - \mu_o) \\ &= \frac{1}{b} (2U(x_i - \mu_o) - 1), \end{aligned} \quad (3.3)$$

where $U(x_i)$ is the step function defined as

$$U(x_i) = \begin{cases} 1, & \text{if } x_i \geq 0 \\ 0, & \text{if } x_i < 0 \end{cases}. \quad (3.4)$$

It is easy to get that $E\left\{\frac{\partial \ln f(x_i|\mu_o, b)}{\partial \mu_o}\right\} = 0$. Then, due to $x_i \in \mathbf{x}$ is i.i.d,

$$\begin{aligned} E\left\{\frac{\partial \ln f_L(\mathbf{x}|\mu_o, b)}{\partial \mu_o}\right\} &= E\left\{\frac{\partial \ln \prod_{i=1}^n f(x_i|\mu_o, b)}{\partial \mu_o}\right\} \\ &= \sum_{i=1}^n E\left\{\frac{\partial \ln f(x_i|\mu_o, b)}{\partial \mu_o}\right\} \\ &= 0. \end{aligned} \quad (3.5)$$

This verifies that the Laplacian distribution satisfies the regularity condition. The Fisher infor-

mation is

$$\begin{aligned} I(\mu_o) &= -E\left\{\frac{\partial^2 \ln f_L(\mathbf{x}|\mu_o, b)}{\partial \mu_o^2}\right\} \\ &= \frac{n}{b^2}. \end{aligned} \quad (3.6)$$

The CRLB is the reciprocal of the Fisher information [58], i.e.

$$\text{CRLB}(\mu_o) = b^2/n = \sigma^2/(2n). \quad (3.7)$$

As shown in Table. 3.1, the MSE of the sample mean μ is two times of the CRLB [79]. Therefore, μ is inefficient in estimating signals in Laplacian noise.

The distribution function of the median output is given by [80]:

$$g(\phi) = \sum_{r=0}^m K(m, r) \frac{1}{2\tilde{b}_r} \exp\left(-\frac{|\phi - \mu|}{\tilde{b}_r}\right), \quad (3.8)$$

where $\tilde{b}_r = b/(m + r + 1)$, $m = (n - 1)/2$ and

$$K(m, r) = (-1)^r \frac{(2m + 1)!}{m!r!(m - r)!(m + r + 1)2^{m+r}}. \quad (3.9)$$

Thus, the variance of the median output is

$$\text{var}(\phi) = 2b^2 \sum_{r=0}^m \frac{K(m, r)}{(m + r + 1)^2}. \quad (3.10)$$

When the filter size n is small, the MSE of ϕ is far away from the CRLB. For example, when $n = 9$, $\text{var}(\phi) \approx 0.175b^2$. It is about 1.58 times of the CRLB. From this we can conclude that ϕ is not the minimum MSE estimator for small filter size though it is the MLE [9]. Therefore, it is not a surprise that the ITM1 filter can outperform the median filter even for the long-tailed Laplacian noise.

Figure 3.2: Normalized MSE against (a) the number of iterations k , and (b) the filter size n for Laplacian noise.

The performance of the ITM1 filter is analyzed based on the Monte Carlo simulations with 10^6 independent input data sets. Fig. 3.2(a) shows the MSE of the ITM1 output against the number of iterations. The filter size is $n = 49$. When the number of iterations is zero, the ITM1 output equals the mean. Its MSE is 2 times of the CRLB. After a few iterations, the MSE of the ITM1 output becomes smaller than that of the median. The MSE of the ITM1 output approaches that of the median when the number of iterations is large enough. The normalized MSE against the filter size is shown in Fig. 3.2(b). The αT filter is superior to the median filter when $n \leq 25$ but inferior when $n > 25$. The ITM1 filter uses the stopping criterion in [7]. This stopping criterion is a general criterion that is applied in all experiments in [7]. Although this stopping criterion is not optimized for Laplacian noise, Fig. 3.2(b) shows that the ITM1 filter still outperforms the median filter.

3.3 The Proposed Fast ITM Filter

From the ITM algorithm (Algorithm 1), we see that all samples are visited in all iterations. This becomes a heavy burden when the filter size and the number of iterations are large. In order to reduce the computational burden, we propose a fast ITM (FITM) filter by only visiting the un-truncated samples in each iteration. This is enabled by the following proposition.

Proposition 1: Samples, once being truncated in an iteration of the ITM algorithm, must be truncated in all subsequent iterations.

Proof: Let $\mathbf{x}_h = \{x_i | x_i > \mu\}$, n_h be the number of the samples in $\mathbf{x}_h$, and $\delta_h = \text{sum}(\mathbf{x}_h - \mu)/n_h$ in the k^{th} iteration. Let x_{i+} , μ_+ , τ_+ , n_{h+} and δ_{h+} are the corresponding notations in the $(k+1)^{\text{th}}$ iteration.

Assume a sample x_{iu} is truncated to the upper bound $\mu + \tau$ in the k^{th} iteration. Obviously, $x_{iu+} = \mu + \tau$, which has the maximum value in $\mathbf{x}$ in the $(k+1)^{\text{th}}$ iteration. As τ decreases monotonically [7], $\tau_+ < \tau$, we have

$$x_{iu+} > \mu_+ + \tau_+, \text{ if } \mu_+ \leq \mu. \quad (3.11)$$

In case of $\mu_+ > \mu$, from $\tau = 2n_h\delta_h/n$ [7], we have

$$\tau = \frac{2}{n} \sum_{x_i > \mu} (x_i - \mu) \geq \frac{2}{n} \sum_{x_i > \mu_+} (x_i - \mu_+ + \mu_+ - \mu). \quad (3.12)$$

As at least one sample x_{iu} is truncated to the upper bound in the k^{th} iteration, we have

$$\frac{2}{n} \sum_{x_i > \mu_+} (x_i - \mu_+) > \frac{2}{n} \sum_{x_{i+} > \mu_+} (x_{i+} - \mu_+) = \tau_+. \quad (3.13)$$

Substituting (3.13) into (3.12) yields

$$\tau > \tau_+ + \frac{2n_{h+}}{n}(\mu_+ - \mu). \quad (3.14)$$

As $\tau_+ = 2n_{h+}\delta_{h+}/n$ [7], (3.14) becomes

$$\tau > \tau_+ + \frac{\tau_+}{\delta_{h+}}(\mu_+ - \mu). \quad (3.15)$$

As $\delta_{h+} \leq x_{iu+} - \mu_+$, we have $\delta_{h+} \leq \tau_+$, if

$$x_{iu+} - \mu_+ \leq \tau_+. \quad (3.16)$$

Therefore, under the condition (3.16), (3.15) becomes

$$\tau > \tau_+ + \mu_+ - \mu. \quad (3.17)$$

Since $x_{iu+} = \mu + \tau$, (3.17) becomes

$$x_{iu+} > \mu_+ + \tau_+. \quad (3.18)$$

The conclusion (3.18) conflicts with (3.16). Hence, the condition (3.16) is not true, which means,

$$x_{iu+} > \mu_+ + \tau_+, \text{ if } \mu_+ > \mu. \quad (3.19)$$

From (3.11) and (3.19), we have

$$x_{iu+} > \mu_+ + \tau_+. \quad (3.20)$$

In the same way, we can prove that if a sample x_{il} is truncated to the lower bound $\mu - \tau$ in the k^{th} iteration,

$$x_{il+} < \mu_+ - \tau_+. \quad (3.21)$$

Inequalities (3.20) and (3.21) prove Proposition 1. ■

Proposition 1 shows that all truncated samples must be truncated in the subsequent iterations. In other words, all truncated samples have the same values of either the lower or upper bound in all subsequent iterations. Therefore, we do not need access such samples one by one. There is also no need to remember the positions of the truncated pixels. We only need count the number of such samples, and replace them by the constant $\mu - \tau$ or $\mu + \tau$ in all subsequent iterations. This leads to the FITM algorithm, which speeds up the truncation procedure by only visiting the un-truncated samples. Let $n_{\tau l}$ and $n_{\tau u}$ be the numbers of the samples smaller than the lower bound and larger than the upper bound, respectively. The proposed FITM algorithm is shown as follows.

Algorithm 2: Truncation Procedure of the FITM Filter

Input: $\mathbf{x}_0 \Rightarrow \mathbf{x}$, $n_{\tau l} = 0$, $n_{\tau u} = 0$; **Output:** $\mathbf{x}$, b_l , b_u , $n_{\tau l}$ and $n_{\tau u}$;

```

1 do
2    $\mu = (\text{sum}(\mathbf{x}) + n_{\tau l}b_l + n_{\tau u}b_u)/n$ ;
3    $\tau = (\text{sum}(|\mathbf{x} - \mu|) + n_{\tau l}(\mu - b_l) + n_{\tau u}(b_u - \mu))/n$ ;
4    $b_l = \mu - \tau$ ,  $b_u = \mu + \tau$ ,  $\mathbf{x} = \{x_i | b_l \leq x_i \leq b_u\}$ , and update  $n_{\tau l}$  and  $n_{\tau u}$ ;
5 while the stopping criterion  $S$  is violated ;

```

Comparing Algorithm 2 with Algorithm 1, we can find that both μ and τ computed in these two algorithms are the same. Therefore, the ITM and FITM filters have the same outputs. The difference is that, in Step 2-4, Algorithm 2 only visits the un-truncated samples while Algorithm 1 visits all the samples in each iteration. This modification, enabled by Proposition 1, speeds up the ITM algorithm.

3.4 Computational Complexity

The computational complexity of the ITM and FITM filters can be measured by the times that all the samples are visited in the iterations. The visiting times is determined by two factors: (a) the number of iterations N_s , and (b) the probability p_k of a sample being visited in the k^{th} iteration. The number of iterations N_s for both the ITM and FITM filters are the same because they utilize the same default stopping criterion in [7]. The probability p_k is different for these two filters. For the ITM filter, $p_k = 1$ because all the samples are visited in each iteration. For the FITM filter, only the un-truncated samples are visited. Therefore, p_k decreases monotonically against the number of iterations.

We use the Monte Carlo simulations [78] to analyze the number of iterations N_s . Three types of noise, Gaussian, Laplacian, and the uniform distributed noise, are employed. 10^6 independent input data sets are used in each experiment. The experimental results in Fig. 3.3 illustrate that the numbers of iterations, which are determined by the default stopping criterion, of different noise types are approximately the same. Fig. 3.3 shows that N_s is approximately a linear function of $\sqrt{n}$. Therefore, we use

$$\hat{N}_s = \sqrt{n} - 1 \quad (3.22)$$

as an upper bound of N_s , which is plotted in Fig. 3.3.

For the ITM filter, as the probability of a sample being visited in the k^{th} iteration is $p_k = 1$, the total visiting times of all the samples is $\sum_{k=1}^{\hat{N}_s} np_k = n(\sqrt{n} - 1)$. Its computational complexity is $\mathcal{O}(n(\sqrt{n} - 1)) = \mathcal{O}(n\sqrt{n})$.

The FITM filter only visits the un-truncated samples in each iteration. As the dynamic threshold τ_k decreases monotonically [7], the probability of a sample within the range $(\mu_{k-1} - \tau_{k-1}, \mu_{k-1} + \tau_{k-1})$ decreases. Therefore, the probability of a sample being visited p_k decreases.

Figure 3.3: Average number of iterations determined by the stopping criterion in [7] against the filter size n .

In order to simplify the analysis of the probability p_k , we employ the uniform distributed noise as an example. The pdf of a uniform distributed random variable X is

$$f_u(X = x) = \begin{cases} 1, & \text{if } -0.5 \leq x \leq 0.5 \\ 0, & \text{otherwise} \end{cases}. \quad (3.23)$$

From (3.23), we find the following lemma of τ_k .

Lemma 1: When the filter size n is sufficiently large, the dynamic threshold τ_k of X drawn from the uniform distribution (3.23) has a recurrence relation

$$\tau_k = \tau_{k-1}(1 - \tau_{k-1}), k > 1, \quad (3.24)$$

with $\tau_1 = 0.25$.

Proof: When the filter size n is sufficiently large, the sample mean equals the expectation

of X , as $\mu = E[X] = 0$. The dynamic threshold of the first iteration is

$$\tau_1 = \frac{1}{n} \sum_{i=1}^n |x_i - \mu| = E[|X|] = 0.25. \quad (3.25)$$

After the $(k-1)^{th}$ iteration, X can be either un-truncated or truncated. Only the samples within the range $(\mu_{k-1} - \tau_{k-1}, \mu_{k-1} + \tau_{k-1})$ are un-truncated. Thus, the probability of a sample un-truncated is $2\tau_{k-1}$, and the probability of truncation is $1 - 2\tau_{k-1}$. As the output of the FITM filter is unbiased, $\mu_{k-1} = 0$. The deviation of an un-truncated sample from the mean is $|x|$, and that of a truncated sample is τ_{k-1} . Therefore, the dynamic threshold of the k^{th} iteration τ_k is

$$\begin{aligned} \tau_k &= E[|X_{k-1}|] = (1 - 2\tau_{k-1})\tau_{k-1} + \int_{-\tau_{k-1}}^{\tau_{k-1}} |x| dx \\ &= \tau_{k-1}(1 - \tau_{k-1}), \end{aligned} \quad (3.26)$$

where X_{k-1} is the random variable X after the $(k-1)^{th}$ iteration. (3.25) and (3.26) complete the proof of (3.24). ■

Since τ_k has the property in Lemma 1, the summation of τ_k is constrained by the following lemma.

Lemma 2: The summation of the dynamic threshold τ_k specified by (3.24) is bounded by two logarithm curves,

$$0.5(\ln(0.5m + 1)) < \sum_{k=1}^m \tau_k < \ln(m + 1). \quad (3.27)$$

Proof: For $k = 1$, we have $\tau_k = 0.25 < 1/(k+1)$. For $k > 1$, we will prove $\tau_k < 1/(k+1)$ is true under the assumption of $\tau_{k-1} < 1/(k-1+1)$. Let $t = 1/\tau_{k-1}$, (3.24) yields $\tau_k = (t-1)/t^2$. Since $t > k$, we have

$$\tau_k = \frac{t-1}{t^2} < \frac{t-1}{t^2-1} = \frac{1}{t+1} < \frac{1}{k+1}. \quad (3.28)$$

Figure 3.4: Average visiting times of a sample against the filter size n . The x -axis is in log scale of n .

This proves that

$$\tau_k < 1/(k+1), k \geq 1. \quad (3.29)$$

Therefore,

$$\sum_{k=1}^m \tau_i < \sum_{k=1}^m \frac{1}{k+1} < \int_1^{m+1} \frac{1}{x} dx = \ln(m+1). \quad (3.30)$$

Similarly, for $k = 1$, we have $\tau_k = 0.25 \geq 1/(2(k+1))$. For $k > 1$, we will prove $\tau_k \geq 1/(2(k+1))$ is true under the assumption of $\tau_{k-1} \geq 1/(2(k-1+1))$. Since $k < t \leq 2k$, we have

$$\tau_k = \frac{t-1}{t^2} \geq \frac{t-1}{t^2+t-2} = \frac{1}{t+2} \geq \frac{1}{2(k+1)}. \quad (3.31)$$

This proves that

$$\tau_k \geq 1/(2(k+1)), k \geq 1. \quad (3.32)$$

Figure 3.5: Average visiting times of a sample against the filter size n . The x -axis is in linear scale of n .

Therefore,

$$\sum_{k=1}^m \tau_k \geq \sum_{k=1}^m \frac{1}{2(k+1)} > \int_2^{m+2} \frac{1}{2x} dx = 0.5 \ln(0.5m + 1). \quad (3.33)$$

Inequalities (3.30) and (3.33) complete the proof of (3.27). ■

As only the un-truncated samples are visited by the FITM algorithm, the probability of a sample being visited at the k^{th} iteration is $p_k = 2\tau_{k-1}$, where we define $\tau_0 = 0.5$. Therefore, its computational complexity is $\mathcal{O}(\sum_{k=1}^{\hat{N}_s} np_k) = \mathcal{O}(2n \sum_{k=1}^{\hat{N}_s} \tau_{k-1})$. From (3.27), we can get that $\mathcal{O}(\sum_{k=1}^{\hat{N}_s} \tau_{k-1}) = \mathcal{O}(\ln(\hat{N}_s))$. Thus, the computational complexity of the FITM filter is $\mathcal{O}(2n \ln(\hat{N}_s)) = \mathcal{O}(2n \ln(\sqrt{n} - 1)) = \mathcal{O}(n \ln(n))$. It is smaller than that of the ITM filter, and has the same order of the quick-sort algorithm.

The Monte Carlo simulations are also carried out to analyze the visiting times for the FITM filter when the filter size is not large enough. Experimental results in Fig. 3.4 illustrate that the average visiting times of a sample in the FITM filter is approximately a linear function of $\ln n$.

Figure 3.6: Normalized time consumption against the number of iterations k . The time consumption is normalized by that of the median filter.

Fig. 3.4 shows that

$$\hat{N}_{\text{FITM}} = 0.7 \ln n \quad (3.34)$$

is a close upper bound for the FITM filter for $9 \leq n \leq 81$. Therefore, the total visiting times of all the samples for the FITM filter is about $0.7n \ln n$. It is smaller than that of the quick-sort algorithm, which approximately equals $2n \ln n$ [81]. The average visiting times for both the ITM and FITM filters are compared in Fig. 3.5. It is seen that the visiting times for the FITM filter is smaller than that for the ITM filter.

We further evaluate the computational complexity of the ITM and FITM filters in two experiments. These experiments are performed under the Window 7 system with the Intel Core i5 CPU 3.2GHz. All of the filters are implemented by C programming language. As data sorting is the basic building block that many rank order statistic filters, such as the popular αT filter, are built around, we implement both the median and the αT filters using the quick-sort algorithm.

Figure 3.7: Normalized time consumption against the filter size n . The time consumption is normalized by that of the median filter.

The first experiment tests the running time of the filters against the number of iterations, and the second experiment tests that with the default stopping criterion given in [7] against the filter size. The time consumption is normalized by that of the median filter. The normalized time consumption against the number of iterations is shown in Fig. 3.6. The filter size is $n = 49$. The time consumption for the ITM filter is a linear function of the number of iterations because all the samples are visited in each iteration. As the FITM filter only visits the un-truncated samples, its time consumption increases slowly compared to that of the ITM filter. The ITM filter is faster than the median filter when the number of iterations $k \leq 5$ but slower for $k > 5$. The FITM filter is faster than the median filter for all the numbers of iterations in Fig. 3.6. The experimental results using the default stopping criterion are shown in Fig. 3.7. As the αT filter requires both arithmetic computing and data sorting operations, its time consumption is larger than that of the median filter. Compared to the median filter, the ITM filter is faster for the filter size $n \leq 49$ but slower for $n > 49$. The proposed FITM filter is faster than both the ITM and

median filters for all filter sizes.

3.5 Summary

In this chapter, some further properties of the ITM filter are analyzed. It shows that the ITM filter outperforms the median filter in dealing with both the short-tailed Gaussian noise and the long-tailed Laplacian noise. The computational complexity of the ITM filter is studied. It is $\mathcal{O}(n\sqrt{n})$. Experimental results show that the ITM filter is faster than the median filter when the filter size $n \leq 49$ but slower when $n > 49$. A fast implementation of the ITM filter is proposed. The computational complexity of the FITM filter is analyzed. The analysis reveals that the computational complexity of the FITM filter is $\mathcal{O}(n \ln n)$. Although it is of the same order as the median filter, experimental results demonstrate that the FITM filter is faster than the standard median filter implemented by the quick-sort algorithm.

Chapter 4

Weighted Iterative Truncated Mean Filter

In this chapter¹, a rich class of filters named weighted ITM (WITM) filters are proposed. By iteratively truncating the extreme samples, the output of the WITM filter converges to the weighted median. Proper stopping criterion makes the WITM filters own merits of both the weighted mean and median filters and hence outperforms the both in some applications. Three structures are designed to enable the WITM filters being low-, band- and high-pass filters. Properties of these filters are presented and analyzed. Experimental evaluations are carried out on both synthesis and real data to verify some properties of the WITM filters.

4.1 Introduction

The linear filters are widely used in digital signal/image processing because of their rigorous mathematical foundation and their efficiency in attenuating additive Gaussian noise. The sample mean is the optimal solution for suppressing additive Gaussian noise in the sense of mean square error (MSE) if all samples have the same variance, and so is the weighted mean if the variances are not identical. However, none of the mean and weighted mean filters is the optimal if the long-tailed noise, such as Laplacian noise, presents. Moreover, in some applications of image

---

¹This work is partially published in IEEE Transactions on Signal Processing (journal paper [2] in author's publication list).

processing, the linear filters are undesirable because they blur the image structures and cannot suppress the impulsive noise effectively. Therefore, nonlinear filters which can preserve signal structures and effectively suppress long-tailed noise were developed.

The median filter [82] is the most widely applied one among the nonlinear filters. It provides a powerful tool for signal/image processing because of its good property in impulsive noise suppression and edge preservation. Similar to the mean filter, the median filter is inefficient if the variances of different samples are not the same. The weighted median filters with positive weights were proposed to deal with the non-identical distributed Laplacian noise. Such filters are used in many applications, e.g., speech signal processing, images filtering [83] and waveform prediction [12]. However, they cannot achieve the acceptable results in some applications, such as equalization, beamforming and system identification, which require band- or high-pass characteristics. To overcome the above limitations, the general weighted median (GWM) filters admitting both positive and negative weights were proposed in [84]. The GWM filters can be designed as band-pass and high-pass filters. They are applied in various applications, such as sigma-delta modulation encoding [85], denoising [86–88], image sharpeners [89], edge detection [13], edge enhancement [90], system identification [91, 92] and multichannel signal processing [93]. The GWM filters have been extended to admit complex value of weights [11]. Designing the weights is a critical part and great effort was devoted in it [84, 94, 95].

Both the median and weighted median filters have some limitations. First, these filters are not as effective as the mean and weighted mean filters in suppressing the short-tailed Gaussian noise. Second, they are not the optimal ones even for the long-tailed Laplacian noise [7, 59]. Moreover, the median and weighted median filters are built on data sorting. It has high computational complexity compared to arithmetic computing and its implementation is also complicated [29].

Filters which can outperform the median filter in suppressing both the short- and long-tailed noise while do not require data sorting are desirable. The myriad filters [30, 61, 96–99] were

designed for the α -stable distributed noise model. Its performance highly depends on the tunable “linearity parameter” [9, 30] computed from the prior knowledge of the noise distribution. The iterative truncated arithmetic mean (ITM) filter [7] employs a simple truncating algorithm to iteratively truncate the extreme samples. Its output approaches the median by increasing the number of iterations. Proper stopping criterion enables the ITM filter outperforms the median filter in suppressing both Gaussian and Laplacian noise. Edge preservation and noise suppression can be achieved within just a few iterations. It also provides an approach to estimate the median by a simple arithmetic computing algorithm. Its implementation given in [59] is faster than the median and myriad filters.

The merits of the ITM filter inspire us to extend it into a rich class of filters. Although the ITM filter outperforms the median filter in suppressing the identical distributed Gaussian and Laplacian noise, analogous to the non-weighted mean and median filters, its performance drops dramatically in dealing with the non-identical distributed noise. Furthermore, the ITM filter cannot be used in applications which require band- or high-pass characteristics. In this chapter, we propose a rich class of filters named weighted ITM (WITM) filters, of which the ITM filter is a special case with all weights being equal. The truncating procedure of the ITM filter is extended to the WITM filters. By iteratively truncating the samples, the output of the WITM filter starts from the weighted mean and approaches the weighted median. A stopping criterion is proposed to terminate the iteration so that the WITM filters can outperform both the weighted mean and median filters in some applications. Three structures are utilized to enable the WITM filters with negative weights being low-, band- and high-pass filters. The superiority of the proposed WITM filters is verified in the experiments.

4.2 Weighted ITM Filters with Positive Weights

The weighted ITM filters are proposed following necessary reviews of the weighted mean and median filters and the ITM filter [7]. A stopping criterion is proposed and the filter properties are discussed.

4.2.1 Weighted Mean and Median Filters

The weighted mean is the maximum likelihood (ML) estimate of location for data sets with Gaussian distribution. Assume a filter window contains n independent Gaussian distributed samples as $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ with unknown constant mean μ_o . The variance of the i^{th} sample is σ_i^2 . The ML estimate of location μ_o is to find the value of μ , which maximizes the likelihood function

$$\begin{aligned} L(\mu|x_1, x_2, \dots, x_n) &= \prod_{i=1}^n f(x_i|\mu, \sigma_i) \\ &= \left(\prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma_i^2}} \right) \exp \left(- \sum_{i=1}^n \frac{(x_i - \mu)^2}{2\sigma_i^2} \right). \end{aligned} \quad (4.1)$$

It is equivalent to minimizing the squares sum

$$G_2(\mu) = \sum_{i=1}^n \frac{(x_i - \mu)^2}{\sigma_i^2}. \quad (4.2)$$

The value of μ minimizing (4.2) is the weighted mean μ_w

$$\mu_w = \arg \min_{\mu} G_2(\mu) = \frac{\sum_{i=1}^n w_i x_i}{\sum_{i=1}^n w_i}, \quad (4.3)$$

where $w_i = 1/\sigma_i^2$. μ_w is the optimal estimate of μ_o because its variance equals to the Cramer-Rao lower bound (CRLB) [58].

Similarly, the ML estimate of location μ_o under Laplacian distribution is equivalent to minimizing

$$G_1(\mu) = \sum_{i=1}^n \frac{|x_i - \mu|}{\sigma_i}. \quad (4.4)$$

The value of μ that minimizes (4.4) is the weighted median ϕ_w

$$\phi_w = \arg \min_{\mu} G_1(\mu) = \text{median}(w_1 \diamond x_1, w_2 \diamond x_2, \dots, w_n \diamond x_n), \quad (4.5)$$

where $w_i = 1/\sigma_i$ and $\diamond$ is the replication operator defined by

$$w_i \diamond x_i = \overbrace{x_i, x_i, \dots, x_i}^{w_i \text{ times}}. \quad (4.6)$$

In fact, the weighted median is searched in the following way to avoid expanding the data and cope with the non-integer weights [84]:

1. Calculate the threshold $T_0 = \frac{1}{2} \sum_{i=1}^n w_i$.
2. Sort the samples x_i .
3. Sum the magnitude of the weights of the sorted samples from the maximum continuing down in order.
4. The output is the sample whose magnitude weight causes the sum to become larger than or equal to T_0 .

The following example illustrates this procedure. Consider a weighted median filter defined by the real-valued weights $\{w_1, w_2, w_3, w_4, w_5\} = \{0.1, 0.2, 0.3, 0.2, 0.1\}$. The output of this filter operating on the observation set $\{x_1, x_2, x_3, x_4, x_5\} = \{-2, 2, -1, 3, 6\}$ is found as follows. Summing the weights gives the threshold $T_0 = \frac{1}{2} \sum_{i=1}^5 w_i = 0.45$. The sorted samples, their corresponding weights, and the partial sum of weights are

sorted samples	-2,	-1,	2,	3,	6
corresponding weights	0.1,	0.3,	0.2,	0.2,	0.1
partial weight sums	0.9,	0.7,	<u>0.5</u> ,	0.3,	0.1.

The weighted median is 2 whose weight causes the weight sum to become larger than T_0 . The median is a special case of the weighted median in which all weights are the same. As the mean square error (MSE) of the median is larger than the CRLB under Laplacian distribution, it does not achieve the minimum MSE although it is the maximum likelihood estimate [59]. The ITM filter [7] outperforms the median filter in suppressing both Gaussian and Laplacian noise and does not require data sorting.

4.2.2 Iterative Truncated Arithmetic Mean Filter

Different from the mean filter that averages all samples and the median filter that chooses one sample as the output, the iterative truncated arithmetic mean (ITM) filter [7] iteratively truncates the extreme samples and uses the truncated mean as the filter's output. Starting from $\mathbf{x} = \mathbf{x}_0$, it truncates samples in $\mathbf{x}$ to a dynamic threshold as shown by Algorithm 1 in Chapter 3.

The ITM filter has two types of outputs. In this chapter, we only concern about the type I output denoted by

$$y_t(\mathbf{x}_0) = \text{mean}(\mathbf{x}). \quad (4.7)$$

Theoretical analysis in [7] shows that the ITM output starts from the mean and approaches the median by increasing the number of iterations. The stopping criterion S given in [7], which terminates the iteration automatically, enables the ITM filter outperforms the median filter in suppressing both Gaussian and Laplacian noise. The implementation given in [59] is faster than the median filter.

4.2.3 The Proposed Weighted ITM Filter with Positive Weights

The weighted ITM (WITM) filter is proposed based on the following theorems.

Theorem 1: For any finite data set $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ and weight set $\mathbf{w} = \{w_1, w_2, \dots, w_n\}$ with all weights being nonnegative rational numbers, the difference between the weighted mean μ_w and weighted median ϕ_w is never great than the weighted mean absolute deviation τ_w . The corresponding formula is

$$|\phi_w - \mu_w| \leq \tau_w \triangleq \sum_{i=1}^n w_i |x_i - \mu_w| / \sum_{i=1}^n w_i. \quad (4.8)$$

Proof: Let

$$\mathbf{x}_e = \{kw_1 \diamond x_1, kw_2 \diamond x_2, \dots, kw_n \diamond x_n\} \quad (4.9)$$

be the expanded data set of $\mathbf{x}$ where k is a constant making kw_i integer for all $1 \leq i \leq n$.

Let n_e , μ_e , ϕ_e and $\tau_e = \text{mean}(|\mathbf{x}_e - \mu_e|)$ be the number of samples, mean, median and mean absolute deviation of $\mathbf{x}_e$, respectively. μ_e divides $\mathbf{x}_e$ into two subsets

$$\mathbf{x}_{eh} = \{x_i | x_i > \mu_e, x_i \in \mathbf{x}_e\}, \quad (4.10)$$

and

$$\mathbf{x}_{el} = \{x_i | x_i \leq \mu_e, x_i \in \mathbf{x}_e\}. \quad (4.11)$$

Let n_{eh} , n_{el} , μ_{eh} and μ_{el} denote the numbers and means of these two set. Obviously, $n_{eh} + n_{el} = n_e$, $\mathbf{x}_{eh} \cup \mathbf{x}_{el} = \mathbf{x}_e$ and

$$\begin{aligned} n_{eh}\mu_{eh} + n_{el}\mu_{el} &= n_e\mu_e \\ &= n_{eh}\mu_e + n_{el}\mu_e. \end{aligned} \quad (4.12)$$

Let $\delta_{eh} = \mu_{eh} - \mu_e$ and $\delta_{el} = \mu_e - \mu_{el}$. With some manipulation (4.12) becomes

$$n_{eh}\delta_{eh} = n_{el}\delta_{el}. \quad (4.13)$$

As the mean absolute deviation τ_e of the expanded data set $\mathbf{x}_e$ can be represented by

$$\begin{aligned} \tau_e &= \frac{1}{n_e} \left[\sum_{x_i > \mu_e} (x_i - \mu_e) + \sum_{x_i \leq \mu_e} (\mu_e - x_i) \right] \\ &= \frac{1}{n_e} [n_{eh}(\mu_{eh} - \mu_e) + n_{el}(\mu_e - \mu_{el})] \\ &= \frac{1}{n_e} (n_{eh}\delta_{eh} + n_{el}\delta_{el}), \end{aligned} \quad (4.14)$$

substituting (4.13) into (4.14) yields

$$\begin{aligned} \tau_e &= 2 \frac{n_{eh}}{n_e} \delta_{eh} \\ &= 2 \frac{n_{el}}{n_e} \delta_{el}. \end{aligned} \quad (4.15)$$

Let $n_{\tau eu}$ and $n_{\tau el}$ be the numbers of samples larger than $\mu_e + \tau_e$ and smaller than $\mu_e - \tau_e$, respectively. For $n_{\tau el} \neq 0$, we have

$$\begin{aligned} n_{\tau el}\tau_e &< \sum_{x_i < \mu_e - \tau_e} (\mu_e - x_i) \\ &\leq \sum_{x_i < \mu_e} (\mu_e - x_i). \end{aligned} \quad (4.16)$$

Since

$$\sum_{x_i < \mu_e} (\mu_e - x_i) = n_{el}\delta_{el}, \quad (4.17)$$

(4.16) becomes

$$n_{\tau el} < \frac{n_{el}\delta_{el}}{\tau_e}. \quad (4.18)$$

Substituting (4.15) into (4.18) yields

$$n_{\tau_{el}} < n_e/2. \quad (4.19)$$

As n_e is the number of samples in $\mathbf{x}_e$, according on the definition of the median we have

$$\phi_e \geq \mu_e - \tau_e. \quad (4.20)$$

Similarly, we have

$$\phi_e \leq \mu_e + \tau_e. \quad (4.21)$$

(4.20) and (4.21) prove that

$$|\mu_e - \phi_e| \leq \tau_e. \quad (4.22)$$

It is easy to see that $\mu_w = \mu_e$, $\phi_w = \phi_e$ and $\tau_w = \tau_e$. This completes the proof of Theorem 1. ■

Theorem 1 guarantees that the weighted median is never changed if we use the weighted dynamic threshold τ_w to truncate the extreme samples of $\mathbf{x}$. The following theorem ensures that the truncating process never idles for any data distribution if the weighted mean μ_w of $\mathbf{x}$ deviates from its weighted median ϕ_w .

Theorem 2: For any finite data set $\mathbf{x}$ and weight set $\mathbf{w}$, there exists at least one sample whose distance from the weighted mean μ_w is greater than the weighted mean absolute deviation τ_w if the weighted mean μ_w deviates from the weighted median ϕ_w , i.e.

$$\exists x_i, x_i \in \mathbf{x}, \text{ that } |x_i - \mu_w| > \tau_w, \text{ if } \mu_w \neq \phi_w. \quad (4.23)$$

Proof: According to the definition of the median, $n_{eh} \neq n_{el}$ if $\mu_e \neq \phi_e$ for the expanded data set $\mathbf{x}_e$. Therefore, from (4.15) we have $\delta_{eh} \neq \delta_{el}$ and $\tau_e < \max(\delta_{eh}, \delta_{el})$. From the definition of δ_{eh} and δ_{el} , it is obvious that there exists at least one sample larger than or equal to $\max(\delta_{eh}, \delta_{el})$.

Thus, we have

$$\exists x_i, x_i \in \mathbf{x}_e, \text{ that } |x_i - \mu_e| > \tau_e, \text{ if } \mu_e \neq \phi_e. \quad (4.24)$$

This means that at least one sample in $\mathbf{x}_e$ is far away from μ_e by a distance larger than τ_e . As $\mu_w = \mu_e, \phi_w = \phi_e, \tau_w = \tau_e$ and $x_i \in \mathbf{x}$ if $x_i \in \mathbf{x}_e$, (4.23) is proven. ■

Theorems 1 and 2 ensure that the extreme samples in $\mathbf{x}$ can be iteratively truncated by using the dynamic threshold τ_w while keeping the weighted median un-changed. These theorems inspire the proposed WITM filter shown in Algorithm 3.

Algorithm 3: Truncation Procedure of the WITM Filter

Input: $\mathbf{w}, \mathbf{x}_0 \Rightarrow \mathbf{x}$; **Output:** Truncated $\mathbf{x}$;

1 **do**

2   Compute the weighted mean: $\mu_w = \sum_{i=1}^n w_i x_i / \sum_{i=1}^n w_i$;

3   Compute the weighted dynamic threshold: $\tau_w = \sum_{i=1}^n w_i |x_i - \mu_w| / \sum_{i=1}^n w_i$;

4    $b_l = \mu_w - \tau_w, b_u = \mu_w + \tau_w$, and truncate $\mathbf{x}$ by:

$$x_i = \begin{cases} b_u, & \text{if } x_i > b_u \\ b_l, & \text{if } x_i < b_l \\ x_i, & \text{otherwise} \end{cases};$$

5 **while** the stopping criterion S is violated ;

The following example illustrates the truncation procedure of Algorithm 3. Consider a WITM filter defined by the positive weight set $\{w_1, w_2, w_3, w_4, w_5\} = \{0.1, 0.2, 0.3, 0.2, 0.1\}$. Let the input set be $\{x_1, x_2, x_3, x_4, x_5\} = \{-2, 2, -1, 3, 6\}$. The truncated samples of the first 4 iterations are shown as follows:

1st iteration   -0.98, **2**, -0.98,   3,   3.4

2nd iteration   -0.75, **2**, -0.75,   2.9,   2.9

3rd iteration   -0.54, **2**, -0.54,   2.7,   2.7

4th iteration   -0.37, **2**, -0.37,   2.6,   2.6.

The weighted median is in bold. It is seen that all the truncated samples approach the weighted median. The convergence property is guaranteed by the following proposition.

Proposition 1: The dynamic threshold $\tau_w(k)$ of the WITM algorithm monotonically decreases to zero by increasing the number of iteration k if the weighted mean μ_w deviates from the weighted median ϕ_w , i.e.

$$\tau_w(k+1) < \tau_w(k), \text{ if } \mu_w \neq \phi_w, \quad (4.25)$$

and

$$\lim_{k \rightarrow \infty} \tau_w(k) = 0, \text{ if } \mu_w \neq \phi_w. \quad (4.26)$$

Proof: For symbolic simplicity, we omit the index k wherever no ambiguity is caused. From (4.15) we have

$$\begin{aligned} \delta_{eh} + \delta_{el} &= \frac{1}{2} n_e \tau_e(k) \left(\frac{1}{n_{eh}} + \frac{1}{n_{el}} \right) \\ &= \frac{n_e^2}{2n_{eh}n_{el}} \tau_e(k). \end{aligned} \quad (4.27)$$

Due to the truncating process of the data in the previous iteration, we have $\delta_{eh} + \delta_{el} < 2\tau_e(k-1)$, if $\mu_e \neq \phi_e$. Therefore,

$$\begin{aligned} \tau_e(k) &= \frac{2n_{eh}n_{el}}{n_e^2} (\delta_{eh} + \delta_{el}) \\ &< \frac{4n_{eh}n_{el}}{n_e^2} \tau_e(k-1) \\ &= \frac{(n_{eh} + n_{el})^2 - (n_{eh} - n_{el})^2}{(n_{eh} + n_{el})^2} \tau_e(k-1) \\ &= \left[1 - \left(\frac{|n_{eh} - n_{el}|}{n_e} \right)^2 \right] \tau_e(k-1). \end{aligned} \quad (4.28)$$

If $\mu_e \neq \phi_e$, it is easy to find that $1 \leq |n_{eh} - n_{el}| \leq n_e$. Therefore, from (4.28) we have

$$\tau_e(k) < \tau_e(k-1), \quad (4.29)$$

and

$$\tau_e(k) < \tau_e(1) \left(1 - \frac{1}{n_e^2}\right)^{k-1}. \quad (4.30)$$

From (4.30) it is straightforward that

$$\lim_{k \rightarrow \infty} \tau_e(k) = 0, \text{ if } \mu_e \neq \phi_e. \quad (4.31)$$

As $\mu_w = \mu_e$, $\phi_w = \phi_e$ and $\tau_w = \tau_e$, (4.29) and (4.31) prove (4.25) and (4.26), respectively. ■

The output of the WITM filter is defined as the weighted mean of the truncated $\mathbf{x}_0$ by Algorithm 3

$$y_{wt}(\mathbf{x}_0, \mathbf{w}) = \sum_{i=1}^n w_i x_i / \sum_{i=1}^n w_i, \quad x_i \in \mathbf{x}. \quad (4.32)$$

By using the weighted mean of the truncated data set $\mathbf{x}$ as the filter output, the WITM filter is expected to own merits of both the weighted mean and median filters.

4.2.4 Stopping Criterion

The output of the WITM filter moves from the weighted mean towards the weighted median by increasing the number of iterations. Since for many applications, neither weighted mean nor weighted median is the optimal solution, proper stopping criterion may enable the WITM filter outperforming the both.

In order to facilitate the following analysis, we separate the data set $\mathbf{x}$ into two subsets by the truncated weighted mean μ_w as

$$\mathbf{x}_h \triangleq \{x_i | x_i \in \mathbf{x}, x_i > \mu_w\} \quad (4.33)$$

$$\mathbf{x}_l \triangleq \{x_i | x_i \in \mathbf{x}, x_i \leq \mu_w\}. \quad (4.34)$$

Let w_h and w_l denote the sum of weights of $\mathbf{x}_h$ and $\mathbf{x}_l$, respectively. One possible stopping

Figure 4.1: Ascending sorted data $x_{(1)} \leq x_{(2)} \leq \dots \leq x_{(n)}$, their corresponding weights $\{w_{(1)}, w_{(2)}, \dots, w_{(n)}\}$ and one possible location of the truncated weighted mean μ_w . The weighted median ϕ_w is denoted by $x_{(m)}$.

criterion S_1 to ensure μ_w close to the weighted median is to meet the condition

$$S_1(\varepsilon_1) : \Delta w \triangleq |w_h - w_l| \leq \varepsilon_1. \quad (4.35)$$

For real data, in general there is no more than one sample having the value equal to the weighted median. The following lemmas analyze the choice of ε_1 in this general case. Although the WITM filter does not need data sorting, we use the ascending sorted data set $\{x_{(1)}, x_{(2)}, \dots, x_{(n)}\}$ as shown in Fig. 4.1 to facilitate the analysis. The corresponding weight set is $\{w_{(1)}, w_{(2)}, \dots, w_{(n)}\}$. Let $x_{(m)}$ denote the weighted median ϕ_w . The following lemma gives the condition that ensures μ_w falls in the interval $(x_{(m-1)}, x_{(m+1)})$.

Lemma 1: Let x_q be the nearest sample to the weighted mean μ_w in the subset that has the larger sum weight. x_q is the weighted median if and only if

$$|w_h - w_l| \leq 2w_q. \quad (4.36)$$

Proof: If $w_h > w_l$,

$$w_h > T_0. \quad (4.37)$$

From (4.36) we have $w_h - w_q \leq w_l + w_q$. Therefore

$$w_h - w_q \leq T_0. \quad (4.38)$$

(4.37) and (4.38) show $x_q = x_{(m)}$ by the definition of weighted median. Similarly, it is straightforward to see that if $x_q = x_{(m)}$, (4.36) holds. The proof for the case $w_h \leq w_l$ is analogous. ■

We see that (4.36) is the sufficient and necessary conditions for μ_w close to $x_{(m)}$, $x_{(m-1)} < \mu_w < x_{(m+1)}$. The next lemma gives further conditions that $x_{(m-1)} < \mu_w < x_{(m)}$ and $x_{(m)} \leq \mu_w < x_{(m+1)}$. Let $w_{(l)} = \sum_{i=1}^{m-1} w_{(i)}$ and $w_{(h)} = \sum_{i=m+1}^n w_{(i)}$. Without loss of generality, we assume $w_{(h)} \geq w_{(l)}$.

Lemma 2: If μ_w falls in $(x_{(m-1)}, x_{(m)})$,

$$w_{(m)} \leq |w_h - w_l| \leq 2w_{(m)}. \quad (4.39)$$

Otherwise, if μ_w falls in $[x_{(m)}, x_{(m+1)})$,

$$|w_h - w_l| \leq w_{(m)}. \quad (4.40)$$

Proof: If μ_w is in $(x_{(m-1)}, x_{(m)})$, we have

$$|w_h - w_l| = w_{(m)} + w_{(h)} - w_{(l)} \geq w_{(m)}. \quad (4.41)$$

From (4.36) and (4.41), (4.39) is proven. If μ_w is in $[x_{(m)}, x_{(m+1)})$, we have

$$|w_h - w_l| = -[w_{(h)} - (w_{(m)} + w_{(l)})] \leq w_{(m)}. \quad (4.42)$$

This completes the proof of Lemma 2. ■

As Δw is smaller for $w_{(h)} > w_{(l)}$ if μ_w falls in $[x_{(m)}, x_{(m+1)})$ than in $(x_{(m-1)}, x_{(m)})$, (4.40) is in general a better stop criterion than (4.36). The following lemma proves that the condition (4.40) can always be met. A necessary proposition is given here to facilitate the proof of the following Lemma 3.

Proposition 2: Samples, once being truncated in an iteration of the WITM algorithm, must be truncated in all subsequent iterations.

The proof can be achieved by expanding $\mathbf{x}$ with the weight set $k\mathbf{w}$ and following the proposition 1 in Chapter 3.

Lemma 3: Assume $w_{(h)} > w_{(l)}$. There is an iteration k in which the weighted truncated mean μ_w falls in the interval $[x_{(m)}, x_{(m+1)})$, i.e.

$$\exists k, x_{(m)} \leq \mu_w(k) < x_{(m+1)}. \quad (4.43)$$

Proof: As the dynamic threshold τ_w monotonically decreases to zero, all samples except the weighted median $x_{(m)}$ will be truncated to the lower bound b_l or upper bound b_h after some iterations. Only three different sample values exist in the truncated $\mathbf{x}$, b_l , $x_{(m)}$ and b_h with the weights $w_{(l)}$, $w_{(m)}$ and $w_{(h)}$. If μ_w is in the interval $(b_l, x_{(m)})$, we will prove that it will move into in the interval $[x_{(m)}, b_h)$ in a finite number of iterations.

For symbolic simplicity of the proof, let $x_{(m)} = 0$ and $w_{(l)} + w_{(m)} + w_{(h)} = 1$. This will not lose the generality of the proof. So we have

$$\mu_w = w_{(l)}b_l + w_{(h)}b_h. \quad (4.44)$$

If $\mu_w \in (b_l, x_{(m)})$, $\mu_w < 0$. Therefore,

$$w_{(h)}b_h < -w_{(l)}b_l. \quad (4.45)$$

Both b_l and b_h are truncated to the new lower bound $b'_l = \mu_w - \tau_w$ and upper bound $b'_h = \mu_w + \tau_w$ in the next iteration based on Proposition 2. As

$$\tau_w = w_{(l)}(\mu_w - b_l) - w_{(m)}\mu_w + w_{(h)}(b_h - \mu_w) \quad (4.46)$$

and $w_{(l)} + w_{(m)} + w_{(h)} = 1$, we have

$$\begin{aligned}
 b'_h &= \mu_w + \tau_w = w_{(h)}b_h - w_{(l)}b_l + 2w_{(l)}\mu_w \\
 &= w_{(h)}b_h(1 + 2w_{(l)}) - w_{(l)}b_l(1 - 2w_{(l)}) \\
 &> w_{(h)}b_h(1 + 2w_{(l)}) + w_{(h)}b_h(1 - 2w_{(l)}) \\
 &= 2w_{(h)}b_h.
 \end{aligned} \tag{4.47}$$

The inequality in (4.47) comes from (4.45) and $(1 - 2w_{(l)}) > 0$. This yields

$$\begin{aligned}
 \alpha_{s+1} &\triangleq \frac{-b'_l}{b'_h} = \frac{-\mu_w + \tau_w}{\mu_w + \tau_w} = 1 - \frac{2\mu_w}{\mu_w + \tau_w} \\
 &< 1 - \frac{2(w_{(l)}b_l + w_{(h)}b_h)}{2w_{(h)}b_h} = -\frac{w_{(l)}}{w_{(h)}} \frac{b_l}{b_h} \\
 &= \frac{w_{(l)}}{w_{(h)}} \alpha_s,
 \end{aligned} \tag{4.48}$$

where $\alpha_s \triangleq -b_l/b_h$. Therefore, if $\mu_w \in (x_{(m-1)}, x_{(m)})$ in the k^{th} iteration,

$$\alpha_k < \alpha_s \left(\frac{w_{(l)}}{w_{(h)}} \right)^{k-s}, k \geq s. \tag{4.49}$$

As $w_{(l)}/w_{(h)} < 1$, there exists an iteration k in which $\alpha_k \leq w_{(h)}/w_{(l)}$. It leads to $\mu_w \geq 0$ in the $(k+1)^{th}$ iteration. This completes the proof of Lemma 3. ■

Lemmas 1, 2 and 3 imply that $\varepsilon_1 = w_q$ can be used as a stopping criterion to ensure μ_w close to ϕ_w . To avoid searching x_q for w_q , a loosen condition $\varepsilon_1 = w_{max}$ is utilized in this chapter, where w_{max} is the maximum value in the weight set $\mathbf{w}$.

In some extreme cases, there could exist multiple samples having the same weighted median value. In this case, the stopping criterion S_1 may never be met. The second stopping criterion

uses a predefined ε_2 to limit the maximum number of iterations k , defined as

$$S_2(\varepsilon_2) : k \geq \varepsilon_2. \quad (4.50)$$

The ε_2 we chosen in this chapter is the same as in [7], which is $\varepsilon_2 = 2\sqrt{n}$.

The stopping criterion S we used in this work to stop the WITM algorithm is a combination of S_1 and S_2 , i.e.

$$S = S_1(\varepsilon_1) \vee S_2(\varepsilon_2). \quad (4.51)$$

The above stopping criterion ensures that the filter output reasonably approaches the weighted median and that it never fails to stop the iteration for any kind of data. In fact, it is very difficult if not impossible to find a stopping criterion optimal for all types of signals and noise. The above stopping criterion is designed for general cases. For the noise types whose optimal location estimator is neither weighted mean nor weighted median, as shown in the experiments, the proposed WITM filter that stops the iteration by the above criterion will outperform the weighted mean and weighted median. However, this stopping criterion is by no means optimal for all types of noise. For particular application, specific stopping criterion could be designed to make the WITM filter closer to the weighted mean or closer to the weighted median filters and hence to achieve a better performance.

4.2.5 Properties of the WITM Filter with Positive Weights

Property 1: The WITM filter's output converges to the weighted median by increasing the number of iterations k , i.e.

$$\lim_{k \rightarrow \infty} y_{wt}(\mathbf{x}_0, \mathbf{w}) = \phi_w(\mathbf{x}_0, \mathbf{w}). \quad (4.52)$$

Proof: As shown in Theorem 1, the truncating algorithm does not change the weighted median of the input data set. Moreover, Proposition 1 shows that the dynamic threshold converges

to zero. Therefore, the output of the WITM filter converges to the weighted median. ■

Property 2: The WITM filter's output is scale and shift invariant, i.e. if $\mathbf{z} = \{\alpha x_i + c\}, \forall x_i, x_i \in \mathbf{x}_0$, we have

$$y_{wt}(\mathbf{z}, \mathbf{w}) = \alpha y_{wt}(\mathbf{x}_0, \mathbf{w}) + c, \quad (4.53)$$

where α and c are two constants. The proof is trivial and hence omitted.

Property 3: The distribution of the WITM filter's output is symmetric if the samples of the input data set $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are drawn from the random variables $\{X_1, X_2, \dots, X_n\}$, all of which have symmetric distributions around the symmetry center c .

Proof: If x_i is symmetrically distributed around c , $2c - x_i$ has the same distribution as x_i . According to Property 2, $y_{wt}(2c - \mathbf{x}_0, \mathbf{w}) = 2c - y_{wt}(\mathbf{x}_0, \mathbf{w})$. Thus, the distribution of $y_{wt}(\mathbf{x}_0, \mathbf{w})$ is symmetric around c . ■

Property 4: The WITM filter's output is an unbiased estimate of the symmetry center c if the samples in $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are drawn from the random variables $\{X_1, X_2, \dots, X_n\}$, all of which have symmetric distributions around c .

Proof: According to Property 3, $y_{wt}(\mathbf{x}_0, \mathbf{w})$ is symmetrically distributed around c . Therefore, $E\{y_{wt}(\mathbf{x}_0, \mathbf{w})\} = E\{X_i\} = c$. This completes the proof of Property 4. ■

4.3 Weighted ITM Filter Admitting Negative Weights

Similar to the weighted mean and median filters, the WITM filter admitting only positive weights can only be a low-pass filter. In this section, two structures of the WITM filter admitting both positive and negative weights, named GWITM and LCWITM filters, are designed following the structures of the general weighted median (GWM) filter [84] and the linear combination of weighted median (LCWM) filter [100]. A new structure, named dual WITM (DWITM) filter, is proposed. The three structures enable the WITM filter being designed as low-, band- and

high-pass filters.

4.3.1 General WITM Filter with Negative Weights

The GWM filter [84] admitting negative weights is

$$y_{gwm} = \text{median}(|w_1| \diamond \text{sign}(w_1)x_1, |w_2| \diamond \text{sign}(w_2)x_2, \dots, |w_n| \diamond \text{sign}(w_n)x_n), \quad (4.54)$$

where $\text{sign}(w) = 1$ if $w \geq 0$ and $\text{sign}(w) = -1$ otherwise. By uncoupling the weight sign from the weight magnitude and merging it with the sample values, the GWM filter can be implemented by the algorithm of weighted median filter with only positive weights. The general WITM (GWITM) filter is analogous to that of the GWM filter. It is

$$y_{gwt}(\mathbf{x}_0, \mathbf{w}) = y_{wt}(\text{sign}(\mathbf{w}) \cdot \mathbf{x}_0, \text{abs}(\mathbf{w})), \quad (4.55)$$

where $\text{sign}(\mathbf{w}) \cdot \mathbf{x} = \{\text{sign}(w_1)x_1, \text{sign}(w_2)x_2, \dots, \text{sign}(w_n)x_n\}$ and $\text{abs}(\mathbf{w}) = \{|w_1|, |w_2|, \dots, |w_n|\}$.

The GWITM filter with negative weights turns to that with only positive weights which can be implemented by Algorithm 3. Take the observation set $\{x_1, x_2, x_3, x_4, x_5\} = \{-2, 2, -1, 3, 6\}$ and the weight set $\{w_1, w_2, w_3, w_4, w_5\} = \{0.1, 0.2, 0.3, -0.2, 0.1\}$ as an example. Using Algorithm 3, the truncated "signed" samples of the first 4 iterations are shown as follows.

1 st iteration	-2,	2,	-1 ,	-2.41,	2.18
2 nd iteration	-2,	1.24,	-1 ,	-2.05,	1.24
3 rd iteration	-1.82,	0.63,	-1 ,	-1.82,	0.63
4 th iteration	-1.64,	0.17,	-1 ,	-1.64,	0.17.

The weighted median is in bold. It shows that all the truncated samples approach to the weighted median.

4.3.2 Linear Combined WITM Filter with Negative Weights

Although the GWM filter has been widely used in many applications [86,90], it has limitation in suppressing the DC component of the signal. Take a random data set $\{7.91, 9.92, 10.24, 10.03, 9.79, 11.47\}$ and their corresponding weight set $\{0.2, 0.3, 0.2, -0.2, -0.3, -0.2\}$ as an example. The observation set is Laplacian noise with the offset value 10. The "signed" data set is $\{7.91, 9.92, 10.24, -10.03, -9.79, -11.47\}$. Although the input data has a small variance, the "signed" data has a large variance and there is a large gap between the positive and the negative samples due to the large offset of the input samples. As the GWM filter selects one of the "signed" samples as the output, it cannot suppress the constant signal effectively. This problem is still not well solved though in [9] the output of the GWM filter is modified to be the average of the weighted median and the next smaller "signed" sample in the sorted data. This phenomenon can be seen by comparing the filter's outputs in Fig. 4.2 and 4.3. Fig. 4.2(a) shows a chirp signal with zero mean. The linear FIR, GWM and GWITM filters are designed as band-pass filters with pass band $[0.1\pi, 0.2\pi]$. The setting of these filters is detailed in the experiment section. The output of the linear FIR filter shown in Fig. 4.2(b) is used as a reference. The outputs of the GWM and GWITM filters are depicted in Fig. 4.2(c) and (d), respectively. All of them have the frequency selection characteristic. The chirp signal adding a constant is shown in Fig. 4.3(a). The corresponding outputs of the linear FIR, GWM and GWITM filters are shown in Fig. 4.3(b), (c) and (d), respectively. It is seen that the GWM filter fails to select the frequency. The output of the GWITM filter has some distortions though it is much better than that of the GWM filter. In order to alleviate this problem, we propose the linear combined WITM (LCWITM) filter. It is based on the LCWM filter [100] that utilized a combination of n low-pass weighted median sub-filters to design band- and high-pass filters. The LCWM filter is defined by

$$y_{LCWM} = \sum_{i=1}^n \alpha_i y_{wm}(\mathbf{x}_0, \mathbf{w}_i), \quad (4.56)$$

Figure 4.2: Outputs of frequency selective filters on chirp signal. (a) Chirp signal. (b) Output of linear FIR filter. (c) Output of GWM filter. (d) Output of GWITM filter.

where $y_{wm}(\mathbf{x}_0, \mathbf{w}_i)$ is the i^{th} weighted median sub-filter with the weight set $\mathbf{w}_i$. $\mathbf{w}_i$ is designed using the algorithm in [100] with the help of the combination matrix $B_{n,m}$ [100] where m is the number of nonzero elements of each sub-filter. The weighting coefficient α_i of the i^{th} sub-filter is calculated based on the coefficients of a prototype FIR filter designed by any of the standard FIR design tool [100].

The structure of the LCWITM filter is set to be the same as that of the LCWM filter. By directly replacing the weighted median filter with the WITM filter, the resulting LCWITM filter is

$$y_{lcwt}(\mathbf{x}_0, \mathbf{w}) = \sum_{i=1}^n \alpha_i y_{wt}(\mathbf{x}_0, \mathbf{w}_i). \quad (4.57)$$

In this chapter, both $\mathbf{w}_i$ and α_i are designed following the method given in [100].

4.3.3 The Proposed Dual WITM Filter with Negative Weights

As distinct low-pass weighted median filters are countable [101], it is not available to achieve arbitrary frequency response by only using two such sub-filters [100]. Therefore, the LCWM

Figure 4.3: Outputs of frequency selective filters on chirp signal with a constant offset. (a) Chirp signal with a constant offset. (b) Output of linear FIR filter. (c) Output of GWM filter. (d) Output of GWITM filter.

filter employs n sub-filters to alleviate this problem. The n sub-filter structure leads to a high computational complexity for the LCWITM filter because it needs truncate the data independently for each sub-WITM filter. Moreover, the LCWM filter employs small-size sub-filters to makes its output close to the linear filter. This, however, reduces the filter's capability in suppressing impulsive noise. This also makes the WITM filter stop too early to suppress the impulsive noise. This observation motivates the proposed dual WITM (DWITM) filter.

According to the sign of the weights, the weight set $\mathbf{w}$ can be separated into two subsets: positive subset $\mathbf{w}_P = \{w_{P1}, w_{P2}, \dots, w_{Pa}\}$ and negative subset $\mathbf{w}_N = \{w_{N1}, w_{N2}, \dots, w_{Nb}\}$ containing all the positive and negative weights, respectively. The output of the weighted mean

filter can be represented as

$$\begin{aligned}
 \mu_w &= \sum_{i=1}^n w_i x_i / \sum_{i=1}^n |w_i| \\
 &= \sum_{i=1}^a w_{Pi} x_{Pi} / \sum_{i=1}^n |w_i| - \sum_{i=1}^b |w_{Ni}| x_{Ni} / \sum_{i=1}^n |w_i| \\
 &= \mu_P \sum_{i=1}^a w_{Pi} / \sum_{i=1}^n |w_i| - \mu_N \sum_{i=1}^b |w_{Ni}| / \sum_{i=1}^n |w_i|, \tag{4.58}
 \end{aligned}$$

where μ_P and μ_N are the weighted means of the samples corresponding to the positive and negative weights, respectively. Equation (4.58) shows that μ_w is the weighted difference between μ_P and μ_N . It means that the output of a band- or high-pass filter is the difference between two low-pass filters. Unlike the weighted median filter, the distinct WITM filters are uncountable because they use a truncated averaging instead of a selecting algorithm. Therefore, it is reasonable to design band- or high-pass filter with two low-pass WITM filters. The proposed DWITM filter is formulated as

$$y_{dwt}(\mathbf{x}_0, \mathbf{w}) = y_{wt}(\mathbf{x}_P, \mathbf{w}_P) \sum_{i=1}^a w_{Pi} / \sum_{i=1}^n |w_i| - y_{wt}(\mathbf{x}_N, \text{abs}(\mathbf{w}_N)) \sum_{i=1}^b |w_{Ni}| / \sum_{i=1}^n |w_i|, \tag{4.59}$$

where $\mathbf{x}_P$ and $\mathbf{x}_N$ are the subsets of $\mathbf{x}_0$ corresponding to $\mathbf{w}_P$ and $\mathbf{w}_N$, respectively. Take $\mathbf{x}_0 = \{7.91, 9.92, 10.24, 10.03, 9.79, 11.47\}$ and $\mathbf{w} = \{0.2, 0.3, 0.2, -0.2, -0.3, -0.2\}$ as an example. According to the signs of the weights, $\mathbf{w}$ and $\mathbf{x}_0$ are separated into $\mathbf{w}_P = \{0.2, 0.3, 0.2\}$, $\mathbf{w}_N = \{-0.2, -0.3, -0.2\}$, $\mathbf{x}_P = \{7.91, 9.92, 10.24\}$ and $\mathbf{x}_N = \{10.03, 9.79, 11.47\}$. The truncated samples and the corresponding outputs of the first 4 iterations are as follows

	truncated $\mathbf{x}_P$	truncated $\mathbf{x}_N$	y_{dwt}
1^{st}	8.56, 9.92, 10.24	10.03, 9.79, 10.99	-0.29
2^{nd}	9.02, 9.92, 10.23	10.03, 9.79, 10.65	-0.18
3^{rd}	9.33, 9.92, 10.17	10.03, 9.79, 10.42	-0.11
4^{th}	9.54, 9.92, 10.10	10.03, 9.82, 10.25	-0.07.

It is seen that the output of the DWITM filter y_{dwt} is not influenced by the constant offset of the input data.

4.4 Experiments

The first experiment tests the WITM filters in suppressing Gaussian and Laplacian noise. Frequency selective WITM filters are evaluated in the second experiment. High-pass WITM filters are tested in the third experiment. The forth experiment evaluates the filters on real image data. For the proposed WITM, GWITM, LCWITM, and DWITM filters, the same stopping criterion parameters $\varepsilon_1 = w_{max}$ and $\varepsilon_2 = 2\sqrt{n}$ are fixed over all experiments in this chapter. Better filtering performances than those shown in this chapter will be obtained if the aforementioned parameters are adjusted for different data sets. Similarly, for the weighted myriad filter, the default setting $K = 1$ provided in the source code [9] is used over all experiments. To show the good performance of the weighted myriad filter, a larger number of iterations $L = 20$ [7] is applied in all experiments though it is shown that the weighted myriad filter well converges in 10 iterations [61].

4.4.1 Attenuation of the Short- and Long-tailed Noise

The performance of the WITM filters is tested on a constant signal contaminated by Gaussian and Laplacian noise with non-identical distribution. For the input data set $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$,

Figure 4.4: Mean absolute error (MAE) normalized by that of the weighted median filter in suppressing non-identical distributed Gaussian noise against (a) the number of iterations with fixed filter size $n = 25$, and (b) the filter size n . The average numbers of iterations for the ITM filter are 1.71, 3.44, 5.42, 7.50 for the filter size from 9 to 81, respectively, and those for the WITM filter are 1.69, 2.51, 3.82, 5.40.

Figure 4.5: Mean absolute error (MAE) normalized by that of the weighted median filter in suppressing non-identical distributed Laplacian noise against (a) the number of iterations with fixed filter size $n = 25$, and (b) the filter size n . The average numbers of iterations for the ITM filter are 2.01, 3.98, 6.05, 8.17 for the filter size from 9 to 81, respectively, and those for the WITM filter are 2.27, 3.99, 5.96, 8.04. The MAE of the mean filter is drastically larger than those of other filters and hence not plotted. “F WITM” represents the WITM filter with fixed numbers of iterations of 7, 12, 17, 22 for the filter size from 9 to 81.

we assume the corresponding standard deviation of the noise be $\{\sigma_1, \sigma_2, \dots, \sigma_n\}$. The σ_i can be chosen as an arbitrary positive value. In this chapter, we restrict σ_i by setting $\sigma_n/\sigma_1 = 5$. The rest are set by $\sigma_2/\sigma_1 = \sigma_3/\sigma_2 = \dots = \sigma_n/\sigma_{n-1}$. As the optimal estimator under Gaussian noise is the weighted mean with weight set $\mathbf{w}_G = \{1/\sigma_1^2, 1/\sigma_2^2, \dots, 1/\sigma_n^2\}$, all weighted estimators under Gaussian noise use the weight set $\mathbf{w}_G$. Similarly, the ML estimator under Laplacian noise is the weighted median with weight set $\mathbf{w}_L = \{1/\sigma_1, 1/\sigma_2, \dots, 1/\sigma_n\}$. Therefore, in dealing with Laplacian noise, all weighted estimators employ the weight set $\mathbf{w}_L$. For each experiment, the mean absolute error (MAE) over 10^6 independent input data sets is used as the performance indicator.

Fig. 4.4 shows the normalized mean absolute error (MAE) of filters' outputs in suppressing the non-identical distributed Gaussian noise. The MAE is normalized by that of the weighted median filter. We first investigate the WITM filter's performance against the number of iterations without applying the proposed stopping criterion. Fig. 4.4(a) depicts the normalized MAEs of the ITM and WITM filters against the number of iterations. MAEs of the myriad filter with fixed 20 iterations and other noniterative filters are illustrated by horizontal lines for a better visual comparison with the ITM and WITM filters. The filter size is $n = 25$. As the weighted mean filter is the optimal in suppressing Gaussian noise, it has the lowest MAE. The MAE of the WITM filter increases against the number of iterations. It equals to that of the weighted mean filter when the number of iterations equals to zero, and approaches to that of the weighted median filter when the number of iterations is large enough. Fig. 4.4(b) shows the normalized MAE of the WITM filter with the proposed stopping criterion (4.51) against the filter size. The ITM filter employs the stopping criterion in [7]. Fig. 4.4(b) demonstrates that the performance of the WITM filter is significantly better than the weighted median filter and all un-weighted filters. The WITM filter achieves a comparable MAE to the weighted myriad filter. The normalized MAE of the WITM filter is smaller than that of the weighted myriad filter when the filter size $n \leq 49$.

Fig. 4.5 is generated from Fig. 4.4 by replacing Gaussian noise with Laplacian noise. Except for different noise types and weight sets, other experimental settings of Fig. 4.5 are the same as Fig. 4.4. Fig. 4.5(a) shows that the MAE of the WITM filter is smaller than that of the weighted median filter after only 3 iterations. Fig. 4.5(b) shows the weighted myriad filter achieves a comparable performance to that of the weighted median filter. The proposed WITM filter outperforms all other filters against all filter sizes. We also plot the MAE of the WITM filter with the fixed number of iterations that minimizes the filter's MAE shown as F WITM in Fig. 4.5(b). The numbers of iterations found by search in the training are 7, 12, 17 and 22 for the filter size 9, 25, 49 and 81, respectively. It shows that the WITM filter with the specific designed number of

Figure 4.6: Normalized running time against filter size n . The running time is normalized by that of the weighted median filter. The y -axis is in log scale. The weighted myriad filters with $L = 20$ and $L = 5$ iterations are both plotted.

iterations can achieve even smaller MAE than that using the general stopping criterion though the WITM filter with the general stopping criterion already outperforms the weighted median filter that is the ML location estimator for the Laplacian noise.

We compare the running time of the WITM filter with other filters under the Window 7 system with the Intel Core i5 CPU 3.2GHz and RAM 4GB. All the filters are implemented by the C programming language. The running time of these filters in suppressing the Gaussian and Laplacian noise is depicted in Fig. 4.6. Although this chapter applies $L = 20$ iterations to the weighted myriad filter for better performance, the running time of the weighted myriad filter with $L = 5$ iterations, which was reported in [61], is also plotted for a fair comparison. It is shown that the WITM filter is faster than both the weighted median and weighted myriad filters

over all filter sizes in Fig. 4.6.

4.4.2 Frequency Selective WITM Filters

A quadratic swept-frequency chirp signal spanning instantaneous angular frequency ranging from 0 to 0.5π is used to test the WITM filters in frequency selection. The weights setting for different filters are analogous to those in [84]. A 31-tap linear FIR filter with band-pass $0.1\pi \leq \omega \leq 0.2\pi$ is designed by Matlab's `fir1` function. The weights of the GWM [84], weighted myriad, GWITM and DWITM filters are set to be the same as those of the linear FIR filter. The LCWM filter is designed to be a symmetric LCWM filter with $B_{16,2}$ following the algorithm in [100]. The weights of the LCWITM filter are set to be identical to those of the LCWM filter.

Fig. 4.7(a)-top shows the chirp test signal. The output of the linear FIR filter depicted in Fig. 4.7(a)-bottom is used as a reference. The results of the GWM and GWITM filters are depicted in Fig. 4.7(b)-top and -bottom, respectively. It is seen that the GWITM filter has better performance than the GWM filter in suppressing the low and high frequency components. Fig. 4.7(c) shows that the output of the LCWITM filter (Fig. 4.7(c)-bottom) has smaller distortion than that of the LCWM filter (Fig. 4.7(c)-top). It is seen that both the DWITM filter (Fig. 4.7(d)) and the weighted myriad filter (Fig. 4.7(e)) produce small distortions compared to other nonlinear filters. The DWITM filter achieves drastically better performance than the median based filters. It achieves the comparable performance to the weighted myriad filter.

The chirp signal contaminated by the additive α -stable noise ($\alpha = 1.2$ and $\gamma = 0.1$) is shown in Fig. 4.8(a)-top. The output of the linear FIR filter is depicted in Fig. 4.8(a)-bottom. It is seen that the linear filter cannot remove the long-tailed noise effectively. The responses of the GWM and GWITM filters are shown in Fig. 4.8(b)-top and -bottom, respectively. The performance of the GWITM filter is better than that of the GWM filter. The parameter setting of the LCWM

Figure 4.7: Frequency selective filter outputs. (a)-top: Chirp test signal, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 3.21, and those for each sub-filter of the LCWITM and DWITM filters are 1.01 and 2.08, respectively.

Figure 4.8: Frequency selective filter outputs in noise. (a)-top: Chirp test signal in α -stable noise with $\alpha = 1.2$ and $\gamma = 0.1$, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 3.32, and those for each sub-filter of the LCWITM and DWITM filters are 1.06 and 2.87, respectively.

filter makes it contain 3-tap sub-filters. The small filter size reduces its capability in suppressing impulse noise, which can be seen from Fig. 4.8(c)-top. This structure also makes the LCWITM filter inefficient in suppressing impulse noise. In the LCWITM filter, the number of iterations for the 3-tap sub-filter is 1 because the stopping criterion is met in the first iteration. As the extreme samples are not sufficiently truncated, the performance of the LCWITM filter shown in Fig. 4.8(c)-bottom is poorer than that of the LCWM filter shown in Fig. 4.8(c)-top. The output of the DWITM filter (Fig. 4.8(d)) and that of the weighted myriad filter (Fig. 4.8(e)) show the best performance among all filters.

4.4.3 Design of High-pass WITM Filters

High-pass filters are tested on a two-tune signal with angular frequency 0.02π and 0.4π shown in Fig. 4.9(a)-top. Weights of different filters are set analogously to those in [84]. A 31-tap linear high-pass filter with a cut-off angular frequency 0.2π is designed by Matlab's `fir1` function. The output of the linear filter is shown in Fig. 4.9(a)-bottom. Instead of applying the weights of the linear FIR filter to all other filters, which may achieve the suboptimal results [84], the fast LMA algorithm [84] is used to optimize the GWM and GWITM filters with 31 taps for the application at hand. For the DWITM filter, as it has two sub-filters, its update function is modified from that given in [84]

$$w_i(k+1) = w_i(k) + \begin{cases} \beta e(k) \text{sign}(x_i(k) - y_{wt}(\mathbf{x}_P, \mathbf{w}_P)), & \text{if } w_i(k) > 0 \\ \beta e(k) \text{sign}(x_i(k) - y_{wt}(\mathbf{x}_N, \text{abs}(\mathbf{w}_N))), & \text{if } w_i(k) \leq 0 \end{cases} \quad (4.60)$$

where $e(k) = d(k) - y_{dwt}$, d is the desired signal, k is the time and β is the update step size. The step size used in all adaptive optimization experiments is $\beta = 0.001$. For the weighted myriad filter, the adaptive weighted myriad filter algorithm [98] is adopted to train the weights.

Figure 4.9: Frequency selective filter outputs. (a)-top: two-tune signal, (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 1.09, and those for each sub-filter of the LCWITM and DWITM filters are 1.00 and 2.18, respectively.

The LCWM and LCWITM filters are set the same as those in Section 4.4.2. The output of the GWM filter shown in Fig. 4.9(b)-top indicates that it still contains low frequency component. Besides, there are small distortions due to the “selection-type” behavior of the GWM filter. The performance of the GWITM filter depicted in Fig. 4.9(b)-bottom is better than the GWM filter. The LCWITM filter (Fig. 4.9(c)-bottom) generates smaller distortions than the LCWM filter (Fig. 4.9(c)-top). Fig. 4.9 shows that the outputs of the DWITM and weighted myriad filters are the closest to the linear filter.

Table 4.1: MAEs for the Filtered Two-tune Signal Contaminated by α -stable Noise.

filter	$\alpha = 0.9$	$\alpha = 1.2$	$\alpha = 1.5$	$\alpha = 1.8$	noise free
linear FIR	2.2458	0.3755	0.1720	0.1182	<u>0.0008</u>
GWM	0.1937	0.1685	0.1540	0.1468	0.1350
GWITM	0.1682	0.1184	0.0941	0.0819	0.0671
LCWM	0.6710	0.5852	0.5468	0.5253	0.4301
LCWITM	2.3493	0.4758	0.2872	0.2398	0.1502
DWITM	<u>0.1420</u>	<u>0.0985</u>	<u>0.0797</u>	<u>0.0693</u>	<u>0.0541</u>
W myriad	<u>0.1266</u>	<u>0.0983</u>	<u>0.0820</u>	<u>0.0734</u>	0.0647

The α -stable noise with $\gamma = 0.1$ and different α values is added to the two-tune signal. The MAE over 10^5 filter outputs is used as an indicator to evaluate the filters. The experimental results for 4 different α values are shown in Table 4.1. For each α value, the smallest MAE among all filters is underlined and in bold font, and the second smallest is in bold font. While the weighted myriad filter outperforms others for $\alpha = 0.9$ and $\alpha = 1.2$, the DWITM filter achieves the best performance for $\alpha = 1.5$ and $\alpha = 1.8$. For the case of noise free, it is not a surprise that the linear FIR filter gets the minimum MAE. The outputs of different filters on the two-tune signal in α stable noise ($\alpha = 1.2$, $\gamma = 0.1$) is shown in Fig. 4.10. This figure demonstrates that the linear filter cannot deal with the long tailed noise effectively. The GWM filter output has distortion though it can remove the long-tailed noise. Similar to the results in Section 4.4.2, Fig. 4.10(c) shows that the LCWM and LCWITM filters cannot effectively suppress the impulsive noise. The DWITM filter and the weighted myriad filter achieve similar

Figure 4.10: Frequency selective filter outputs in noise. (a)-top: two-tune signal in α -stable noise ($\alpha = 1.2$, $\gamma = 0.1$), (a)-bottom: linear FIR filter output. (b)-top: GWM filter output, (b)-bottom: GWITM filter output. (c)-top: LCWM filter output, (c)-bottom: LCWITM filter output. (d): DWITM filter output. (e): weighted myriad filter output. The average number of iterations for the GWITM filter is 1.36, and those for each sub-filter of the LCWITM and DWITM filters are 1.02 and 1.89, respectively.

results and outperform other filters.

We further test the high-pass filters' performances for different noise levels. The parameter settings for the filters are the same as before. ϵ -contaminated ($\epsilon = 0.5$) normal distributed noise with the distribution $P_\epsilon = \{(1 - \epsilon)\Phi + \epsilon H\}$ [60] is added on the two tune-signal, where Φ and H are Gaussian and α -stable ($\alpha = 1.5$) noise, respectively. The standard deviation σ of Gaussian noise is set the same as the dispersion parameter γ of the α -stable noise, i.e. $\sigma = \gamma$. Among the linear FIR, GWM, GWITM, DWITM and weighted myriad filters, the performance of the linear FIR filter turns from the worst to the best by increasing the signal to noise ratio (SNR). Thus, we choose the range of SNR so that the linear FIR filter performs from the worst to the best. Experimental results of the 5 weighted filters, linear FIR, GWM, GWITM, DWITM and weighted myriad filters are shown in Fig. 4.11. As the performances of the LCWM and LCWITM filters are much poorer than other filters, they are not shown in this figure. It is seen that the GWITM filter has a better performance than the GWM filter. The DWITM filter achieves the best performance among these 4 nonlinear filters.

4.4.4 Image Denoising

The original image "Lena" of size 512×512 is corrupted by ϵ -contaminated ($\epsilon = 0.5$) normal distributed noise with the distribution $P_\epsilon = \{(1 - \epsilon)\Phi + \epsilon H\}$ [60], where Φ and H are Gaussian ($\sigma^2 = 100$) and α -stable ($\alpha = 1.4, \gamma = 10$) noise, respectively. Noisy pixels which are out of the range $[0, 255]$ are truncated. The original and noise contaminated images are shown in Fig. 4.12(a) and (b). The filter size is 5×5 . For the weighted mean, weighted median and WITM filters, the fast LMA algorithm [84] is used to train the weights. For the weighted myriad filter, the algorithm in [96] for training the weighted myriad smoother is used to design the weights. Analogous to that in [86], the 60×60 image region of the bottom left part of the noisy "Lena" is used as the training data. The whole image is used to test the filters. For the switching bilateral

Figure 4.11: Normalized mean absolute error (MAE) against the signal to noise ratio (SNR) of input signal. The MAE is normalized by that of the weighted median filter. The MAEs of the LCWM and LCWITM filters are drastically larger than those of other filters and hence not plotted.

Figure 4.12: "Lena" of size 512×512 : (a) original image and (b) corrupted image by ϵ -contaminated ($\epsilon = 0.5$) Gaussian ($\sigma_n^2 = 100$) and α -stable ($\alpha = 1.4, \gamma = 10$) noise.

filter (SBF) [64, 102], as there are several parameters needed to design carefully, the default setting provided by the authors is used.

Table 4.2: MAEs, MSEs, PSNRs and Running Time of the Noise Contaminated "Lena" Image. For the Myriad and Weighted Myriad Filters, the Running Time with $L = 5$ Iterations is Shown in Brackets.

filter	MAE	MSE	PSNR	running time (s)
mean	5.59	70.93	29.62	<u>0.052</u>
median	4.65	51.83	30.99	0.51
ITM	4.58	51.26	31.03	0.39
myriad	5.86	79.51	29.13	2.42 (1.29)
W mean	5.31	58.73	30.44	0.058
W median	4.42	42.56	31.84	0.73
WITM	<u>4.26</u>	<u>39.75</u>	<u>32.14</u>	0.45
W myriad	4.97	62.79	30.15	2.62 (1.48)
SBF	4.82	48.97	31.23	1.11

The MAE, MSE and PSNR over 10 runs of the noise contaminated images are shown in Table 4.2. It shows that the weighted filters outperform the corresponding un-weighted filters and the WITM filter gets the best performance for all three indicators: MAE, MSE and PSNR. The average numbers of iterations for the ITM and WITM filters are 3.6 and 3.7, respectively.

Figure 4.13: PSNR of the filtered image against the noise level σ of input image.

The weighted myriad filter prefers the most repeated values of the samples [30, 96], which reduces its ability in preserving the image details. This adversely affects its performance in image denoising. The performance of the SBF filter is better than those of the weighted myriad and weighted mean filters, but poorer than those of the weighted median and WITM filters in suppressing the mixed Gaussian and α -stable noise. This experiment is expanded with different levels of noise with changing $\sigma = \gamma$ from 1 to 20. Results are shown in Fig. 4.13. It is seen that, for all the σ values in Fig. 4.13, the WITM filter has the best performance in this image denoising experiment. Besides “Lena”, other standard images “Peppers” and “Baboon” are also applied in the test. Their results are omitted as the relative performances of the filters are very similar to those of “Lena” reported in Table 4.2 and Fig. 4.13.

We compare the running time of the WITM filter with other nonlinear filters under the Window 7 system with the Intel Core i5 CPU 3.2GHz and RAM 4GB. All the filters are im-

plemented by the C programming language. The running time of these filters in filtering the “Lena” image is given in Table 4.2. It is shown that both the ITM and WITM filters are, though significantly slower than the mean and weighted mean filters, faster than other nonlinear filters in Table 4.2.

4.5 Summary

A rich class of filters named weighted ITM (WITM) filters are proposed in this chapter. Different from the weighted median filters which rely on the time-consuming data sorting, the WITM filters employ an iteratively arithmetic computing algorithm to approximate the weighted median. By iteratively truncating the extreme samples, the output of the WITM filter moves from the weighted mean towards the weighted median. The proposed stopping criterion enables the WITM filters being terminated within a few iterations in all experiments of this chapter. The WITM filters outperform both the weighted mean and weighted median filters in many de-noising applications. By employing the structures of the GWM and LCWM filters, the corresponding GWITM and LCWITM filters can be designed as band-pass and high-pass filters. Due to the limitation of the GWM filter’s structure, the GWITM filter cannot suppress the DC component effectively. The LCWM filter’s structure reduces its capability in suppressing impulsive noise. This structure also makes the LCWITM filter has poorer performance in suppressing impulsive noise. In order to alleviate these problems, the DWITM filter is proposed by utilizing the difference of two low-pass WITM filters to design band- and high-pass filters. The superiority of the proposed filters is demonstrated by the comprehensive simulation results.

Chapter 5

Iterative Trimmed and Truncated Arithmetic Mean Filter

In this chapter¹, an iterative trimmed and truncated arithmetic mean (ITTM) algorithm is proposed, and the corresponding ITTM filters are developed. Here, trimming a sample means removing it and truncating a sample is to replace its value by a threshold. Simultaneously trimming and truncating the extreme samples enables the proposed filters to attenuate the mixed additive and exclusive noise in an effective way. The proposed trimming and truncating rules ensure that the output of the ITTM filter converges to the median. It offers an efficient method to estimate the median without time-consuming data sorting. Our analysis shows that the ITTM filter has a linear computational complexity $\mathcal{O}(n)$. Compared to the median filter and the iterative truncated arithmetic mean (ITM) filter, the proposed ITTM filter suppresses noise more effectively in some cases and has lower computational complexity. Experiments on synthetic data and real images verify the filter's properties.

---

¹This work is partially submitted to Signal Processing (journal paper [3] in author's publication list).

5.1 Introduction

Noise suppression has drawn great attention and is used in a broad range of applications, such as imaging, communications, geology, hydrology and economics [9]. A noise corrupted signal can be modeled as

$$x_i = \begin{cases} s_i + v_i, & \text{with probability } p \\ e_i, & \text{with probability } (1 - p) \end{cases}, \quad (5.1)$$

where s_i , v_i and e_i denotes the noise free signal, the additive and exclusive noise, respectively. The occurrence probability of the two types of noise is controlled by p , $p \in [0, 1]$. The additive noise v_i is in general symmetrically distributed with zero mean. It could be short- or long-tailed noise, such as Gaussian or Laplacian noise. The exclusive noise e_i could be impulsive noise with uniform distribution, or pepper & salt noise. Great effort has been devoted to developing the noise suppression filters based on the noise model.

Many filters were designed to attenuate the additive noise that corresponds to $p = 1$. The most frequently occurring noise in practice is the additive Gaussian noise, and the optimal filter in suppressing it is the mean filter. Its simplicity in realization and the availability of rigorous mathematical tool lead to the rich class of the linear finite impulse response (FIR) filters. The linear FIR filters are effective in attenuating the additive Gaussian noise but not the long-tailed noise. This results in the development of the nonlinear filters. The median filter [9], which is the most widely used one among the nonlinear filters, provides a powerful tool for signal processing. It has good properties in long-tailed noise suppression and structure preservation. However, it destructs fine signal details and cannot effectively suppress the additive Gaussian and other short-tailed noise. This leads to the various extensions of the median filter, including the weighted median filters [9], the weighted rank order Laplacian of Gaussian filter [103, 104], the steerable weighted median filter [13], the fuzzy rank filters [15] and the truncation filters [18]. The merits of the mean and median filters lead to another branch of filters which

make compromises between these two filters, such as the mean-median (MEM) filter [25] and the median affine (MA) filter [26]. The output of these filters varies smoothly between the mean and median by adjusting some free parameters. Selecting the optimal parameters to make them adaptive to signal types is not an easy task though some efforts were made [26].

The exclusive noise occurs if p in (5.1) is smaller than 1. As this noise completely replaces the original sample, the effective way is to trim (remove) such samples and use the noise free ones in a local region to restore the signal. The median filter is optimal in suppressing the exclusive noise by trimming all the samples except the middle one. The filters to suppress the mixed additive and exclusive noise include the α -trimmed mean (α T) filter [22], the modified trimmed mean (MTM) filter [24] and the switching bilateral filter (SBF) [102]. The α T filter discards some samples strictly relying on their rank. This may not be effective as it does not consider the dispersion of the data [26]. The MTM filter is sensitive to the small variation of samples located close to the threshold [26]. The SBF filter separates the impulsive noise from the Gaussian noise [102] and suppresses these two types of noise. In case of the long-tailed noise, its performance drops. In addition, all these filters require both data sorting and arithmetic computing. Compared to the arithmetic computing, the data sorting is much more time-consuming [29].

Nonlinear filters without data sorting were desirable and proposed in [7, 59, 105]. The iterative truncated arithmetic mean (ITM) filter [7] iteratively truncates the extreme samples to a dynamic threshold that ensures the filter's output converges to the median. The stopping criterion of the ITM filter makes it own merits of both the arithmetic mean and order statistic median operations in attenuating the short- and long-tailed noise. By truncating the extreme samples, the ITM1 filter outperforms both the mean and median filters in suppressing Laplacian noise and the Gaussian-Laplacian mixed noise [7]. By discarding all the truncated samples and using the mean of the remaining ones as the output, the ITM2 filter surpasses other filters in attenuating the impulsive-contaminated Gaussian noise [7]. A realization of the ITM filter [59] is verified

to perform faster than the standard median filter. The ITM filter is extended to the weighted ITM filter to realize the band- and high-pass filters [105]. However, keeping all the truncated samples (ITM1) or trimming all the truncated samples (ITM2) may not be optimal if both the additive and exclusive noise exist. Moreover, further analysis in this chapter reveals that the truncation threshold are largely affected by the extreme samples even if they are truncated. This reduces the convergence of the truncation threshold, and therefore leads to a high computational complexity.

In this chapter, we propose a trimmed and truncated arithmetic mean (ITTM) algorithm to alleviate the above problems. The proposed algorithm iteratively trims and truncates the extreme samples simultaneously. Without sorting, the extreme samples are symmetrically trimmed from the input data set, and the remaining ones are truncated to a dynamic threshold. Three types of filter outputs are developed on the basis of the ITTM algorithm. The proposed trimming and truncating rules guarantee the filters' outputs approaching the median by increasing the number of iterations. With the stopping criterion given in [7] to terminate the iteration, the proposed ITTM filter is not only faster than the ITM filter, but also more effective in attenuating some types of noise.

5.2 The Proposed ITTM Filter

The iterative trimmed and truncated arithmetic mean (ITTM) filters are proposed following the analysis of the ITM filters.

5.2.1 Iterative Truncated Arithmetic Mean Filter

As distinct from the mean filter that averages all samples and the median filter that chooses one sample as the output, the iterative truncated arithmetic mean (ITM) filter [7] iteratively truncates

the extreme samples and uses the truncated mean as the filter output. Starting from $\mathbf{x} = \mathbf{x}_0$, it truncates samples in $\mathbf{x}$ to a dynamic threshold as shown by Algorithm 1 in Chapter 3.

The ITM filter has two types of outputs [7]. The type I output ITM1 is

$$y_{t1} = \text{mean}(\mathbf{x}). \quad (5.2)$$

Let $\mathbf{x}_r = \{x_i | b_l < x_i < b_u\}$ and n_r be the number of samples in $\mathbf{x}_r$. The type II output ITM2 is

$$y_{t2} = \begin{cases} \text{mean}(\mathbf{x}_r), & \text{if } n_r > \xi \\ \text{mean}(\mathbf{x}), & \text{otherwise} \end{cases}. \quad (5.3)$$

The parameter ξ is used to avoid an unreliable mean in case too few samples remain in $\mathbf{x}_r$. It is set to $\xi = n/4$ in [7].

5.2.2 The Proposed ITTM Filters

Keeping all the truncated samples makes the ITM1 filter less effective in suppressing the exclusive noise. Trimming all the truncated samples causes the ITM2 filter not optimal in dealing with the additive noise. Neither the ITM1 nor ITM2 filters can effectively deal with the case that both the additive and exclusive noise exist. In addition, a large number of iterations may be in demand for the ITM algorithm to converge.

Choosing the truncation threshold is a critical step in designing the ITM filter. The threshold should converge to zero yet ensure the sample median ϕ within the bounds, $b_l \leq \phi \leq b_u$. The mean absolute deviation (MAD) of the samples from the mean, i.e. $\tau \triangleq \text{mean}(|\mathbf{x} - \mu|)$, satisfies the above requirements and achieves the fastest convergence among the 3 possible thresholds given in [7]. Therefore, the MAD is chosen as the truncation threshold for the ITM filter. In the following, further analysis reveals that the extreme samples largely affect the MAD even though they are truncated.

Figure 5.1: MAD of the samples from the mean against the number of iterations. The input data set is $\mathbf{x}_0 = \{-7, -6, -6, -2, -1, 0, 1, 3, 6, 8, 24\}$.

Take a random set $\mathbf{x}_0 = \{-7, -6, -6, -2, -1, 0, 1, 3, 6, 8, 24\}$ as an example. In the first iteration of the ITM algorithm, $\mu = 1.8$ and $\tau = 6.1$. The truncated $\mathbf{x}$ of this iteration is

$$\text{ite 1: } \mathbf{x} = \{\underbrace{\underline{-4.3}, \underline{-4.3}, \underline{-4.3}}_{n_{\tau l}=3}, -2, -1, \mathbf{0}, 1, 3, 6, \underbrace{\overline{7.9}, \overline{7.9}}_{n_{\tau u}=2}\},$$

where the median is in bold font, samples truncated to the lower/upper bounds are underlined/overlined, and $n_{\tau l}$ and $n_{\tau u}$ denote the number of samples truncated to the lower and upper bounds, respectively. In the second iteration, the dynamic threshold decreases to $\tau = 3.9$. It is larger than the MAD of the un-truncated samples $\{-2, -1, 0, 1, 3, 6\}$ that equals 2.2. The truncated $\mathbf{x}$ of the second iteration is

$$\text{ite 2: } \mathbf{x} = \{\underbrace{\underline{-3.0}, \underline{-3.0}, \underline{-3.0}}_{n_{\tau l}=3}, -2, -1, \mathbf{0}, 1, 3, \underbrace{\overline{4.8}, \overline{4.8}, \overline{4.8}}_{n_{\tau u}=3}\}.$$

This shows that the extreme samples, though being truncated, still have great influence on the next dynamic threshold. The dynamic threshold τ against the number of iterations k is plotted in Fig. 5.1 by the solid line. It shows that τ decays slower with increasing k .

The influence of the extreme samples on the dynamic threshold is exposed by the following mathematic analysis. It is not difficult to find that the truncated means of the $(k+1)^{th}$ and k^{th} iterations, μ_+ and μ , follow

$$\mu_+ = \frac{2n_t}{n}\mu + \frac{n_\tau}{n}b + \frac{n_r}{n}\mu_r, \quad (5.4)$$

where $n_t = \min\{n_{\tau l}, n_{\tau u}\}$, $n_\tau = |n_{\tau u} - n_{\tau l}|$, $\mu_r = \text{mean}(\mathbf{x}_r)$ and

$$b = \begin{cases} \mu + \tau, & \text{if } n_{\tau u} > n_{\tau l} \\ \mu - \tau, & \text{if } n_{\tau u} \leq n_{\tau l} \end{cases}. \quad (5.5)$$

Similarly, the dynamic threshold of the $(k+1)^{th}$ iteration is

$$\tau_+ = \frac{2n_t}{n}\tau + \frac{n_\tau}{n}|b - \mu_+| + \frac{n_r}{n}\tau_r, \quad (5.6)$$

where $\tau_r = \text{mean}(|\mathbf{x}_r - \mu_+|)$. (5.4) and (5.6) indicate that μ_+ and τ_+ directly relate to μ and τ with the normalized number of truncated samples $2n_t/n$. It approaches to 1 by increasing the number of iterations. As $n_\tau/n + n_r/n = 1 - 2n_t/n$, the second and third terms in (5.6) play a weak role in determining the dynamic threshold. This leads to the decrease of the decay of the dynamic threshold as shown in Fig. 5.1.

Trimming the truncated samples in the subsequent iterations is helpful to increase the convergence since they slow down the decay of the dynamic threshold. Unfortunately, trimming all truncated samples may violate the rule of the ITM algorithm that the output converges to the median. As the median is within the truncation bounds, trimming the same number of the truncated samples from both sides of the mean does not change the median. If the number of trimmed samples from each side is $n_t = \min\{n_{\tau l}, n_{\tau u}\}$, the resulting mean and dynamic

threshold of the trimmed and truncated input data set are

$$\mu'_+ = \frac{n_\tau}{n - 2n_t}b + \frac{n_r}{n - 2n_t}\mu_r, \quad (5.7)$$

and

$$\tau'_+ = \frac{n_\tau}{n - 2n_t}|b - \mu'_+| + \frac{n_r}{n - 2n_t}\tau'_r, \quad (5.8)$$

where $\tau'_r = \text{mean}(|\mathbf{x}_r - \mu'_+|)$. From (5.6) and (5.8), it is clear that $\tau'_+ < \tau_+$ if the change of the mean has a negligible effect on τ'_+ .

For the previous given example, the trimmed and truncated $\mathbf{x}'$ of the first two iterations are

$$\text{ite 1: } \mathbf{x}' = \{\underline{-4.3}, -2, -1, \mathbf{0}, 1, 3, 6\},$$

and

$$\text{ite 2: } \mathbf{x}' = \{-2, -1, \mathbf{0}, 1, \overline{2.9}\}.$$

The curve of τ' against the number of iterations k is plotted in Fig. 5.1 by the dash line. It shows that the convergence of τ' is faster than τ .

The following Theorem 1 guarantees that trimming the extreme samples leads to a higher convergence of the truncation threshold.

Theorem 1: For any finite data set, simultaneously trimming the minimum and maximum samples decreases the MAD if the mean deviates from the median. Let $\mathbf{x} = \{x_a, \mathbf{x}_r, x_b\}$, where $\mathbf{x}_r = \{x_1, x_2, \dots, x_{n-2}\}$ and $x_a \leq \{x_i\}_{i=1}^{n-2} \leq x_b$, and τ and τ_r be the MAD of $\mathbf{x}$ and $\mathbf{x}_r$ respectively. We have

$$\tau_r < \tau, \text{ if } \mu \neq \phi, \quad (5.9)$$

where μ and ϕ are the mean and median of $\mathbf{x}$.

Proof. Let $\mu_r = \text{mean}(\mathbf{x}_r)$, and n_h and n_l denote the number of samples in $\{x_i | x_i \in$

$\mathbf{x}_r, x_i > \mu_r\}$ and $\{x_i | x_i \in \mathbf{x}_r, x_i \leq \mu_r\}$, respectively. The MAD τ_r of $\mathbf{x}_r$ satisfies

$$\begin{aligned}
 (n-2)\tau_r &= \sum_{i=1}^{n-2} |x_i - \mu_r| \\
 &= \sum_{x_i > \mu_r} (x_i - \mu_r) + \sum_{x_i \leq \mu_r} (\mu_r - x_i) \\
 &= \sum_{x_i > \mu_r} (x_i - \mu) + \sum_{x_i \leq \mu_r} (\mu - x_i) + (n_h - n_l)(\mu - \mu_r) \\
 &\leq \sum_{i=1}^{n-2} |x_i - \mu| + (n_h - n_l)(\mu - \mu_r).
 \end{aligned} \tag{5.10}$$

Based on (5.10), the MAD τ of $\mathbf{x}$ can be expressed as

$$\begin{aligned}
 \tau &= \frac{1}{n} \left[\sum_{i=1}^{n-2} |x_i - \mu| + (x_b - x_a) \right] \\
 &\geq \frac{1}{n} [(n-2)\tau_r - (n_h - n_l)(\mu - \mu_r) + (x_b - x_a)].
 \end{aligned} \tag{5.11}$$

As $n_h + n_l = n - 2$ and

$$n(\mu - \mu_r) = \sum_{i=1}^{n-2} x_i + x_a + x_b - \frac{n}{n-2} \sum_{i=1}^{n-2} x_i = x_a + x_b - 2\mu_r, \tag{5.12}$$

the second and third terms of (5.11) can be reformulated as

$$\begin{aligned}
 &(x_b - x_a) - (n_h - n_l)(\mu - \mu_r) \\
 &= (x_b - x_a) - (n_h - n_l)(x_a + x_b - 2\mu_r)/n \\
 &= 2(n_l + 1)(x_b - \mu_r)/n + 2(n_h + 1)(\mu_r - x_a)/n.
 \end{aligned} \tag{5.13}$$

Since $\tau_r = \frac{2}{n-2} \sum_{\substack{x_i > \mu_r \\ x_i \in \mathbf{x}_r}} (x_i - \mu_r)$ [7], we have

$$x_b - \mu_r \geq \frac{1}{n_h} \sum_{\substack{x_i > \mu_r \\ x_i \in \mathbf{x}_r}} (x_i - \mu_r) = \frac{n-2}{2n_h} \tau_r. \tag{5.14}$$

Similarly,

$$\mu_r - x_a \geq \frac{1}{n_l} \sum_{\substack{x_i \leq \mu_r \\ x_i \in \mathbf{x}_r}} (\mu_r - x_i) = \frac{n-2}{2n_l} \tau_r. \quad (5.15)$$

Substituting (5.14) and (5.15) into (5.13) yields

$$\begin{aligned} & (x_b - x_a) - (n_h - n_l)(\mu - \mu_r) \\ & \geq \frac{2(n_l + 1)}{n} \frac{(n-2)}{2n_h} \tau_r + \frac{2(n_h + 1)}{n} \frac{(n-2)}{2n_l} \tau_r \\ & = \tau_r \frac{n-2}{n} \left(\frac{(n_h - n_l)^2 + 2n_h n_l + (n_h + n_l)}{n_h n_l} \right) \\ & \geq \tau_r \frac{n-2}{n} \left(2 + \frac{n_h + n_l}{n_h n_l} \right) \geq \tau_r \frac{n-2}{n} \left(2 + \frac{4}{n_h + n_l} \right) \\ & = 2\tau_r. \end{aligned} \quad (5.16)$$

Therefore, substituting (5.16) into (5.11) yields

$$\tau \geq \tau_r. \quad (5.17)$$

Note that $\tau = \tau_r$ occurs if and only if all the inequalities of (5.14), (5.15) and (5.16) are respectively equal, i.e.

$$x_b - \mu_r = \frac{n-2}{2n_h} \tau_r, \quad (5.18)$$

$$\mu_r - x_a = \frac{n-2}{2n_l} \tau_r, \quad (5.19)$$

and

$$(x_b - x_a) - (n_h - n_l)(\mu - \mu_r) = 2\tau_r, \quad (5.20)$$

which require (a) all samples are equal to either x_b or x_a and (b) $n_h = n_l$. This specific case does not occur if $\mu \neq \phi$. This completes the proof of (5.9). ■

Theorem 1 guarantees that trimming a pair of extreme samples (one minimum and one maximum samples) from any finite data set decreases the MAD of the samples from the mean. Increasing the number of trimmed sample pairs further decreases the MAD. This inspires the proposed iterative trimmed and truncated arithmetic mean (ITTM) algorithm shown by Algorithm 4.

Algorithm 4: Procedure of the ITTM Algorithm

Input: $\mathbf{x}_0 \Rightarrow \mathbf{x}$; **Output:** Trimmed and truncated $\mathbf{x}$;

1 **do**

2   Compute the sample mean: $\mu = \text{mean}(\mathbf{x})$;

3   Compute the dynamic threshold: $\tau = \text{mean}(|\mathbf{x} - \mu|)$;

4    $b_l = \mu - \tau$, $b_u = \mu + \tau$, and trim all sample pairs (x_i, x_j) from $\mathbf{x}$ that satisfy $x_i \geq b_u$, $x_j \leq b_l$;

5   Truncate the rest samples by:

$$x_i = \begin{cases} b_u, & \text{if } x_i > b_u \\ b_l, & \text{if } x_i < b_l ; \\ x_i, & \text{otherwise} \end{cases}$$

6 **while** the stopping criterion S is violated ;

Let $n_{\tau u}(k)$ and $n_{\tau l}(k)$ be the number of samples respectively satisfying $x_i \geq b_u$ and $x_j \leq b_l$ in the k^{th} iteration. Obviously, the number of samples trimmed from $\mathbf{x}$ in the k^{th} iteration in Step 4 of Algorithm 4 is $2 \min\{n_{\tau l}(k), n_{\tau u}(k)\}$.

Three types of the ITTM filter outputs are proposed. The type I and II outputs are analogous to those of the ITM filter. Assume the total number of the trimmed samples on each side of the mean be n_t . By padding the trimmed samples with the lower or upper bound, the padded $\mathbf{x}$ is $\mathbf{x}_p = \{n_t \diamond b_l, \mathbf{x}, n_t \diamond b_u\}$, where $\diamond$ is the replication operator defined by

$$n_t \diamond x = \underbrace{x, x, \dots, x}_{n_t \text{ times}}. \quad (5.21)$$

The type I output ITTM1 is

$$y_{tt1} = \text{mean}(\mathbf{x}_p). \quad (5.22)$$

The type II output ITTM2 is

$$y_{tt2} = \begin{cases} \text{mean}(\mathbf{x}_r), & \text{if } n_r > \xi \\ y_{tt1}, & \text{otherwise} \end{cases}, \quad (5.23)$$

where $\mathbf{x}_r = \{x_i | b_l < x_i < b_u\}$, and n_r is the number of samples in $\mathbf{x}_r$. Resembling the ITM filter, the ITTM1 filter keeps all the truncated samples, and the ITTM2 filter trims all the truncated samples. As the dynamic threshold of the ITTM algorithm decreases faster than that of the ITM algorithm, under a same stopping criterion, the ITTM1 and ITTM2 filters will be faster than the ITM1 and ITM2 filters.

Besides these two outputs, a third type of output ITTM3 is proposed based on the noise model (5.1). Keeping all the truncated samples (ITTM1) or trimming all the truncated samples (ITTM2) may not be optimal if both the additive and exclusive noise exist. This motivates us to design the ITTM3 filter which trims the impulsive samples and truncates the long-tailed noise.

The proposed type III output ITTM3 is

$$y_{tt3} = \begin{cases} \text{mean}(\mathbf{x}), & \text{if } n - 2n_t > \xi \\ y_{tt1}, & \text{otherwise} \end{cases}, \quad (5.24)$$

where $n - 2n_t$ is the number of un-trimmed samples remained in $\mathbf{x}$. The switch condition is used to avoid the unreliable mean in case too many samples are trimmed in Algorithm 2. In this chapter, $\xi = n/4$ is chosen, which is the same as that in [7].

The ITTM filter's output moves from the mean towards the median by increasing the number of iterations. Since for many real signals/images, neither mean nor median is the optimal solution, proper stopping criterion is applied to suppress noise and preserve edges within a few

iterations. In order to facilitate the comparison between the ITM and ITTM filters, we use the stopping criterion S proposed in [7]. It is designed for general cases, and includes 4 parts. The first one terminates the iteration if the trimmed and truncated mean is close to the median. It is

$$S_1(\varepsilon_1) : \Delta n \triangleq |n_h - n_l| \leq \varepsilon_1. \quad (5.25)$$

If $S_1(1)$ is met, no sample exists between the truncated mean and the median. The second part S_2 uses a predefined maximum number ε_2 to limit the number of iteration k

$$S_2(\varepsilon_2) : k \geq \varepsilon_2. \quad (5.26)$$

The third part S_3 is to handle edges

$$S_3(\varepsilon_3) : n_\tau \triangleq |n_{\tau u} - n_{\tau l}| \geq \varepsilon_3. \quad (5.27)$$

If choosing $\varepsilon_3 = \sqrt{n}$, an auxiliary condition S_4 is needed to avoid an immature stop

$$S_4 : n_\tau(k) = n_\tau(k-1). \quad (5.28)$$

A sophisticated stopping criterion S chosen in [7] is a combination of the above, that is

$$S = S_1(1) \vee S_2(2\sqrt{n}) \vee S_3[(n - \sqrt{n})/2] \vee [S_3(\sqrt{n}) \wedge S_4]. \quad (5.29)$$

5.3 Properties of the ITTM Filters

Property 1: (Faster convergence) The truncation threshold τ of the ITTM algorithm has a faster convergence than that of the ITM algorithm. It decreases monotonically to zero if the mean μ

deviates from the median ϕ , i.e.

$$\tau(k+1) < \tau(k), \text{ if } \mu \neq \phi, \quad (5.30)$$

and

$$\lim_{k \rightarrow \infty} \tau = 0, \text{ if } \mu \neq \phi. \quad (5.31)$$

Proof: For symbolic simplicity, we omit the index k wherever no ambiguity is caused. Let

$$\mathbf{x}_h = \{x_i | x_i > \mu, x_i \in \mathbf{x}\}, \quad (5.32)$$

and

$$\mathbf{x}_l = \{x_i | x_i \leq \mu, x_i \in \mathbf{x}\}. \quad (5.33)$$

Let n_h , n_l , μ_h and μ_l denote the numbers and means of these two set. Obviously, $n_h + n_l = n - n_t$, where n_t is the number of trimmed samples, $\mathbf{x}_h \cup \mathbf{x}_l = \mathbf{x}$ and

$$\begin{aligned} n_h \mu_h + n_l \mu_l &= (n - n_t) \mu \\ &= n_h \mu + n_l \mu. \end{aligned} \quad (5.34)$$

Let $\delta_h = \mu_h - \mu$ and $\delta_l = \mu - \mu_l$. With some manipulation, (5.34) becomes

$$n_h \delta_h = n_l \delta_l. \quad (5.35)$$

As the mean absolute deviation τ of the trimmed and truncated $\mathbf{x}$ can be represented by

$$\begin{aligned}
 \tau &= \frac{1}{n - n_t} \left[\sum_{x_i > \mu} (x_i - \mu) + \sum_{x_i \leq \mu} (\mu - x_i) \right] \\
 &= \frac{1}{n - n_t} [n_h(\mu_h - \mu) + n_l(\mu - \mu_l)] \\
 &= \frac{1}{n - n_t} (n_h \delta_h + n_l \delta_l),
 \end{aligned} \tag{5.36}$$

substituting (5.35) into (5.36) yields

$$\begin{aligned}
 \tau &= 2 \frac{n_h}{n - n_t} \delta_h \\
 &= 2 \frac{n_l}{n - n_t} \delta_l.
 \end{aligned} \tag{5.37}$$

Therefore, from (5.37) we have

$$\begin{aligned}
 \delta_h + \delta_l &= \frac{1}{2} (n - n_t) \tau(k) \left(\frac{1}{n_h} + \frac{1}{n_l} \right) \\
 &= \frac{(n - n_t)^2}{2n_h n_l} \tau(k).
 \end{aligned} \tag{5.38}$$

Due to the trimming and truncating process of the data in the previous iteration, we have $\delta_h + \delta_l < 2\tau(k - 1)$, if $\mu \neq \phi$. Therefore,

$$\begin{aligned}
 \tau(k) &= \frac{2n_h n_l}{(n - n_t)^2} (\delta_h + \delta_l) \\
 &< \frac{4n_h n_l}{(n - n_t)^2} \tau(k - 1) \\
 &= \frac{(n_h + n_l)^2 - (n_h - n_l)^2}{(n_h + n_l)^2} \tau(k - 1) \\
 &= \left[1 - \left(\frac{\Delta n}{n - n_t} \right)^2 \right] \tau(k - 1).
 \end{aligned} \tag{5.39}$$

If $\mu \neq \phi$, it is easy to find that $1 \leq \Delta n \leq n$. Therefore, from (5.39) we get the following

inequality

$$\begin{aligned}
\tau(k) &< \left[1 - \frac{1}{(n - n_t)^2} \right] \tau(k-1) \\
&\leq \tau(1) \prod_{i=1}^{k-1} \left[1 - \frac{1}{(n - n_t(i))^2} \right] \\
&\leq \tau(1) \left(1 - \frac{1}{n^2} \right)^{k-1}.
\end{aligned} \tag{5.40}$$

From (5.40) it is straightforward to get the inequality (5.31). Theorem 1 has proved that trimming the extreme samples leads to a smaller mean absolute deviation. Therefore, the dynamic threshold of the ITTM algorithm converges to zero faster than that of the ITM algorithm if there are samples trimmed. In fact, the ITTM algorithm can trim samples in most cases only except the specific case in which more than half samples equal to the median and all the rest samples larger (all the rest samples smaller) than the median. In this case the ITTM algorithm degrades to the ITM algorithm. As this case is rare for real data, it is reasonable to claim that the dynamic threshold of the ITTM algorithm has a higher convergence rate than that of the ITM algorithm. ■

Fig. 5.2 shows the average of the dynamic thresholds over 10^6 Laplacian distributed input data sets against the number of iterations. They are normalized by the MAD of the input data sets. The filter size is $n = 49$. This figure demonstrates that the convergence rate of the ITTM algorithm is higher than that of the ITM algorithm.

Property 2: (Converge to the median) The ITTM1, ITTM2 and ITTM3 outputs y_{tt1} , y_{tt2} and y_{tt3} converge to the median ϕ of the samples in the filter window, i.e.

$$\lim_{k \rightarrow \infty} y_{tt1} = \lim_{k \rightarrow \infty} y_{tt2} = \lim_{k \rightarrow \infty} y_{tt3} = \phi. \tag{5.41}$$

Proof: Let $n_{\tau u}$ and $n_{\tau l}$ be the numbers of samples larger than $\mu + \tau$ and smaller than $\mu - \tau$,

Figure 5.2: Average MAD of the samples from the mean against the number of iterations. The input data sets are Laplacian noise. The filter size is $n = 49$.

respectively. For $n_{\tau l} \neq 0$, we have

$$\begin{aligned}
 n_{\tau l} \tau &< \sum_{x_i < \mu - \tau} (\mu - x_i) \\
 &\leq \sum_{x_i < \mu} (\mu - x_i).
 \end{aligned} \tag{5.42}$$

Since

$$\sum_{x_i < \mu} (\mu - x_i) = n_l \delta_l, \tag{5.43}$$

(5.42) becomes

$$n_{\tau l} < \frac{n_l \delta_l}{\tau}. \tag{5.44}$$

Substituting (5.37) into (5.44) yields

$$n_{\tau l} < (n - n_t)/2. \tag{5.45}$$

As $n - n_t$ is the number of samples in $\mathbf{x}$, according to the definition of the median we have

$$\phi \geq \mu - \tau. \quad (5.46)$$

Similarly, we have

$$\phi \leq \mu + \tau. \quad (5.47)$$

(5.46) and (5.47) prove that $\mu - \tau \leq \phi \leq \mu + \tau$. It means that, using τ as the dynamic threshold, ϕ is not changed during the truncating procedure. Moreover, trimming the same number of samples from both sides of the sorted data also does not change ϕ . Therefore, ϕ is not changed in the iterative trimming and truncating procedures. Since Property 1 shows that the dynamic threshold converges to zero, the outputs y_{tt1} , y_{tt2} and y_{tt3} converge to ϕ . ■

Property 3: (Scale and shift invariance) The ITTM filter is invariant to scale and shift, i.e. if $\mathbf{z}_0 = \{\alpha x_i + c\}, \forall x_i, x_i \in \mathbf{x}_0$, we have

$$y_{tt}(\mathbf{z}_0) = \alpha y_{tt}(\mathbf{x}_0) + c, \quad (5.48)$$

where α and c are two constants, and y_{tt} is a general notation of all the three types of the ITTM filter's outputs. The proof is trivial and hence omitted.

Property 4: (Symmetric distribution) The distribution of the ITTM filter's output is symmetric, if the samples of the input data set $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are drawn from the random variable X with a symmetric distribution.

Proof: If x_i is symmetrically distributed around c , $2c - x_i$ has the same distribution as x_i . According to Property 3, $y_{tt}(2c - x_1, 2c - x_2, \dots, 2c - x_n) = 2c - y_{tt}(x_1, x_2, \dots, x_n)$. Thus, the distribution of y_{tt} is symmetric around c . ■

Property 5: (Unbiased estimate) The ITTM filter output is an unbiased estimate of the population mean of X , if the samples in $\mathbf{x}_0 = \{x_1, x_2, \dots, x_n\}$ are drawn from the random

Figure 5.3: Profile outputs of the mean, median, ITTM1, ITTM2 and ITTM3 filters of size $n = 11 \times 11$ after 3 iterations for a step edge.

variable X with a symmetric distribution.

Proof: According to Property 4, y_{tt} is symmetrically distributed around c . Therefore, $E\{y_{tt}\} = E\{X\} = c$. This completes the proof of Property 5. ■

Property 6: (Edge preservation) The ITTM2 filter of size in odd number preserves image step edges with any number of iterations.

Proof: The proof of this property is similar to that in [7]. If the filter window only covers one side of the edge, all the values in the filter window are the same. In this case, the ITTM2 filter doesn't change the pixel values. When the filter window crosses the edge, more pixels on the side in which the window center resides are included than pixels on the other side. Therefore, the pixels on this side will not be changed and those on the other side are truncated from the first iteration. As the ITTM2 output is average of the un-truncated samples, this filter keeps the step edge. ■

A step edge profile is shown in Fig. 5.3. The filter size for all filters is $n = 11 \times 11$. It is seen that the output of the ITTM1 filter is the same as that of the median filter after the first iteration. Although the ITTM1 and ITTM3 filters blur the edge, they produce lighter affect than the mean filter.

Property 7: (Impulsive noise suppression) The ITTM2 filter removes impulse $\mathbb{D}_1$ from the homogeneous area $\mathbb{D}_2$ with any iterations:

$$x_i = \begin{cases} c_1, & \text{for } x_i \in \mathbb{D}_1, n_1 < n/2 \\ c_2, & \text{for } x_i \in \mathbb{D}_2, n_2 > n/2 \end{cases}, \mathbb{D}_1 \cup \mathbb{D}_2 = \mathbf{x}_0, \quad (5.49)$$

where $c_1 \neq c_2$, n_1 and n_2 are the numbers of samples in sets $\mathbb{D}_1$ and $\mathbb{D}_2$, respectively.

Proof: This property is inherited from the ITM2 filter [7]. As $n_1 < n_2$, all samples equalling to c_1 are truncated and all samples equalling to c_2 are within the bounds from the first iteration. Therefore, the ITTM2 output equals to c_2 . ■

5.4 Computational Complexity

The computational complexity depends on how the algorithm is implemented. An implementation of the ITTM filter is proposed based on the following proposition.

Proposition 1: Samples, once being truncated in an iteration of the ITTM algorithm, must be either trimmed or truncated in all subsequent iterations.

Proof. Let $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ be the input data of the k^{th} iteration. For the symbolic simplicity, let $\mathbf{x}_+$, $\mathbf{x}_{h+}$, x_{i+} , μ_+ , τ_+ , n_{h+} and δ_{h+} be the corresponding notations in the $(k+1)^{th}$ iteration.

If the number of samples smaller than the lower bound equals to that larger than the upper bound, all these samples are trimmed in this iteration according to the definition of the ITTM

algorithm. In this case, Proposition 1 is established. For the case that these two numbers are not equal, the same number of extreme samples are trimmed from the input data set, and the remaining asymmetric ones are truncated to the bound. In the following, we will prove that Proposition 1 is true in this case. Assume that m samples are trimmed from each side of the input data, and a sample x_{iu} is truncated to the upper bound $\mu + \tau$ in the k^{th} iteration. Obviously, $x_{iu} = \mu + \tau$, which has the maximum value of the input data set $\mathbf{x}_+$ of the $(k + 1)^{th}$ iteration. As Property 1 proves that the dynamic threshold decreases monotonically, $\tau_+ < \tau$. Therefore, we have

$$x_{i+} > \mu_+ + \tau_+, \text{ if } \mu_+ \leq \mu, \quad (5.50)$$

In case of $\mu_+ > \mu$, from $\tau = 2n_h\delta_h/n$, we have

$$\begin{aligned} \tau &= \frac{2}{n} \sum_{x_i \in \mathbf{x}, x_i > \mu} (x_i - \mu) \geq \frac{2}{n} \sum_{x_i \in \mathbf{x}, x_i > \mu_+} (x_i - \mu_+ + \mu_+ - \mu) \\ &= \frac{2}{n} \sum_{x_i \in \mathbf{x}, x_i > \mu_+} (x_i - \mu_+) + \frac{2(n_{h+} + m)}{n} (\mu_+ - \mu). \end{aligned} \quad (5.51)$$

As there are m samples trimmed from $\mathbf{x}_h$ and at least one sample x_{iu} truncated to the upper bound in the k^{th} iteration, we have

$$\begin{aligned} \frac{2}{n} \sum_{x_i \in \mathbf{x}, x_i > \mu_+} (x_i - \mu_+) &> \frac{2}{n} \sum_{x_{i+} \in \mathbf{x}_+, x_{i+} > \mu_+} (x_{i+} - \mu_+) + \frac{2m}{n} (x_{iu} - \mu_+) \\ &= \frac{n - 2m}{n} \tau_+ + \frac{2m}{n} (\tau - (\mu_+ - \mu)). \end{aligned} \quad (5.52)$$

Substituting (5.52) into (5.51) yields

$$\tau = \frac{n - 2m}{n} \tau_+ + \frac{2m}{n} \tau + \frac{2n_{h+}}{n} (\mu_+ - \mu). \quad (5.53)$$

With some manipulation, (5.53) becomes

$$\tau = \tau_+ + \frac{2n_{h+}}{n - 2m}(\mu_+ - \mu). \quad (5.54)$$

As $\tau_+ = 2n_{h+}\delta_{h+}/(n - 2m)$, (5.54) becomes

$$\tau > \tau_+ + \frac{\tau_+}{\delta_{h+}}(\mu_+ - \mu). \quad (5.55)$$

As $\delta_{h+} \leq x_{iu+} - \mu_+$, we have $\delta_{h+} \leq \tau_+$, if

$$x_{iu+} - \mu_+ \leq \tau_+. \quad (5.56)$$

Therefore, under the condition (5.56), (5.55) becomes

$$\tau > \tau_+ + \mu_+ - \mu. \quad (5.57)$$

Since $x_{iu+} = \mu + \tau$, (5.57) becomes

$$x_{iu+} > \mu_+ + \tau_+. \quad (5.58)$$

The conclusion (5.58) conflicts with (5.56). Hence, the condition (5.56) is not true, which means,

$$x_{iu+} > \mu_+ + \tau_+, \text{ if } \mu_+ > \mu. \quad (5.59)$$

From (5.50) and (5.59), we have

$$x_{iu+} > \mu_+ + \tau_+. \quad (5.60)$$

In the same way, we can prove that if a sample x_{il} is truncated to the lower bound $\mu - \tau$ in the k^{th} iteration,

$$x_{il+} < \mu_+ - \tau_+. \quad (5.61)$$

Inequalities (5.60) and (5.61) guarantee that the truncated samples will be either trimmed or truncated in the following iteration. This completes the proof of Proposition 1. ■

As all truncated samples must be either trimmed or truncated in the subsequent iterations, we do not need access such samples for computing the mean, threshold and checking whether they should be trimmed, truncated or not. We only need count the number of such samples in all subsequent iterations. Let $n_{\tau u}$ and $n_{\tau l}$ be the total numbers of the samples $\{x_i\}$ and $\{x_j\}$ in $\mathbf{x}_0$ satisfying $x_i \geq b_u$ and $x_j \leq b_l$, respectively. The proposed implementation of the ITTM algorithm is shown by Algorithm 5.

Algorithm 5: An Implementation of the ITTM Algorithm

Input: $\mathbf{x}_0 \Rightarrow \mathbf{x}_r, n_t = 0, n_\tau = 0$; **Output:** $\mathbf{x}_r, b_l, b_u, b, n_{\tau l}, n_{\tau u}, n_\tau$ and n_t ;

```

1 do
2    $\mu = (\text{sum}(\mathbf{x}_r) + n_\tau b) / (n - 2n_t);$ 
3    $\tau = (\text{sum}(|\mathbf{x}_r - \mu|) + n_\tau |b - \mu|) / (n - 2n_t);$ 
4    $b_l = \mu - \tau, b_u = \mu + \tau, \mathbf{x}_r = \{x_i | b_l < x_i < b_u\}$  and update  $n_{\tau l}$  and  $n_{\tau u}$ ;
5   Compute  $n_t = \min\{n_{\tau l}, n_{\tau u}\}, n_\tau = |n_{\tau l} - n_{\tau u}|$ , and  $b = b_u$  if  $n_{\tau u} > n_{\tau l}$ , else  $b = b_l$ ;
6 while the stopping criterion  $S$  is violated ;

```

The three types of the ITTM filter outputs are reformulated as

$$y_{tt1} = \frac{\text{sum}(\mathbf{x}_r) + n_{\tau l}b_l + n_{\tau u}b_u}{n}, \quad (5.62)$$

$$y_{tt2} = \begin{cases} \text{mean}(\mathbf{x}_r), & \text{if } n_\tau > \xi \\ y_{tt1}, & \text{otherwise} \end{cases}, \quad (5.63)$$

and

$$y_{tt3} = \begin{cases} \frac{\text{sum}(\mathbf{x}_r) + n_\tau b}{n - 2n_t}, & \text{if } n - 2n_t > \xi \\ y_{tt1}, & \text{otherwise} \end{cases}. \quad (5.64)$$

The results of (5.62), (5.63) and (5.64) are the same as those of (5.22), (5.23) and (5.24), respectively.

The computational complexity of the ITTM filter can be measured by the times that all

Figure 5.4: Average number of iterations determined by the default stopping criterion against the filter size n . The x -axis is in log scale of n .

samples of $\mathbf{x}_0$ are visited in all iterations. It is determined by the number of iterations N_s , and the probability of a sample being visited in the k^{th} iteration.

We use the Monte Carlo simulations [78] to analyze the number of iterations N_s . The stopping criterion is set the same as that in [7] shown in (5.29). Three types of noise, Gaussian, Laplacian and the uniform distributed noise, are simulated. 10^6 independent input data sets are used in each experiment. The experimental results in Fig. 5.4 illustrate that the numbers of iterations of different noise types have the same tendency. They are approximately linear functions of $\ln n$. Therefore, we use

$$\hat{N}_s = 0.8 \ln n \quad (5.65)$$

as an upper bound of N_s , which is plotted in Fig. 5.4.

As Algorithm 5 only accesses the un-trimmed and un-truncated samples, the visited samples in the k^{th} iteration are the ones within the range $(\mu_{k-1} - \tau_{k-1}, \mu_{k-1} + \tau_{k-1})$. As the dynamic threshold decreases monotonically, the probability of a sample within this range decreases. Therefore, the probability of a sample being visited p_k decreases. In order to simplify the analysis of p_k , we employ the uniform distributed noise as an example. The probability density function of a uniform distributed random variable X is given by

$$f_u(X = x) = \begin{cases} 1, & \text{if } -0.5 \leq x \leq 0.5 \\ 0, & \text{otherwise} \end{cases}. \quad (5.66)$$

From (5.66), we find the following lemma of τ_k .

Lemma 1: When the filter size n is sufficiently large, the dynamic threshold τ_k of the trimmed and truncated X drawn from the uniform distribution (5.66) follows

$$\tau_k = \frac{1}{2^{k+1}}, k \geq 1. \quad (5.67)$$

Proof: When the filter size n is sufficiently large, the sample mean equals the expectation of X , i.e. $\mu = E[X] = 0$. The dynamic threshold of the first iteration is

$$\tau_1 = \frac{1}{n} \sum_{i=1}^n |x_i - \mu| = E[|X|] = \frac{1}{2}. \quad (5.68)$$

As the output of the ITTM filter is unbiased, $\mu_{k-1} = 0$. Thus, almost all samples outside the range $(-\tau_{k-1}, \tau_{k-1})$ are trimmed. Therefore, the dynamic threshold of the k^{th} iteration τ_k is

$$\tau_k = \frac{1}{2\tau_{k-1}} \int_{-\tau_{k-1}}^{\tau_{k-1}} |x| dx = \frac{\tau_{k-1}}{2}. \quad (5.69)$$

From (5.68) and (5.69) we get (5.67). This completes the proof of Lemma 1. ■

As only the un-truncated samples are visited by the ITTM algorithm, the probability of a

Figure 5.5: Average visiting times of a sample against the filter size n .

sample being visited at the k^{th} iteration is $p_k = 2\tau_{k-1}$, and we define $\tau_0 = 0.5$. Therefore, the computational complexity of the ITTM algorithm is $\mathcal{O}(\sum_{k=1}^{\hat{N}_s} np_k) = \mathcal{O}(2n \sum_{k=1}^{\hat{N}_s} \tau_{k-1})$. From Lemma 1 we get that $1 \leq 2 \sum_{k=1}^{\hat{N}_s} \tau_{k-1} < 2$. Thus, the computational complexity of the ITTM algorithm is $\mathcal{O}(n)$. It is smaller than the standard median filter and the fast ITM (FITM) filter given in [59], both of which are $\mathcal{O}(n \log n)$.

The Monte Carlo simulations are also carried out to analyze the visiting times when the filter size n is not large enough. 10^6 independent input data sets are used in each experiment. The experimental results in Fig. 5.5 shows that the average visiting times of a sample in the ITTM filter window can be constrained by the upper bound $\hat{N}_{ave} = 2$. Therefore, the average total visiting times of all the samples for the ITTM filter are smaller than $2n$.

We further evaluate the computational complexity of the ITTM filter in two experiments. These experiments are performed under the Window 7 system with the Intel Core i5 CPU

Figure 5.6: Normalized time consumption against the number of iterations k . The time consumption is normalized by that of the median filter. The y -axis is in log scale.

3.2GHz. All filters are implemented by C programming language. As data sorting is the basic building block of many rank order statistic filters, such as the popular αT filter, we implement both the median and αT filters using the quick-sort algorithm. The ITM filter is implemented based on the fast algorithm FITM given in [59]. The MA, OM (optimal Myriad) and MLC (Mean-LogCauchy) filters are implemented according to the algorithms given in [26, 61, 99]. The first experiment tests the running time of the filters against the number of iterations, and the second experiment tests that with the stopping criterion against the filter size. 10^6 independent Laplacian distributed input data sets are used in each experiment. The time consumption is normalized by that of the median filter. The normalized time consumption with the filter size $n = 49$ against the number of iterations is shown in Fig. 5.6. The running time of all the iterative-algorithm based filters, the MA, OM, MLC, FITM and ITTM filters, increases along the number of iterations. As the threshold of the ITTM filter decreases faster than that of the

Figure 5.7: Normalized time consumption against the filter size n . The time consumption is normalized by that of the median filter. The y -axis is in log scale.

ITM filter, its time consumption increases slowly compared to that of the FITM filter. Both the FITM and ITTM filters are faster than the median filter for all the numbers of iterations in Fig. 5.6. The ITTM filter is the fastest one. The computational complexity of the MA, OM and MLC filters is much higher than the median filter. The experimental results using the stopping criterion are plotted in Fig. 5.7. The stopping criterion given by (5.29) for both the FITM and ITTM filters is set the same as that in [7]. The fixed number of iterations 10 is applied for the MA, OM and MLC filters. As the MLC filter is a linear combination of the mean and OM filters, its computational complexity is a little bit higher than that of the OM filter. Similarly, the time consumption of the αT filter is slightly higher than that of the median filter because it averages a part of the data after data sorting. The MA, OM and MLC filters are much slower than the median filter. Both the FITM and ITTM filters are faster than the median filter. The ITTM filter is the fastest one for all filter sizes.

5.5 Experiments

Although better performance can be achieved by tuning the parameters, the proposed filter does not optimize the parameters for specific noise distribution or data set. The same parameters of the ITTM filter are employed throughout all experiments. In order to facilitate the comparison between the ITM and ITTM filters, the parameter setting proposed in [7] is applied to these two filters. Their stopping criterion is fixed as (5.29) and $\xi = n/4$ is fixed for the ITM2, ITTM2 and ITTM3 filters. All additive noise applied in this work has i.i.d. and zero mean. σ_n is used to denote the standard deviation of Gaussian noise. This section reports six sets of experiments.

The filters' noise attenuation capability in a constant signal is tested in the first two sets of experiments, and the filters' overall performance in preserving image structure and attenuating noise is tested in the next two sets. The ITTM filters are compared with the mean, median, α T [22, 23], MEM [25, 27] and ITM [7] filters. The mean absolute error (MAE) over 10^7 independent outputs is used as the performance indicator for synthetic data. The mean square error (MSE) is used for real images. As none of the α -adapted α T filters [28, 106] outperform an α -fixed α T filter averagely over the experiments, the α -fixed α T filter is compared in this section. $\alpha = 0.25$ is chosen as the α T filter approaches the mean if $\alpha \rightarrow 0$ and approaches the median if $\alpha \rightarrow 0.5$.

In the last two sets of experiments, we compare the proposed filters with four iterative-algorithm based filters, including the MA [26], OM [30], MLC [25] and ITM [7] filters. The α -stable noise is tested as the MA, OM and MLC filters were designed to solve the problem of this noise model. A coefficient λ is used in the MLC filter to linearly combine the mean and LogCauchy filters to suppress the ϵ contaminated Gaussian noise [60]. The coefficient λ was suggested to be equal to the prior probability of the Gaussian noise [25, 27]. $\lambda = 0.5$ is chosen for both the MLC and MEM filters in this work.

Figure 5.8: MAE against the filter size n in suppressing (a) Gaussian and (b) Laplacian noise. The MAE is normalized by that of the median filter. The average iteration numbers of the ITM filter are 1.5, 3.3, 5.3, 7.5 in (a) and 1.8, 3.6, 5.7, 7.7 in (b) for the filter size from 9 to 81, respectively. The average iteration numbers of the ITTM filter are 1.4, 2.1, 2.6, 3.1 in (a) and 1.5, 2.3, 2.9, 3.4 in (b).

5.5.1 Single Type of Noise in Constant Signal

Filters' performance in suppressing single type of noise on a constant signal is tested. Experimental results in suppressing Gaussian and Laplacian noise are shown in Fig. 5.8(a) and (b), respectively. The filters' MAE is normalized by that of the median filter.

As the mean filter is the optimal one for Gaussian noise, the ITM, ITTM, MEM and αT filters perform between the mean and median filters by making a compromise between them. The median filter is not the minimum MSE estimator [59] though it is the maximum likelihood estimator for the Laplacian distribution. Therefore, it is not a surprise that the ITM1 filter and the proposed ITTM1 and ITTM3 filters outperform the median filter even for the long-tailed Laplacian noise. The αT filter is better than the median filter for $n = 9$ and surprisingly much worse than the median filter for larger filter size. It is seen that the numbers of iterations of the

Figure 5.9: MAE against the filter size n in suppressing (a) Laplacian ϵ -contaminated Gaussian noise, and (b) Laplacian and impulsive ϵ -contaminated Gaussian noise. The MAE is normalized by that of the median filter. The average iteration numbers of the ITM filter are 1.6, 3.4, 5.4, 7.5 in (a) and 1.6, 3.4, 5.5, 7.6 in (b) for the filter size from 9 to 81, respectively. The average iteration numbers of the ITTM filter are 1.4, 2.2, 2.8, 3.2 in (a) and 1.4, 2.2, 2.8, 3.2 in (b).

ITTM filters are smaller than those of the ITM filters for both Gaussian and Laplacian noise.

5.5.2 Mixed Types of Noise in Constant Signal

The mixed noise is generated by the noise model (5.1). It contains two types of noise, the additive noise with probability p and the exclusive noise with probability $1 - p$. The additive noise is set to have the ϵ -contaminated normal distribution as $\mathcal{P}_\epsilon = \{(1 - \epsilon)\Phi + \epsilon H\}$ [60], where Φ and H are Gaussian and a longer-tailed distributions, respectively, $\epsilon \in [0, 1]$. Similar to the setting in [7], we choose the Laplacian distribution as H with the variance $1.3\sigma_n$ and $\epsilon = 0.5$. The distribution $I(x) = 0.5\delta(x - 6\sigma_n) + 0.5\delta(x + 6\sigma_n)$ is applied as the exclusive noise.

We first set $p = 1$ so that only the additive noise exists. Results in Fig. 5.9(a) show that all

the ITM and ITTM filters have better performances than both the mean and median filters. Then, we decrease the probability of the additive noise to $p = 0.9$. The probability of the exclusive noise increases to $1 - p = 0.1$. The experimental results are shown in Fig. 5.9(b). Here it is evidenced that the proposed ITTM3 filter that trims and truncates the samples performs the best, outperforming filters ITM1 and ITTM1 that only truncate the extreme samples and outperforming filters ITM2, ITTM2 and αT filters that only trim extreme samples. Note that results of the mean filters are not shown in Fig. 5.8(b) and Fig. 5.9(b) as they are much worse than all other filters.

5.5.3 Noise Step Edge

A vertical or horizontal step edge with the grey level -1 on one edge side and 1 on the other side is tested. Such an edge is corrupted by Gaussian noise of different levels. The MAE is computed from the outputs of the filter windows which cover both sides of the edge. Experimental results are shown in Fig. 5.10. Filters' MAE is normalized by the MAD of the noise.

The normalized MAE of the median filter over different noise levels is almost a constant. This confirms the excellent ability of the median filter in edge preservation. The mean, MEM, αT , ITM1, ITTM1 and ITTM3 filters all blur the edge. Therefore, when the noise level is low, their performances are poorer than the median filter. Since they suppress Gaussian noise more effectively than the median filter, their MAEs are about the same as that of the median filter when σ_n reaches 0.64. The performance of the ITTM2 filter is similar to that of the ITM2 filter. Both of them significantly outperform the others for the low and medium noise levels. Their performances approach those of the median at the two highest noise levels in which the pixel gray levels of the two edge sides are overlapped.

Figure 5.10: MAE normalized by that of the median filter for the filter size 3×3 in (a) and 11×11 in (b) against the noise level σ_n . The average iteration numbers of the ITM filter are 1, 1, 1, 1, 2.7, 3.7, 2 in (a) and 1.8, 3.8, 4.7, 11.6, 13.5, 14.7, 12.2 in (b). The average numbers of ITTM iterations are 1, 1, 1, 1, 2.3, 2.8, 1.6 in (a) and 1.8, 2.6, 3.0, 6.7, 6.6, 5.2, 4.1 in (b).

(a) Crowd

(b) Bank

(c) Girl

Figure 5.11: Real images in testing: Crowd, Bank, Girl of size 512×512 .

Figure 5.12: Average MSEs over 10 runs for the 3 images at 5 different noise levels of (a) Crowd, (b) Bank and (c) Girl. The average number of ITM iterations is closely around 3.4. The average number of ITTM iterations is closely around 2.1.

5.5.4 Real Images

Images (Crowd, Bank and Girl) shown in Fig. 5.11 are used to test the filters. These images represent different complexity levels and image types. The gray level of these images ranges from 0 to 255 and their size is 512×512 . The filter size is $n = 5 \times 5$.

Additive Gaussian noise of different levels σ_n is used to contaminate the images. As the median filter can effectively keep the image structures but cannot effectively suppress Gaussian noise, its performance is the best for small σ_n but worst for large σ_n among the 8 filters. Therefore, 5 different noise levels are selected for each image so that the median filter performs best at level $\sigma_n(1)$ and worst at level $\sigma_n(5)$ [7]. The other 3 noise levels are chosen by $\sigma_n(5)/\sigma_n(4) = \sigma_n(4)/\sigma_n(3) = \sigma_n(3)/\sigma_n(2) = \sigma_n(2)/\sigma_n(1)$. All MSEs are normalized by that of the median filter. Average MSEs over 10 runs for image Crowd, Bank and Girl are plotted in Fig. 5.12(a), (b) and (c), respectively. The proposed ITTM3 filter performs best except for the

lowest noise level where the median filter is the best. As the abrupt structures of real images that fall in the filter window can be considered as impulsive noise or long-tailed noise, the noise of the real images should be modeled by (5.1) with $1 - p > 0$ even if only additive Gaussian noise is added. This result further confirms that the ITTM3 filter has good performance in attenuating the noise mixed by the short-, long-tailed and impulsive noise. The average number of iterations for the ITTM filters is 2.1. It is smaller than that of the ITM filters which is 3.4.

5.5.5 α -Stable Noise Corrupted Homogeneous Images

We compare the ITTM filters with the iterative-algorithm based filters in this section. The homogeneous images corrupted by the α -stable noise are used to test the filters. The heaviness of the noise tails (degree of impulsiveness) is controlled by adjusting the parameter α , ($0 < \alpha \leq 2$). The noise impulsiveness increases as α decreases [25]. The OM filter [30] employs the “linearity parameter” k , where $k = \sqrt{\alpha/(2 - \alpha)}\gamma^{1/\alpha}$ and γ is the dispersion of the α -stable noise, to effectively suppress this noise with different α and γ . The MLC and OM filters are implemented based on the algorithm given in [61, 99]. For the MA, MLC and OM filters, the fixed number of iterations 20 is applied as more iterations cannot increase the performance visibly [7]. The size for all filters is set to 5×5 .

Table 5.1: MSEs for Homogeneous Image corrupted by α -Stable Noise.

α	0.5	0.8	1.2	1.5	1.8
mean	486.41	217.44	73.050	32.235	14.175
MA	215.84	62.108	18.550	11.315	9.2205
OM	28.796	8.2874	10.617	12.479	11.642
MLC	164.71	62.017	24.940	15.216	10.555
ITM1	26.924	13.343	11.409	10.631	10.074
ITM2	15.050	12.482	12.320	12.186	11.559
ITTM1	18.827	11.827	11.128	10.779	10.516
ITTM2	13.503	11.547	11.759	11.925	11.679
ITTM3	17.536	11.252	10.796	10.678	10.698

Five different α values are set for the α -stable noise with $\gamma = 10$. For each α , 10^7 independent input data sets are generated to get the MSEs. Table 5.1 records the MSEs of various filters. The noise impulsiveness is illustrated by the results of the mean filter. For each α value, the smallest MSE among all filters is in bold font and underlined, and the second smallest is in bold font. The OM filter is the maximum likelihood estimation of the samples with the α -stable distribution ($\alpha = 1$). With the help of the noise distribution information α and k , the OM filter achieves the best performance for α value around 1. Among the filters which do not require the prior knowledge of the noise distribution, including the ITM, ITTM and MA filters, the ITTM3 filter outperforms others for α value around 1. The MA filter achieves the best performance for α value approaching 2, where the α -stable noise degenerates to Gaussian noise. For high degree of impulsiveness ($\alpha = 0.5$) the ITTM2 and ITM2 filters perform best. The average iteration numbers of the ITTM filter are 4.2, 3.3, 3.1, 2.9 and 2.8, respectively for the α values from 0.5 to 1.8. The corresponding average iteration numbers of the ITM filter are 6.6, 5.1, 4.7, 4.5 and 4.3. It is seen that the numbers of iterations of the ITTM filters are smaller than those of the ITM filters for all α values.

5.5.6 Gaussian and α -Stable Noise Corrupted Real Image

The original image “Lena” shown in Fig. 5.13 is corrupted by ϵ -contaminated [60] ($\epsilon = 0.5$) Gaussian ($\sigma_n^2 = 100$) and α -stable ($\gamma = 10$) noise. The image size is 512×512 . The settings of the filters are the same as those in Section 5.5.5. Fig. 5.14 shows the average MSEs of filters over 10 different noised versions of the image “Lena”. All MSEs are normalized by that of the median filter. It demonstrates that the ITTM3 filter achieves the best performance for all the 5 values of α . The performance of the ITTM1 filter is the second best in dealing with this real image. The average numbers of ITTM iterations are 3.1, 2.7, 2.5, 2.4 and 2.3, respectively for the α values from 0.5 to 1.8. They are smaller than the corresponding average numbers of ITM iterations, which are 4.5, 4.0, 3.8, 3.7 and 3.6.

Figure 5.13: Real image “Lena” of size 512×512 tested for the mixed α -stable noise.

Figure 5.14: Normalized MSEs for real image “Lena” corrupted by ϵ -contaminated ($\epsilon = 0.5$) Gaussian ($\sigma_n^2 = 100$) and α -stable ($\gamma = 10$) Noise.

5.6 Summary

The proposed iterative trimmed and truncated arithmetic mean (ITTM) filters circumvent the data sorting process and guarantee the outputs approaching the median with the increasing number of iterations. It is shown in the experiments that the proposed ITTM filters with the rule proposed in [7] that automatically stops the iterations own some merits of both the mean and median filters, and outperform these two fundamental filters in many de-noising applications. By simultaneously trimming and truncating the extreme samples, the ITTM algorithm has a higher convergence rate than the ITM algorithm. The resulting ITTM filters are hence faster than the ITM filters. The computational complexity of the ITTM filters is of $\mathcal{O}(n)$. It is smaller than that of the median and ITM filters, both of which are of $\mathcal{O}(n \log n)$ [59]. Although the ITTM filters use an iterative algorithm, only a few iterations are needed in all the experiments to achieve a good de-noising performance. The number of iterations of the ITTM filter is smaller than that of the ITM filter in all the experiments of this work.

Three types of the ITTM filter outputs, ITTM1, ITTM2 and ITTM3, are proposed. By averaging the truncated input samples, the ITTM1 filter has the best performance in attenuating the short- and long-tailed additive noise among these three filters. By trimming all the truncated samples, the ITTM2 filter has the best performance in suppressing exclusive noise. Simultaneously trimming and truncating the extreme samples leads the ITTM3 filter the best one in attenuating the noise mixed by both the additive and exclusive noise. As the gray value abrupt change of image structure in a filter window can be considered as exclusive or impulsive noise, the ITTM3 filter achieves the best performance in dealing with the real images of this chapter. The superiority and flexibility of the proposed filters are demonstrated by the comprehensive simulation results with the same parameter setting throughout all experiments.

Chapter 6

Rank Order Laplacian of Gaussian Filter and ROLG Interest Point Detector

In this chapter¹, we propose a rank order filter named rank order Laplacian of Gaussian (ROLG) filter. Based on this filter, we develop the ROLG interest point detector. The ROLG filter is designed to detect the image local structures where a significant majority of pixels are brighter or darker than a significant majority of pixels in their corresponding surroundings. Compared to the linear filter based detectors, e.g. SIFT detector, the proposed rank order filter based detector is more robust to abrupt variations of images caused by illumination and geometric changes. Experiments on the benchmark databases demonstrate that the proposed ROLG detector achieves superior performance comparing to four state-of-the-art detectors. Evaluation experiments are also conducted on face recognition problems. The results on five face databases further demonstrate that the ROLG detector significantly outperforms the other detectors.

6.1 Introduction

As the most popular detector, the SIFT detector [48] employs the DoG filter to generate the blob map. However, the response of the DoG filter is easily affected by the strong and abrupt

---

¹This work is partially published in ICASSP2012 (conference paper [1] in author's publication list) and partially published in Pattern Recognition (journal paper [4] in author's publication list).

structures near the structure to be detected. This makes the SIFT detector unstable to detect the low contrast image structures, where other strong and abrupt structures caused by illumination or geometric changes partially fall in the detection window. Moreover, due to the second order derivative nature of the SIFT detector, many unstable and spurious points are often detected around the structures. To solve these problems of the SIFT detector, we design our detector based on the rank order filter.

Most state-of-the-art rank order filters, such as the weighted rank order filter and the weighted median filter, are designed to remove noise or detect edges [9, 13, 100]. Very few studies have been carried out on applying rank order filters to interest point detection. Paler et al. [69] developed a corner detector based on the median filter. The difference between the median filtered image I_m and the original image I , $I_d = I_m - I$, is used as the corner map. As it functions as a high-pass filter, this detector is sensitive to noise. Ren and Jiang [107] employed a rank order filter pair to detect human eyes. The difference of the outputs of the two rank order filters, one for the eyeball and the other for the surrounding pixels, are computed. The local maxima are used as the eye candidates. This eye detector is effective in dealing with iris reflections and other dark objects near eyeballs. However, it cannot be used as an interest point detector because it has the following three limitations. Firstly, it can only detect dark regions. Secondly, it has high response on edges and contours. Local extrema may be detected along edges and contours. These extrema are unstable because they are sensitive to small intensity changes in their neighborhoods. Thirdly, it is not invariant to scale changes.

In this work, a novel rank order filter with weights proportional to the coefficients of the LoG filter is proposed and, hence, is named rank order Laplacian of Gaussian (ROLG) filter. It is used to detect the image local structures where a majority of pixels are brighter or darker than a majority of pixels in their corresponding surroundings. The new interest point detector is built on the proposed ROLG filter to detect image local structures in multiple scales.

Figure 6.1: LoG filter. (a) Shape of the LoG filter. (b) Two parts of the LoG filter. S_1 corresponds to the surrounding ring containing positive weights. S_2 corresponds to the inner circular disk containing all negative weights.

6.2 The Proposed ROLG Filter

As a necessary preliminary of the study, the properties of the LoG filter and the weighted rank order filter are discussed in Section 6.2.1 and 6.2.2.

6.2.1 LoG Filter

The LoG filter with the shape shown in Fig. 6.1(a) is defined by

$$w(x, y, \sigma) = -\frac{1}{\pi\sigma^4} \left[1 - \frac{x^2 + y^2}{2\sigma^2} \right] e^{-\frac{x^2 + y^2}{2\sigma^2}}, \quad (6.1)$$

where σ is the standard deviation of the Gaussian function, and also named scale factor. For input image $I(x, y)$, the output of the LoG filter at (x, y) can be expressed as the difference of

two weighted averages:

$$\begin{aligned}
 r(x, y, \sigma) &= \sum_{(m,n) \in S} w(m, n, \sigma) I(x - m, y - n) \\
 &= \sum_{(m,n) \in S_1} w_+(m, n, \sigma) I(x - m, y - n) - \\
 &\quad \sum_{(m,n) \in S_2} w_-(m, n, \sigma) I(x - m, y - n), \tag{6.2}
 \end{aligned}$$

where S is the region of the filter, S_1 and S_2 (as shown in Fig. 6.1(b)) are the two parts of S with $S_1 \cup S_2 = S$ and $S_1 \cap S_2 = \phi$ (ϕ is the null set). S_1 corresponds to the surrounding ring containing the positive weights of the filter, and S_2 corresponds to its inner disk containing all the negative weights. w_+ and w_- are the absolute values of the LoG coefficients in S_1 and S_2 , respectively.

It is not difficult to see that the LoG filter produces extrema at corners and blobs. Several detectors are built on the LoG filter [44, 47–49]. The SIFT detector [48], which is the most famous one, employs the DoG filter to approximate the normalized LoG filter that significantly accelerates the computation process.

The LoG filter is ineffective in dealing with the sparse but strong noise, such as the salt & pepper noise. Even a small portion of pixels can greatly affect the output adversely if their grey values largely deviate from those of the image structure to be detected. However, a small portion of pixels have almost no influence on the output of the rank order filter even if their grey values are extremely high or low. This motivates us to design a weighted rank order filter with weights proportional to those of the LoG filter for interest point detection.

6.2.2 Weighted Rank Order Filter

The output of the weighted rank order filter [9, 13] is defined as follows. Assume the weights for the input series $\mathbf{x} = \{x_1, x_2, \dots, x_q\}$ are $\mathbf{w} = \{w_1, w_2, \dots, w_q\}$. For the ascending sorted

$\tilde{\mathbf{x}} = \{\tilde{x}_1, \tilde{x}_2, \dots, \tilde{x}_q\}$, their corresponding weights are rearranged as $\tilde{\mathbf{w}} = \{\tilde{w}_1, \tilde{w}_2, \dots, \tilde{w}_q\}$. The output of the weighted rank order filter with rank r_w , $r_w \in \{1, 2, \dots, \sum_{j=1}^{j=q} w_j\}$, is given by

$$\begin{aligned} y_{r_w} &= \text{rank}_{r_w}\{w_1 \diamond x_1, w_2 \diamond x_2, \dots, w_q \diamond x_q\} \\ &= \text{rank}_{r_w}\{\tilde{w}_1 \diamond \tilde{x}_1, \tilde{w}_2 \diamond \tilde{x}_2, \dots, \tilde{w}_q \diamond \tilde{x}_q\}, \end{aligned} \quad (6.3)$$

where $\diamond$ is the replication operator defined by

$$w_i \diamond x = \underbrace{x, x, \dots, x}_{w_i \text{ times}}. \quad (6.4)$$

Take the input series $\mathbf{x} = \{10, 8, 9\}$, the weights $\mathbf{w} = \{2, 3, 1\}$, and the rank $r_w = 4$ as an example. The output of the weighted rank order filter is

$$\begin{aligned} y_4 &= \text{rank}_4\{2 \diamond 10, 3 \diamond 8, 1 \diamond 9\} \\ &= \text{rank}_4\{3 \diamond 8, 1 \diamond 9, 2 \diamond 10\} \\ &= \text{rank}_4\{8, 8, 8, 9, 10, 10\} \\ &= 9, \end{aligned}$$

which is the 4th element of the expanded data.

In order to avoid replicating the data, which is time consuming and needs more storage spaces, a cumulative sum of the sorted weights is defined by

$$c_i = \frac{1}{w_s} \sum_{j=1}^i \tilde{w}_j, \quad (6.5)$$

where $w_s = \sum_{j=1}^q \tilde{w}_j$ is the total sum of the weights, $i \in \{1, 2, \dots, q\}$ and $c_0 = 0$. Then, the

output of the weighted rank order filter with a normalized rank $r_{nw} \in [0 \ 1]$ is given by

$$y_{r_{nw}} = \tilde{x}_{i_o}, \{i_o : c_{i_o-1} < r_{nw} \leq c_{i_o}\}. \quad (6.6)$$

6.2.3 The Proposed ROLG Filter

One direct way to apply the weighted rank order filter to interest point detection is to replace the weighted average in (6.2) by the weighted median, as

$$\begin{aligned} r_{wm}(x, y, \sigma) = & \underset{(m,n) \in S_1}{median}(\hat{w}_+(m, n, \sigma) \diamond I(x - m, y - n)) - \\ & \underset{(m,n) \in S_2}{median}(\hat{w}_-(m, n, \sigma) \diamond I(x - m, y - n)), \end{aligned} \quad (6.7)$$

where $\hat{w}_+(m, n, \sigma) = w_+(m, n, \sigma) / \sum w_+$ and $\hat{w}_-(m, n, \sigma) = w_-(m, n, \sigma) / \sum w_-$. With these weighting coefficients, pixels near the boundary between S_1 and S_2 , which are uncertain to be grouped to S_1 or S_2 , are assigned with small weights to weaken their influence on the filter output. The difference of the two weighted median filters (6.7) has similar role to the LoG filter (6.2) and, hence, can be used to detect interest points.

However, when noise exists, filter (6.7) produces very strong response on an edge if one median filter captures one side of the edge while the other median filter happens to capture the other side of the edge. Such strong response results in many local extrema being detected along edges, which are undesirable for interest point detection. Therefore, additional rules are imposed to enhance the robustness of the detector. Median filter is a special case of the rank order filter, as median is equal to rank 0.5. Replacing the weighted median filter by the weighted rank order filter, (6.7) is reformulated as

$$\begin{aligned} r_{wr}(x, y, \sigma, \lambda_1, \lambda_2) = & \underset{(m,n) \in S_1}{rank_{\lambda_1}}(\hat{w}_+(m, n, \sigma) \diamond I(x - m, y - n)) - \\ & \underset{(m,n) \in S_2}{rank_{\lambda_2}}(\hat{w}_-(m, n, \sigma) \diamond I(x - m, y - n)), \end{aligned} \quad (6.8)$$

where λ_1 and λ_2 are two different rank factors for the two weighted rank order filters.

In order to suppress the edge response, we require a significant majority of pixels ($> 50\%$, e.g. 60%) in the surrounding ring brighter than a significant majority of pixels ($> 50\%$, e.g. 60%) in the inner disk, or a significant majority of pixels in the surrounding ring darker than a significant majority of pixels in the inner disk. Otherwise, the outputs are set to zero to suppress noise and edges. This idea can be realized by introducing a positive nonzero offset parameter δ and two functions, $\mathcal{P}$ and $\mathcal{N}$, as

$$\mathcal{P}(x, y, \sigma, \delta) = r_{wr}(x, y, \sigma, 0.5 - \delta, 0.5 + \delta), \quad (6.9)$$

and

$$\mathcal{N}(x, y, \sigma, \delta) = r_{wr}(x, y, \sigma, 0.5 + \delta, 0.5 - \delta). \quad (6.10)$$

$\mathcal{P}(x, y, \sigma, \delta) > 0$ ($\mathcal{N}(x, y, \sigma, \delta) < 0$) is used to check that a significant majority of pixels in the surrounding ring are brighter (darker) than a significant majority of pixels in the inner circle at point (x, y) . With this idea, the proposed ROLG filter is defined by

$$r_{ROLG}(x, y, \sigma, \delta) = \begin{cases} \mathcal{P}(x, y, \sigma, \delta), & \text{if } \mathcal{P}(x, y, \sigma, \delta) > 0 \\ \mathcal{N}(x, y, \sigma, \delta), & \text{if } \mathcal{N}(x, y, \sigma, \delta) < 0 \\ 0. & \text{otherwise} \end{cases} \quad (6.11)$$

6.3 Analyses of the ROLG Filter

In this section, illustrative analyses of the ROLG response on blobs, corners and edges are presented.

6.3.1 Responses of the ROLG Filter on Blobs

One drawback of the LoG filter in detecting blobs is that spurious local extrema are produced around the blobs. Examples are shown in Fig. 6.2(b) and Fig. 6.3(b). For a 1D blob, one negative peak at the center and two positive peaks on both sides of the center are generated by the LoG filter. For a 2D blob, besides a peak at the center of the blob, a ring (indicated by the red circle shown in Fig. 6.3(b)) is produced around it. Many unstable extrema may be detected on this ring in the presence of even very small noise.

Fig. 6.2(c) and Fig. 6.3(c) show the responses of the proposed ROLG filter on a 1D blob and a 2D blob, respectively. It is clear to see that the ROLG filter produces a peak at the center of the blob, and does not generate any peaks around the blob. This is explained by the filtering process on a 1D blob illustrated in Fig. 6.4. When the 1D-ROLG-filter mask is on one side of the blob, input values within the mask are monotonically increasing or decreasing. Take mask 1 in Fig. 6.4 as an example. Within this mask, input values are monotonically increasing. Compared to its inner part, half of its surrounding parts (left part) is darker, and the other half (right part) is brighter. The ROLG response is set to 0, because it does not satisfy that a significant majority of pixels in the surrounding parts are brighter or darker than a significant majority of pixels in the inner part. When the ROLG mask is at the center of the blob, e.g. mask 2 in Fig. 6.4, a major part of the inner region is brighter than a major part of its surrounding regions. Therefore, a

Figure 6.2: Response on a 1D blob. (a) A 1D blob. (b) Response of the LoG filter. (c) Response of the ROLG filter.

Figure 6.3: Response on a 2D blob. (a) A 2D blob. (b) Absolute value of the LoG response. (c) Absolute value of the ROLG response.

Figure 6.4: A 1D blob and two 1D-ROLG-filter masks on the blob. In each mask, the white region corresponds to the inner part of the ROLG filter mask, and the gray regions correspond to its surrounding parts.

peak is generated by the ROLG filter at the center of the blob. Similar to the 1D case, the ROLG filter produces a peak at the center of the 2D blob, and does not generate a ring around the blob. This avoids detecting spurious points around the 2D blob.

6.3.2 Responses of the ROLG Filter on Edges and Corners

Another problem of the LoG filter in detecting interest points is that local extrema are often detected along edges. For a 1D edge, two peaks near the edge are produced by the LoG filter as shown in Fig. 6.5(b). For a 2D corner, besides a peak on the corner, strong responses are generated along the edges as shown in Fig. 6.6(b). Local extrema may be detected along edges.

The ROLG responses on a 1D edge and a 2D corner are shown in Fig. 6.5(c) and Fig. 6.6(c), respectively. These two figures demonstrate that the ROLG filter suppresses the responses of edges. As shown in Fig. 6.7, when the 1D-ROLG-filter mask is on the edge, values within the mask are monotonically increasing. Half of the surrounding parts is brighter than the inner part, and the other half is darker. No majority of pixels in the inner part are brighter or darker than a majority of pixels in the surrounding parts. Thus, the responses of the ROLG filter on edges are set to 0.

6.4 Interest Point Detection by the ROLG Filter

In this section, we propose the ROLG interest point detector based on the ROLG filter. The advantages of the ROLG filter in detecting interest points in a single scale are discussed in Section 6.4.1. The algorithm to eliminate ridge responses is presented in Section 6.4.2. The proposed ROLG detector to detect interest points in multiple scales is given in Section 6.4.3.

6.4.1 Interest Point Detection in a Single Scale

Previous sections have shown that the ROLG filter has the following advantages in detecting interest points:

Figure 6.5: Response on a 1D edge. (a) A 1D edge. (b) Response of the LoG filter. (c) Response of the ROLG filter.

Figure 6.6: Response on a corner. (a) A 2D corner. (b) Absolute value of the LoG response. (c) Absolute value of the ROLG response.

Figure 6.7: A 1D edge and a 1D-ROLG-filter mask on the edge. In this mask, the white region corresponds to the inner part of the ROLG filter mask, and the gray regions correspond to its surrounding parts.

1. The sparse but strong structures has small or no influence on the output of the ROLG filter. Thus, the ROLG filter is robust to the strong and abrupt variations of images.
2. Structures, which partially fall in a detection window, have limited influence on the response of the ROLG filter. Hence, the mutual influence of structures on the response of the ROLG filter is limited.
3. Only one peak is produced at the center of a blob, and no ring is generated around the blob. This property of the ROLG filter avoids detecting spurious points around a blob.
4. The ROLG filter suppresses the response of edges. Therefore, no point is detected along edges.

Figure 6.8: Points detected on a single scale. ‘*’ denotes desired point and ‘+’ denotes undesired point. From left to right of (a), (b), (c) and (d) are input images, absolute value of the LoG responses, local extrema of the LoG responses, absolute value of the ROLG responses, and local extrema of the ROLG responses, respectively.

Fig. 6.8 shows some image structures and the detected points based on the LoG filter and the proposed ROLG filter. The input image of Fig. 6.8(a) contains a blob and a small black stripe. The abrupt structure drastically changes the LoG response to the blob. The peak of the LoG response deviates from the true position of the blob. Many false peaks are detected around the blob. The response of the ROLG filter shows that the abrupt structure has small impact on

the output of the ROLG filter. The blob is correctly detected by the ROLG filter. Two close blobs are contained in the input image of Fig. 6.8(b). The mutual influence of these two blobs results in many spurious points being detected on the LoG response. However, their mutual influence on the ROLG response does not generate spurious peaks. Thus, these two blobs are detected correctly. Fig. 6.8(c) compares the response of the ROLG filter to that of the LoG filter on a blob. It is clear to see that many false peaks are detected on the LoG response around the blob, while no false peak is detected on the response of the ROLG filter. Fig. 6.8(d) shows that many local intensity extrema are detected along the edges on the LoG response. In contrast, the response of the ROLG filter on the edges is 0 and, hence, no false peak is detected on the edges.

6.4.2 Eliminating Ridge Responses

The response of the ROLG filter on a ridge is strong if its scale is close to the width of the ridge. Points detected on the ridge are unstable to small amounts of noise. We employ the algorithm given in [48] to remove such kind of unstable points.

Points on the ridge have a small principal curvature along the ridge but a large one in the perpendicular direction. The two principal curvatures at the location and scale of the interest point can be computed from the 2×2 Hessian matrix $\mathbf{H}$:

$$\mathbf{H} = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{bmatrix}, \quad (6.12)$$

where I_{xx} , I_{xy} , and I_{yy} are the second order derivatives. The derivatives are estimated by the differences between neighboring sample points.

The principal curvatures of I are proportional to the eigenvalues of $\mathbf{H}$. Therefore, the ratio between the larger eigenvalue and the smaller eigenvalue of $\mathbf{H}$ can be used to remove the points on the ridge. If the ratio is larger than some threshold r , it means that the principal curvatures

in one direction is larger than r times of that in the perpendicular direction. In order to avoid explicitly computing the eigenvalues, the function given in [48]

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r} \quad (6.13)$$

is used to check that the ratio of the eigenvalues of $\mathbf{H}$ is below r . The experiments in this chapter use $r = 10$, as suggested in [48].

6.4.3 Algorithm for the ROLG Detector

Interest point detection in multiple scales is an important issue in vision applications. Similar to the Hessian-Laplace detector [74], we can weight the responses in different scales, and choose the local maximum points both in spatial and scale dimensions as the interest points. In the implementation, we employ a straightforward method by detecting the interest points in each scale as done in [51].

The proposed algorithm for the ROLG detector is summarized below:

1. Initialize the ROLG filter by setting the offset parameter δ and the scale parameter σ .
2. Generate the corner/blob map by filtering the input image with the ROLG filter (6.11).
3. Detect peaks on the corner/blob map, and remove peaks which are on ridges. Remaining peaks are the interest points in this scale.
4. Update the ROLG filter by a larger scale σ , and go back to step 2 to detect interest points in a new scale until the maximal scale is reached.

Fig. 6.9 gives a visual comparison between the SIFT detector and the ROLG detector on a face image. The parameters are chosen so that the numbers of points detected by both the detectors are the same. It is observed that SIFT detects many spurious points around the eyeballs

Figure 6.9: Interest points detected by two detectors. (a) SIFT detector, (b) ROLG detector. Local maxima are indicated in blue, and local minima are indicated in red. The radius of the yellow circle is 2 times the scale σ of the interest point at its center.

while it misses a lot of points in other areas. The ROLG detector alleviates these problems. In the next section, statistical experiments are carried out for further comparison between the ROLG detector and other state-of-the-art ones.

6.5 Experiments

Experiments in Section 6.5.1 test the repeatability and the discrimination of the interest points. Experiments in Section 6.5.2 test the performance of the interest points in the application of face recognition. In both experiments, the SIFT descriptor [48] is used to represent the detected regions.

The proposed ROLG filter has an offset parameter δ , $0 \leq \delta \leq 0.5$. The Oxford database [31] is used to evaluate the sensitivity of the offset parameter δ with respect to the detection result. Both the absolute measures and the relative measures given in Section 6.5.1 are employed. The

experimental results are shown in Fig. 6.10. It is not a surprise that both the number of the repeated points and the number of the matched points decrease with the increase of the offset parameter δ , as shown in Fig. 6.10(a). However, as shown in Fig. 6.10(b), the repeatability and the matching score increase first and then decrease. Both of them reach and maintain the maximum within the range of $0.05 \leq \delta \leq 0.15$. This clearly shows the noise- and edge suppression function of a nonzero offset parameter δ in the proposed ROLG filter. Therefore, we choose $\delta = 0.1$ for the proposed ROLG filter in all experiments of this chapter.

6.5.1 Repeatability and Discrimination Tests

The goal here is to evaluate the ROLG detector under different image variations. The evaluation is based on the protocols suggested in [31]. Detectors are evaluated by both the absolute measures and the relative measures. The absolute measures include the number of repeated interest points and the number of matched points. Each interest point corresponds to a detected region. Two regions are repeated if their overlap is the maximal and above some threshold (in our experiments, the threshold is 60%). Two regions are matched if they satisfy two conditions: (1) the two regions are repeated, (2) their descriptors are the nearest neighbor in the descriptor space.

The relative measures include repeatability score and matching score. Note that the definitions of the repeatability and the matching score in this chapter are slightly different from that in [31]. In [31], the repeatability for a given pair of images is computed as the ratio between the number of the repeated points and the smaller number of the detected points in the pair of images. This may cause some inaccuracy problems in some cases. For example, assume 100 points are detected in image 1. Due to some variations, such as illumination change, assume 50 points are detected in image 2. If the 50 points are repeated, by the definition of repeatability in [31], the repeatability is 100%. This result is undesirable because 50 points are not detected

(a)

(b)

Figure 6.10: Sensitivity of the offset parameter δ with respect to the detection results. (a) The average number of repeated points (top line) and matched points (lower line). (b) The average value of repeatability (top line) and matching score (lower line).

Figure 6.11: Samples from the 8 data sets of the Oxford database. ‘boat’ and ‘bark’: scale and rotation change. ‘graf’ and ‘wall’: viewpoint change. ‘ubc’: JPEG compression. ‘bikes’ and ‘trees’: image blur. ‘leuven’: lighting change. Two (the 1st and the 6th) of the six images are shown for each data set. The top image in each set is used as the reference image in the experiments.

in image 2. Therefore, in this chapter we define the repeatability as the ratio between the number of the repeated points and the larger number of the detected points in a given image pair. Similarly, we also define the matching score as the ratio between the number of correct matches and the larger number of the detected points in a given image pair. All these points should be within the common area and the common scales of the image pair.

The publicly available Oxford database provided by [31] is used to evaluate the detectors.

This database contains 8 data sets, from which images are shown in Fig. 6.11. These data sets include five different changes in imaging conditions for structured and textured scenes: scale change, viewpoint change, JPEG compression, image blur, and lighting change. Each data set consists of 6 images with 5 homographies between the first image and the other five images. In all experiments reported here, interest points are detected on the downsampled images.

The parameter setting for the ROLG detector is as follow. Interest points are detected in 12 scales: $\{\sigma_n\}_{n=1,2,\dots,12} = \{1.6 \times 2^{1/3}, 1.6 \times 2^{2/3}, 3.2, \dots, 1.6 \times 2^4\}$. Instead of continuously increasing the ROLG mask size, the 12 scales are divided into 4 octaves by downsampling the previous octave. Each octave contains 3 scales $\{\sigma_{no}\}_{no=1,2,3} = \{1.6 \times 2^{1/3}, 1.6 \times 2^{2/3}, 3.2\}$.

Four benchmark detectors, the MSER detector [53], the Harris-affine (HR-A) detector [44], the Hessian-affine (HS-A) detector [44], and the SIFT detector [48], are compared with the ROLG detector. The default parameters given by the authors are used for each detector.

The results of the absolute measures and relative measures are shown in Fig. 6.12 and Fig. 6.13, respectively. Each figure in Fig. 6.12 and Fig. 6.13 contains 8 columns, corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of each figure shows the results on the scale change structured sequence ‘boat’, the scale change textured sequence ‘bark’, the viewpoint change structured sequence ‘graf’, the viewpoint change textured sequence ‘wall’, the JPEG compression sequence ‘ubc’, the blurring structured sequence ‘bikes’, the blurring textured sequence ‘trees’, and illumination change sequence ‘leuven’, respectively.

Results for the scale change and in-plane rotation are shown in the 1st and 2nd columns of Fig. 6.12 and Fig. 6.13. On both the structured scene (Fig. 6.11 boat) and the textured scene (Fig. 6.11 bark), the ROLG detector gives the best results.

Results for the viewpoint change are shown in the 3rd and 4th columns of Fig. 6.12 and Fig. 6.13. When the viewpoint change is small, the ROLG detector obtains higher repeatability

Figure 6.12: Number of repeated interest points (a) and number of correct matched interest points (b) on the Oxford database. Each figure contains 8 columns corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.

score and larger numbers of repeated points and matched points than those of the other detectors on the structured scene (Fig. 6.11 graf). As the mask of the ROLG filter is not adapted to viewpoint change, the performance of the ROLG filter drops faster than that of the MSER detector under the viewpoint change increasing. The most stable one for the structured scene is the MSER detector, but its number of repeated points is small. On the textured scene (Fig. 6.11 wall), the ROLG detector outperforms the other detectors.

The 5th columns of Fig. 6.12 and Fig. 6.13 show the results for the JPEG compression

Figure 6.13: Repeatability score (a) and matching score (b) on the Oxford database. Each figure contains 8 columns corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.

sequence (Fig. 6.11 ubc). The Hessian-affine detector and the Harris-affine detector show best performance, but the number of repeated points is small. When the distortion under JPEG compression is low, the repeatability score and the matching score of the ROLG detector are as good as those of the Hessian-affine detector. Moreover, both the number of repeated points and the number of matched points of the ROLG detector are larger than those of the other detectors.

The 6th and 7th columns of Fig. 6.12 and Fig. 6.13 show the results for blur images. The ROLG detector outperforms the other detectors on both the structured scene (Fig. 6.11 bikes)

and the textured scene (Fig. 6.11 trees).

Results for the illumination change (Fig. 6.11 leuven) are shown in the 8th column of Fig. 6.12 and Fig. 6.13. As the ROLG filter is robust to the variations caused by illumination changes, the ROLG detector outperforms the other detectors.

We compare the running time of the SIFT and ROLG detectors under the Window 7 system with the Intel Core i5 CPU 3.2GHz and RAM 4GB. The code of the SIFT detector is downloaded from <http://www.vlfeat.org/~vedaldi/assets/sift/versions/sift-0.9.16.tar.gz>. The ROLG detector is implemented by the Matlab programming language. Both the image size and the number of detected interest points affect the running time. From Fig. 7.8(a) it is seen that the ROLG and SIFT detectors detect the similar number of repeated points on the ‘bark’ data set. Therefore, we test the running time of these two detectors on this data set. The average running time per image of size 382×255 for the ROLG detector is 4.1s and that for the SIFT detector is 1.2s. It is not a surprise that the SIFT detector is faster than the ROLG detector as the former optimizes its speed by utilizing the nice properties of the Gaussian linear filter.

6.5.2 Application to Face Recognition

Face recognition is an active research topic [108–112], and some work has been done to apply the SIFT detector and descriptor in face recognition [42]. In the following experiments, we compare the ROLG detector with 5 state-of-the-art detectors, the SIFT detector [48], the MSER detector [53], the Harris-affine (HR-A) detector [44], the Hessian-affine (HS-A) detector [44], and the SURF detector [49]. Using the default parameters given by their respective authors, all of these detectors detect too few points and lead to very poor performance. Thus, we decrease the contrast threshold and find that zero is the best for all detectors. Therefore, the thresholds used to remove low contrast interest points are set to zero for all detectors. The MSER detector is controlled by several parameters. Even if we set the contrast threshold to the smallest zero,

Figure 6.14: Sample images in AR, ORL, GT, FERET and LFW databases. They show the typical image variations of the same persons in each database.

there is no point detected on the ORL database. In order to make MSER detector workable for the face recognition experiments, we reduce the minimum size of output region of the MSER detector. The minimum size of output region is set to $1/4$ of its default setting to make it workable on all face databases as its recognition performance is better than those with $1/2$ and $1/8$ on the ORL database. This optimal parameter setting is further confirmed on the AR database. The matching procedures described in [48] are employed in these experiments.

AR [113], ORL [114], Georgia Tech (GT) [115], FERET [116], and labeled faces in the wild (LFW) [117] databases are chosen to test the discriminative power of the interest points in face recognition. Some sample images of these databases are shown in Fig. 6.14. Before the interest point detection, images are resized to those commonly used in most other face recognition approaches. The rank 1 recognition rates and the cumulative matching curves are used to evaluate the detectors. The cumulative matching curve of the Harris-affine detector is not drawn in Fig. 6.17, Fig. 6.18 and Fig. 6.19, because it is drastically lower than that of the other detectors in these three figures.

Figure 6.15: Cumulative matching curves on AR database.

6.5.2.1 Results on AR Database

Table 6.1: Rank one recognition rate on AR database.

	ROLG	SIFT	SURF	MSER	HS-A	HR-A
AR	98.3%	94.3%	92.6%	92.7%	88.6%	74.5%

Color images of the AR database are converted to gray images and normalized into the size of 60×85 . 75 subjects with 14 nonoccluded images per subject are selected. The first 7 images of all subjects are chosen as gallery set, and the remaining 7 images as probe set.

Table 7.1 gives the rank one recognition rates. Fig. 6.15 shows the cumulative matching curves. From Table 7.1, it is clear to see that the ROLG detector outperforms the other detectors. Images in the AR database are taken under controlled conditions of the illumination and viewpoints [113]. The variations of the test images are well represented by the gallery images. Hence, the ROLG detector, the SIFT detector, the SURF detector and the MSER detector achieve high recognition rates. The Harris-affine detector gives the worst performance because

Figure 6.16: Cumulative matching curves on ORL database.

human face is a non-rigid surface, and there are few sharp corners in a face image.

6.5.2.2 Results on ORL Database

Table 6.2: Rank one recognition rate on ORL database.

	ROLG	SIFT	SURF	MSER	HS-A	HR-A
ORL	96.5%	90.0%	78.5%	91.0%	80.0%	66.5%

Images of the ORL database are normalized into the size of 50×57 . The first 5 images of all 40 subjects are chosen as gallery set, and the remaining 5 images as probe set.

The rank one recognition rates are shown in Table 6.2 and the cumulative matching curves are shown in Fig. 6.16. Although the ORL database is smaller than the AR database, the performance of all the detectors here are poor compared to that on the AR database. The main reason could be the smaller image size, and the information captured by the detectors on the ORL database is less than that on the AR database. Nevertheless, as shown in Table 6.2 and Fig. 6.16, the ROLG detector still outperforms the other detectors.

Figure 6.17: Cumulative matching curves on GT database.

6.5.2.3 Results on GT Database

Table 6.3: Rank one recognition rate on GT database.

	ROLG	SIFT	SURF	MSER	HS-A	HR-A
GT	91.1%	84.0%	84.6%	81.1%	74.0%	47.4%

Color images of the GT database are converted to gray images and normalized into the size of 60×80 . The first 8 images of all 50 subjects are chosen as gallery set, and the remaining 7 images as probe set.

Rank one recognition rates and cumulative matching curves are shown in Table 6.3 and Fig. 6.17, respectively. Images in the GT database have large variations in expression, pose and illuminations. Hence, the performance of all the detectors is poor compared with that of the AR database. However, the ROLG detector still outperforms the other detectors, and its rank one recognition rate is still larger than 90%.

Figure 6.18: Cumulative matching curves on FERET database.

6.5.2.4 Results on FERET Database

Table 6.4: Rank one recognition rate on FERET database.

	ROLG	SIFT	SURF	MSER	HS-A	HR-A
FERET	98.2%	89.9%	89.6%	89.3%	85.3%	49.7%

Images in the FERET database are cropped into the size of 60×80 . 1194 subjects with 2 images per person are selected. The first image of all subjects is chosen as gallery set, and the second image as probe set.

The experiment results are shown in Table 6.4 and Fig. 6.18. Although the number of subjects of the FERET database is drastically larger than that of the GT database, the rank one recognition rates of all detectors on the FERET database are higher than those on the GT database. The reason is that the variation between the gallery set and the test set is small for the FERET database. For this high quality database, the ROLG detector significantly outperforms the other detectors over all ranks.

Figure 6.19: Cumulative matching curves on LFW database.

6.5.2.5 Results on LFW Database

Table 6.5: Rank one recognition rate on LFW database.

	ROLG	SIFT	SURF	MSER	HS-A	HR-A
LFW	36.4%	27.6%	20.9%	16.1%	16.0%	10.4%

Color images are converted to gray images and cropped into the size of 64×64 . 134 subjects with 10 images per person are selected. The first 5 images of all subjects are chosen as gallery set, and the remaining 5 images as probe set.

Table 6.5 and Fig. 6.19 show the experiment results. For the LFW database, significant variations in face expression, pose, illumination and occlusion exist. These variations result in very poor performance for all detectors in this experiment. For this very difficult database, the ROLG detector also significantly outperforms the other 5 detectors over all ranks.

6.6 Summary

In this chapter, a novel nonlinear filter named rank order Laplacian of Gaussian (ROLG) filter is proposed, based on which a new interest point detector called ROLG detector is developed. The proposed ROLG filter is a weighted rank order filter. Compared to the SIFT detector, the ROLG detector detects less spurious and unstable points, and is more robust to abrupt variations of images caused by illumination and geometric changes. Experiment results demonstrate that its performance is better compared to 4 state-of-the-art detectors in terms of the repeatability and the discrimination of the interest points. The application of interest point detectors to face recognition on five databases further verifies the superiority of the proposed ROLG detector.

Chapter 7

Interest Point Detection Based on Vote of Confidence

In this chapter¹, a vote of confidence (VC) based detector is proposed to detect bright or dark regions from images. Whether a local region is bright or dark is voted by all pixels in this region and its surroundings. Compared to the contrast based detectors, such as the popular SIFT detector, the VC detector is robust to illumination change and abrupt variations. Experiments are conducted on benchmark database to verify the superior performance of the VC detector in terms of the repeatability and discrimination. This detector is also evaluated in the applications of face recognition and image matching.

7.1 Introduction

Voting without the restriction of color, race, income and gender is the basic building block of democracy. It also plays an important role in the fields of economic, education and engineering. In computer vision and pattern recognition, algorithms using the voting were widely developed to achieve a better performance [34, 118, 119]. The interest point detection, which provides an effective way to represent images with sparse local patches, is an important research topic in

---

¹This work is partially published in ICASSP2013 (conference paper [1] in author's publication list).

computer vision, image processing and recognition [1, 31, 32, 38, 41, 120]. It has been proven to be well suitable to dealing with the challenges of clutter, occlusion and variations of viewing condition [121]. In this chapter, we propose a vote of confidence (VC) based detector. It is inspired by the analyses of the existing detectors given as follows.

As a powerful tool for computer vision, various detectors have been developed in the last decades [2, 3, 6, 43–45, 47, 48, 50, 52, 53, 68, 73, 103, 122–125], most of which are designed directly based on image contrast. The representative ones include the Harris [43, 122], Hessian [47], Harris Laplace/affine [44], Hessian Laplace/affine [44], SIFT [48] and SURF [2] detectors. They detect interest points from the responses derived from the first or second order derivative of image intensity. Such detectors prefer the local structures with high contrast. Low contrast structures will not be easily detected even if they are stable under difference variations. Moreover, the first or second order derivative amplifies the image noise. This causes these detectors sensitive to noise. The ROLG detector [103] uses the rank order filter instead of the linear filter to reduce the influence of noise and the nearby structures. However, it still prefers the structures which have high contrast.

Detectors which are not directly designed from image contrast are also developed. The MSER [53, 123], MSCR [124], PCBR [6] and BPLR [125] detectors are based on the image segmentation algorithms. As image segmentation is still a challenging task, the performance of these detectors becomes poor under image blurring in which the boundaries of structures turn to unclear [1]. The statistical properties of local regions are employed by the SUSAN [45], FAST [68], and salient region [50, 73] detectors. Both the SUSAN and FAST detectors use the similarity between the nucleus (central pixel) and its surrounding pixels to generate the corner map. As the corner maps of the both detectors are generated from local regions with fixed size, these two detectors are not scale invariance. The number of interest points detected by the salient region detector is small due to the greedy cluster method used to group the nearby interest points [1].

Figure 7.1: (a) input image. (b) an enlarged image patch. (c)&(d) voting for brightness: (c) voting results by the pixels in the surrounding part and (d) voting results by the pixels in the inner part. (e)&(f) voting for darkness: (e) voting results by the pixels in the surrounding part and (f) voting results by the pixels in the inner part. For each voting pixel in (c) to (f), green color represents voting with 1 while blue color means voting with 0. Best viewed in color.

Instead of computing the corner map directly from the image contrast (e.g. SIFT [48]), or using the image segmentation algorithm (e.g. MSER [53]) or based on the local statistical property (e.g. salient region detector [50]), we proposed a vote of confidence (VC) based detector in this chapter. Each local region is separated into a concentric ring and its surrounding circle. Mutual voting is conducted by these two parts. The interest points are selected using the non-maximum suppression algorithm for the map of the voting confidence score. The scale of each interest point is estimated by a proposed grouping method instead of the local maximum in the scale space as done in [48]. The proposed VC detector is robust to illumination changes and effective to cluttered structures.

7.2 Vote of Confidence Based Detector

Instead of requiring all the pixels in the local regions brighter or darker than their surrounding as done by the MSER detector [53], the brightness/darkness of a local region is measured by a linear combination of the two levels of confidence defined as follows.

Confidence level 1: the normalized number of pixels in a region that are brighter/darker than

Figure 7.2: Voting maps of (a) bright regions and (b) dark regions. Best viewed in color.

the majority of the pixels in its surrounding region.

Confidence level 2: the normalized number of pixels in the surrounding region that are brighter/darker than the majority of the pixels in its surrounded center region.

By using these two confidence levels, the measurement of the brightness/darkness has a larger toleration of the illumination variation and abrupt structures than using only one. In order to simplify the problem of local region detection, we restrict the local region to a circle image patch, which is also adopted by many detectors [3, 45, 48, 50]. Each local region (for example, shown in Fig. 7.1(b)) is separated into two parts: inner circle disk S_1 and its surrounding ring S_2 . Confidence levels 1&2 are generated from these two parts.

The vote of confidence (VC) is proposed in Section 7.2.1 to measure the degree of brightness and darkness. Algorithm to remove the unstable points on ridge is presented in Section 7.2.2. The VC detector in multiple scales is given in Section 7.2.3.

7.2.1 Voting Algorithm

Two quantitative measurements, named vote of confidence for brightness and darkness (VCB and VCD), are proposed. As the VCB and VCD follow the analogous rules, we take the VCB

as an example to derive the voting algorithm.

A bright image patch should be bright in the central region and dark in its surrounding. Therefore, for each image patch, such as Fig. 7.1(b), the VCB is determined by two parts: the VC that the inner circle S_1 is bright and the VC that its surrounding ring S_2 is dark. The confidence of brightness for S_1 is voted by the pixels in S_2 , and similarly the confidence of darkness for S_2 is voted by the pixels in S_1 . The following question is that how a pixel votes its counterpart region, for example, how a pixel I_i in S_2 votes for the brightness of S_1 ? Obviously, if all the pixels in S_1 are brighter than I_i , I_i should vote 1 for the brightness of S_1 . However, this makes it sensitive to impulsive noise and abrupt structures. In order to alleviate this problem, in this chapter the voting rule is set as follow.

Voting rule: If a pixel I_i is brighter/darker than more than half pixels in the counterpart region S_j , it votes 1 for the darkness/brightness of S_j , otherwise, it votes 0. Let the median for S_j be ϕ_j . The voting rule for brightness is

$$vb(I_i, S_j) = \begin{cases} 1, & \text{if } I_i < \phi_j \\ 0, & \text{otherwise} \end{cases}, \quad (7.1)$$

and that for the darkness is

$$vd(I_i, S_j) = \begin{cases} 1, & \text{if } I_i > \phi_j \\ 0, & \text{otherwise} \end{cases}. \quad (7.2)$$

One example of the voting for the bright region is shown in Fig. 7.1(c) and (d), in which green color pixels represent voting with 1 while blue color pixels mean voting with 0. Fig. 7.1(c) depicts the voting results of the pixels in S_2 for the brightness of S_1 . As S_1 is bright, the majority of the pixels in S_2 vote 1 for the brightness of S_1 . Similarly, in Fig. 7.1(d) the majority of the pixels in S_1 vote 1 for the darkness of S_2 .

The VCB is a linear combination of the normalized voting results for the brightness of the inner circle (Confidence level 2) and that for the darkness of the surrounding ring (Confidence level 1). The VCB at location (x, y) is

$$VCB(x, y) = \sum_{(m,n) \in S_2} \frac{vb(I(x-m, y-n), S_1)}{s_2} + \sum_{(m,n) \in S_1} \frac{vd(I(x-m, y-n), S_2)}{s_1}. \quad (7.3)$$

Some properties of the VCB are concluded in the following Propositions.

Proposition 1: $0 \leq VCB(\mathbf{x}) \leq 2$. If all pixels in S_1 are brighter than more than half of the pixels S_2 and all pixels in S_2 are darker than more than half of the pixels S_1 , the VCB for this image patch is 2. If a major part of S_1 is brighter than the median value of pixels in S_2 and a major part of S_2 is darker than the median value of pixels in S_1 , $VCB > 1$. If all pixels in S_1 are darker than or equal to the median value of pixels in S_2 and all pixels in S_2 are darker than or equal to the median value of pixels in S_1 , the VCB equals to 0.

In order to suppress the artificial noise caused by the countable voting results, a Gaussian filter $G(x, y, \sigma_0)$ (served as an interpolator as done in [126]) to smooth the voting score is needed, e.g.

$$\widehat{VCB}(x, y) = VCB(x, y) * G(x, y, \sigma_0), \quad (7.4)$$

where $*$ is the convolution operator, and σ_0 is set to 1 in this work. The response of the VCB (named VCB map) for Fig. 7.1(a) is shown in Fig. 7.2(a). It is seen that the bright regions have high responses while the dark regions have low responses.

The voting rules of the dark region can be derived in a similar way. One example of the voting for the dark region is shown in Fig. 7.1 (e) and (f). As this image patch is not a dark one, the majority of pixels in S_2 vote 0 for the brightness of S_1 , and the majority of pixels in S_1 vote

0 for the darkness of S_2 . The VCD is defined as

$$VCD(x, y) = \sum_{(m,n) \in S_2} \frac{vd(I(x-m, y-n), S_1)}{s_2} + \sum_{(m,n) \in S_1} \frac{vb(I(x-m, y-n), S_2)}{s_1}, \quad (7.5)$$

and its smoothed version is

$$\widehat{VCD}(x, y) = VCD(x, y) * G(x, y, \sigma_0). \quad (7.6)$$

The VCD map for the image in Fig. 7.1(a) is shown in Fig. 7.2(b). It enhances the dark regions and suppresses the bright regions.

7.2.2 Ridge Suppression

Interest points are extracted from the VC (VCB and VCD) maps by detecting the local peaks. However, the VC on ridge may be larger than 1 when the scale of the image patch matches with the width of the ridge. Slight vibration may cause false detection of interest points on the ridge. Such kind of unstable interest points need be removed.

Although the peaks of the VC map on ridge have large amplitude, the difference between the peak value and the maximum value in the corresponding surrounding region S_2 is small. Hence, we employ the ratio

$$\lambda_1 = (R(x, y) - \max\{R(x-m, y-n) | (m, n) \in S_2\}) / \max\{R(x-m, y-n) | (m, n) \in S_2\}, \quad (7.7)$$

where $R(x, y)$ represents the VC response $\widehat{VCB}(x, y)$ or $\widehat{VCD}(x, y)$ for symbolic simplicity, to remove the unstable interest points on the ridge. If λ_1 is small, it means the peak is very similar to its nearby region. Such interest point candidate is most likely on the ridge. We remove such candidates if $\lambda_1 < 0.05$, which is chosen experimentally.

7.2.3 VC Detector in Multiple Scales

Interest point detection in multiple scales is an important issue in vision applications. It is known that the local structures exist over a range of scale [50]. Lindeberg [74] employed a set of matching filters to detect the scale of the local features from this range. This method has been widely used in detectors, such as SIFT [48], SURF [2] and Harris/Hessian-laplacian [44] detectors. When a matching filter matches with the local structure, the response is maximized. Hence, the scale of the local structure is set proportional to the scale of the matched filter. The chosen matching filters in [74] are a set of normalized LoG filters with the standard division σ increasing. These filters are optimal in detecting the structures with the LoG shape. It is not the case for irregular structures. In real world, the shapes of local structures appear in different ways, and most of them are different from the LoG filter. Sometimes, it is difficult to get a stable local maximum in both the scale and spatial dimensions. In this case, using the local maximum along the scale dimension to determine the scales are not reliable. This problem also exists in the MSER detector [53]. This detector employs the local minima of the area-changing rate of the extremal regions along the intensity levels to determine the scale. Sometimes, no sharp valley exists along the scale dimension. Small disturbance may influence the scale selection. For example, Fig. 7.3 shows the scales detected by the MSER detector on the blob structures. Multiple scales are detected on one blob.

By changing the radius of the local image patches S_1 and S_2 , the VC detector achieves the multi-scale detection of local structures. Similar to other detectors, the structures detected by the VC detector appear in a wide range of scales. For example, Fig. 7.4 shows the response of the VCB at the center of a bright blob along the scale dimension. It is seen that no sharp maximum is generated along the scale dimension. In this case, the local extremum along the scale dimension is sensitive to noise, and it is unreliable in determining the scale size. From another perspective, as the local structure appears in multi-scales, the result should be more

Figure 7.3: Interest points detected by MSER detector. The circles indicate the scales of the detected local structures.

reliable if we use all such scales instead of only one to make a decision. In the following, a grouping method is proposed to cluster the connected interest points.

Let an interest point at location (x_i, y_i) and scale s_i be $\mathcal{P}_i = (x_i, y_i, s_i, R_i, F_i)$ where R_i is the VC response and $F_i \in \{Bright, Dark\}$ is the flag of bright or dark region. Each $\mathcal{P}_i$ is selected by requiring: (a) R_i is the local peak along the spacial dimensions, and (b) R_i is a local peak along the scale dimension with some toleration, i.e.

$$(R_i > \frac{1}{1 + \lambda_2} R(x_i, y_i, s_{i+})) \wedge (R_i > \frac{1}{1 + \lambda_2} R(x_i, y_i, s_{i-})), \quad (7.8)$$

where s_{i+} and s_{i-} are two immediate neighbors of s_i . λ_2 is set to be 0.1 in this chapter. The connection of two interest points $\mathcal{P}_i$ and $\mathcal{P}_j$ is defined as: 1) they are both bright (or both dark) regions, and 2) they are close along both the spacial and scale dimensions. In this work, the distance in the spacial dimensions is restricted as $\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} < 0.3s_i$, and in the scale dimension s_i and s_j should be the immediate neighbor or only one discrete scale exists

Figure 7.4: The VCB against the scale dimension at the center of a blob.

between them. All the connected interest points are clustered into one group. Assume a group $\mathbf{G}_p$ contains N interest points as $\mathbf{G}_p = \{\mathcal{P}_{p1}, \mathcal{P}_{p2}, \dots, \mathcal{P}_{pN}\}$. A representative interest point $\mathcal{P}_p = (x_p, y_p, s_p, R_p, F_p)$ for this group is generated by setting the location

$$x_p = \sum_{i=1}^N x_i / N, \quad (7.9)$$

$$y_p = \sum_{i=1}^N y_i / N, \quad (7.10)$$

the scale

$$s_p = \sum_{i=1}^N R_i^2 s_i / \sum_{i=1}^N R_i^2 \quad (7.11)$$

and the response

$$R_p = \max\{R_1, R_2, \dots, R_N\}. \quad (7.12)$$

By weighting the scale with R_i^2 , the scales with large VC responses have high influence on determining the scale of the corresponding group.

The proposed algorithm for the VC detector is summarized as follow:

1. Generate the VC response on multi-scales.
2. Detect the local maximums on both the scale and spatial dimensions with some toleration.
3. Remove the points on ridges.
4. Group the interest points which are corresponding to the same structure.

7.3 Experiments

A visual comparison is given to illustrate the performance of the VC detector under the illumination variation in Section 7.3.1. Experiment in Section 7.3.2 is used to evaluate the detectors under intraclass variation based on the protocols given in [73]. Experiment in Section 7.3.3 is designed to test the repeatability and the discrimination of the interest points based on the protocols given in [31]. Experiment in Section 7.3.4 is carried out to test the performance of the interest points in the application of face recognition. In these experiments, VC, HR-A, HS-A, Salient, S_r stand for vote of confidence detector, Harris affine detector [44], Hessian affine detector [44], salient region detector [50], and radial saliency detector [3], respectively.

7.3.1 Visual Inspection

The images from the Oxford database [31] ‘leuven’ data set are used to depict some visual results of the VC detector under illumination changes. The setting of different detectors is the same as that given in Section 7.3.3. It is seen that the number of the interest points detected by the SIFT detector (shown in Fig. 7.5(b) and (f)) and MSER detectors (shown in Fig. 7.5(c) and (g)) decreases with the scenes darkening. In contrast, the illumination change has little effect on the VC detector. Most of the structures detected by the VC detector are repeated in these two scenes (shown in Fig. 7.5(d) and (h)). Besides, although some structures in this scene cluster with each other, the VC detector can still separate them and detect them out.

Figure 7.5: (a)&(e) Input images. (b)&(f) Interest points detected by SIFT detector. (c)&(g) Interest points detected by MSER detector. (d)&(h) Interest points detected by VC detector.

7.3.2 Intraclass Variations

The VC detector under intraclass variations is tested following the rules given in [3, 73]. 200 images from the Caltech Motorbikes set² is used to evaluate the detectors. The ground-truth locations³ given in [73] is utilized to estimate the affine transformation of the motorbikes in images.

Similar to that in [3], experiments are conducted on images with lower image resolution. Scores are computed on scales from $s=3$ to $s=22$ of half-sampled images. A detected region is considered to be matched with that in the original image if they satisfy the following three conditions recommended in [73]: (1) the distance between the positions of these two detected regions is within 10 pixels of the original image size, (2) the difference between their scales is within 20 percent of the original one, and (3) the normalized mutual information between

²This is part of the Caltech-101 object categories database. Available from <http://www.robots.ox.ac.uk/~vgg/data/>.

³Available from <http://www.robots.ox.ac.uk/~timork/Saliency/AffineInvariant Saliency.html>.

(a)

(b)

Figure 7.6: (a) Repeatability results for the Caltech Motorbikes images with a clear background. (b) Repeatability results for the Caltech Motorbikes images with clutter background

these two detected regions ($MI(A, B) = 2(H(A) + H(B) - H(A, B))/(H(A) + H(B))$) is greater than 0.2 [3]. The measurement of the average correspondence score S given in [73] is as follows. Assume there are M images in the data set. For each image in this set, the N regions with the highest responses are detected. Every image in this set can be considered as the reference image. For the i^{th} reference image, the correspondence score S_i is the ratio between the total number of matched regions and the total number of detected regions for all the rest images in the data set, defined as

$$S_i = \frac{\text{Total number of matches}}{\text{Total number of detected regions}} = \frac{N_m^i}{N(M-1)}. \quad (7.13)$$

As suggested in [73], the score S_i is calculated for $M/2$ different selections of the reference images and averaged to give S . Experimental results are shown in Fig. 7.6.

For the Motorbike data set with clear background, all the motorbikes have similar structures, such as two wheels, seat, tank and so on. Most of them have similar properties. For example, the majority of the inner region of the front wheel is brighter than its surrounding. Although they have some variation between different wheels, our proposed detector only counts the number of pixels which are brighter (darker) than its counterpart. Hence, such kind of structures are detected and repeated between different motorbikes, as shown in Fig. 7.7(a). Experimental results in 7.6(a) show that our detector outperforms the other three.

When dealing with the images with cluttered background, as shown in 7.7(b), the background is more blurred compared to the motorbikes which the shot focuses on. There is no other pervious knowledge to know which part is important, and which part is the background. For the salient region detector [50], most of the high response interest points are on the motorbikes because the salience of the smoothed region is small. Hence, its performance is better compared with our proposed detector. However, this also means the salient region detector [50] is sensitive to image blur. Our proposed detector is insensitive to image blur, and interest points

Figure 7.7: Circles represent 10 regions of the highest local maximum of the VC detector detected on images in the Motorbikes database: left are the images clear background and right are the images with clutter in the background.

Figure 7.8: (a) Number of repeated interest points and (b) number of corrected matched interest points on the Oxford database. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.

are detected not only on the motorbikes, but also in the background. Thus, its performance is not as good as the salient region detector. However, its performance is still better than the SIFT detector [48] and the radius saliency detector [3].

7.3.3 Repeatability and Discrimination Tests

The aim of this experiment is to evaluate the detectors under different variations based on the protocol in [31]. Detectors are compared by repeatability and matching score. Two detected regions are repeated if their overlap is above a certain threshold (it is set to be 60% as suggested in [31]). The repeatability/matching score is the ratio between the number of repeated/matched points and the larger number of detected points in the same scenes of each image pair.

The test data sets are chosen from the standard publicly available Oxford database in [31].

8 data sets are contained in this database, named as ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’. These data sets include five changes of images: scale change (‘boat’ and ‘bark’), viewpoint change (‘graf’ and ‘wall’), JPEG compression (‘ubc’), image blur (‘bikes’ and ‘trees’) and lighting change (‘leuven’). Each data set contains 6 images with 5 homographies between the first image and the other five images.

Similar to that done in [3], interest points are detected on the half-sampled images. For the VC detector, interest points are detected on 5 octaves by half-sampling the previous octave. In each octave, local extrema are detected on 6 scales: $\{\sigma_n\}_{n=1,2,\dots,6} = \{3, 4, \dots, 8\}$. The threshold to remove the low VC points is set to be 1.5. Four detectors, the MSER [53], HR-A [44], HS-A [44] and SIFT [48] detectors are compared with the VC detector. The default parameters of these four detectors supplied by authors are employed here. SIFT descriptor [48] is used to describe interest points for all detectors included here.

The results of the absolute and relative measures are shown in Fig. 7.8 and Fig. 7.9, respectively. Each figure in Fig. 7.8 and Fig. 7.9 contains 8 columns, corresponding to the 8 data sets. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of each figure shows the results on the scale change structured sequence ‘boat’, the scale change textured sequence ‘bark’, the viewpoint change structured sequence ‘graf’, the viewpoint change textured sequence ‘wall’, the JPEG compression sequence ‘ubc’, the blurring structured sequence ‘bikes’, the blurring textured sequence ‘trees’, and the illumination change sequence ‘leuven’, respectively.

The 1st and 2nd columns of Fig. 7.8 and Fig. 7.9 depict the results of the scale changes. It is seen that on both the structured scene (‘boat’) and the textured scene (‘bark’), the VC detector gives the best repeatability and matching score. The number of repeated and matched points are similar for the SIFT and VC detector in these two scenes. The VC detector produces less unstable interest points than the SIFT detector.

Figure 7.9: (a) Repeatability score and (b) matching score on the Oxford database. In each column, horizontal axis represents the image index in the corresponding data set. From left to right of (a) and (b) are the results on the image sequence of ‘boat’, ‘bark’, ‘graf’, ‘wall’, ‘ubc’, ‘bikes’, ‘trees’ and ‘leuven’, respectively.

The 3rd and 4th columns of Fig. 7.8 and Fig. 7.9 show the results for viewpoint changes. For the structure scene (‘graf’), the VC detector achieves the best results when the variation is small. However, as the VC detector does not adopt the affine-invariance techniques, its performance drops faster than that of the MSER detector in case the viewpoint change increasing. The MSER detector is more stable than other detectors under large affine-variation. However, results in Fig. 7.8 show that its numbers of repeated and matched points are small. On the textured scene (‘wall’), the numbers of repeated and matched points of the VC and SIFT detectors are drastically larger than others. The VC detector achieves the best performance in terms of the repeatability and matching score.

The results for the JPEG compression sequence (‘ubc’) are depicted in the 5th columns of Fig. 7.8 and Fig. 7.9. The HS-A and HR-A detectors show the highest repeatability and

matching score, though their numbers of repeated and matched points are small. As the JPEG compression generates some low contrast unstable structures which can be detected by the VC detector, the repeatability score and the matching score of the VC detector are not as good as those of the HS-A and HR-A detectors. But it is still better than the SIFT detector. Moreover, both the numbers of repeated and matched points of the VC detector are larger than those of the HR-A, HS-A and MSER detectors.

The results for blurred images are shown in the 6th and 7th columns of Fig. 7.8 and Fig. 7.9. It is seen that the VC detector outperforms others on both the structured scene ('bikes') and the textured scene ('trees'). The 8th column of Fig. 7.8 and Fig. 7.9 shows the results for the illumination change ('leuven'). As the VC detector is independent of the image local contrast, it is more stable than the other detectors under the illumination variation.

7.3.4 Application to Face Recognition

Face recognition is an active research topic [108–110] and some work has been done to apply SIFT descriptor in face recognition [42, 127]. In this part, the VC detector is compared with the MSER [53], HR-S [44], HS-A [44], and SIFT detectors [48]. As the default setting produces too few interest points for the face recognition for all detectors, the contrast threshold is set to be zero for all detectors in this experiment. For the VC detector, the threshold to remove the low VC points is set to be 1. For the MSER detector, the minimum size of output region is set to be 1/4 of its default setting to make it workable on all face databases. All the detected interest points are described by the SIFT descriptor. The matching algorithm is the one given in [48].

The AR [113], ORL [114], GT [115] and FERET [116] databases are used to evaluate these detectors. For the AR database, gray images are normalized into the size of 60×85 . 75 subjects with 14 nonoccluded images per person are selected. The first 7 images of all subjects are chosen as gallery set, and the remaining 7 images as probe set. Images in the ORL database are

normalized into the size of 50×57 . The first 5 images of all 40 subjects are chosen as gallery set, and the remaining 5 images as probe set. Gray images in the GT database are normalized into the size of 60×80 . The first 8 images of all 50 subjects are chosen as gallery set, and the remaining 7 images as probe set. For the FERET database, images are cropped into the size of 60×80 . 1194 subjects with 2 images per person are selected. The first 1 image of all subjects is chosen as gallery set, and the remaining 1 image as probe set.

Table 7.1 shows the recognition rate on the four databases. It is clear to see that the VC detector outperforms the other detectors. Images in the AR database are taken under controlled conditions of the illumination and viewpoints [113]. The variations of the test images are well represented by the gallery images. Hence, the VC, SIFT and MSER detector achieve high recognition rates. The Harris-affine detector gives the worst performance because human face is a non-rigid surface, and there are few sharp corners in a face image. Although the ORL database is smaller than the AR database, the performance of all the detectors here are poor compared to that on the AR database. The main reason could be the smaller image size, and the information captured by the detectors on the ORL database is less than that on the AR database. Images in the GT database have large variations in expression, pose and illuminations. Hence, the performance of all the detectors is poor compared with that of the AR and ORL databases. However, the VC detector still outperforms the other detectors, and its recognition rate is still larger than 90%. Although the number of subjects of the FERET database is drastically larger than that of the GT database, the recognition rates of all detectors on the FERET database are higher than those on the GT database. The reason is that the variation between the gallery set and the test set is small for the FERET database. For this high quality database, the VC detector significantly outperforms the other detectors.

Table 7.1: Recognition Rate on AR, ORL, GT, and FERET Databases.

	AR	ORL	GT	FERET
VC	96.8%	94.5%	91.1%	96.7%
SIFT	94.3%	90.0%	84.0%	89.9%
HS-A	88.6%	80.0%	74.0%	85.3%
HR-A	74.5%	66.5%	47.4%	49.7%
MSER	92.7%	91.0%	81.1%	89.3%

7.4 Summary

A vote of confidence based detector is presented in this chapter to detect bright and dark regions from images. Voting rules are proposed to tolerate the impulsive noise and abrupt structures. Grouping algorithm is designed to determine the scales of local structures. Compared to the LoG filter, the VC response is independent of image contrast and robust to cluttered surrounding. Experiment results demonstrate that the VC detector has better performance in dealing with scale, blurring and illumination changes in terms of repeatability and matching score in most cases. Its superiority is further verified on the experiments of face recognition and image matching.

Chapter 8

Conclusions and Future Work

In this thesis we present several median based approaches for noise suppression and interest point detection. This chapter is organized as follows. In Section 8.1, our contributions on designing the noise suppression filters are presented. The contributions in developing the interest point detectors are given in Section 8.2. Some potential directions for future work are discussed in Section 8.3.

8.1 Contributions in Noise Suppressing

Both the mean and median operations have their own advantages and disadvantages in noise suppression. The arithmetic mean is effective in suppressing the short-tailed noise but not for the long-tailed noise. It blurs signal structures. Correspondingly, the median filter is effective in suppressing the long-tailed noise, and has the advantage in edge preservation. However, its computational complexity is higher compared to the arithmetic mean. Moreover, it cannot effectively attenuate the short-tailed noise. It is desirable to design filters which can effectively suppress both the shot- and long-tailed noise with lower computational complexity. The ITM algorithm offers a new bridge between the mean and median filters by iteratively truncating the extreme samples in the input data set. The ITM filter built on the ITM algorithm owns merits of

both the arithmetic mean and order statistic median operations. Our work in noise suppressing is inspired by the ITM filter. The contributions are as follows:

8.1.1 Fast ITM Filter

We analyze some further properties of the ITM filter. By employing the Cramer-Rao lower bound and the Monte Carlo simulations, it is verified that the ITM filter outperforms the median filter in suppressing both Gaussian and Laplacian noise. It is proved that the truncated samples will always be truncated in the subsequent iterations. As all truncated samples must be either truncated to the lower or upper bound in the subsequent iterations, we do not need to access such samples for computing the mean, threshold, and checking whether they should be truncated or not. This observation inspires the proposed fast ITM (FITM) filter. The analysis reveals that the computational complexity of the ITM and FITM filters are of order $O(n\sqrt{n})$ and $O(n \log n)$, respectively. Although the FITM filter is of the same order as the median filter, experiment demonstrates that the FITM filter is fast than the standard median filter implemented by the quick-sort algorithm.

8.1.2 Weighted Iterative Truncated Mean Filter

We propose a rich class of filters named weighted ITM (WITM) filters. By iteratively truncating the extreme samples with a dynamic threshold, the output of the WITM filter moves from the weighted mean towards the weighted median. Proper stopping criterion is proposed to enable the WITM filters being terminated within a few iterations. Experiments show that the WITM filters outperform both the weighted mean and weighted median filters in many denoising applications. Three structures of the weighted ITM filter, named GWITM, LCWITM and DWITM filters, are proposed. By assigning both the positive and negative weights, these filter structures enable the WITM filter being designed as low-, band- and high-pass filters.

8.1.3 Iteratively Trimmed and Truncated Arithmetic Mean Filter

We propose the iterative trimmed and truncated arithmetic mean (ITTM) filters. The ITTM algorithm has a higher convergence rate than the ITM algorithm by simultaneously trimming and truncating the extreme samples. The proposed algorithm for the ITTM filter terminated by the stopping criterion in [7] has a linear computational complexity with the order of $\mathcal{O}(n)$. It is smaller than that of the median and FITM filters, both of which are $\mathcal{O}(n \log n)$. Although the ITTM filters use an iterative algorithm, only a few iterations is needed in all the experiments of this thesis to achieve a good de-noising performance.

We propose three types of outputs for the ITTM filter, named ITTM1, ITTM2 and ITTM3. The ITTM1 filter uses the average of the truncated samples as the output. It shows that the ITTM1 filter has the best performance in attenuating the short- and long-tailed additive noise among these three filters. The ITTM2 filter trims all the truncated samples and uses the average of the remaining samples as the filter output. Trimming all the extreme samples enables the ITTM2 filter having the best performance in suppressing exclusive impulsive noise. The ITTM3 filter simultaneously trims and truncates the extreme samples and uses the average of the remaining ones as the output. This filter has the best performance in attenuating the noise mixed by the short-, long-tailed and impulsive noise. For some real image cases, the ITTM3 filter achieves the best performance as some pixels of real images in a filter window can be considered as exclusive, impulsive or long-tailed noise.

8.2 Contributions in Interest Point Detection

The merits of the median operation inspire our work in developing the interest point detectors. The contributions are listed as follows.

8.2.1 Interest Point Detection Using Rank Order LoG Filter

Based on the median filter, we propose a weighted rank order filter named rank order Laplacian of Gaussian (ROLG) filter. Compared to the Laplacian of Gaussian filter, it is more effective in suppressing the response of edges and robust to the abrupt variations. We propose the ROLG detector based on the ROLG filter to detect the bright and dark regions from images. It shows that the ROLG detector detects fewer spurious and unstable points, and is more robust to abrupt variations of images caused by illumination and geometric changes.

8.2.2 Interest Point Detection Based on Vote of Confidence

We propose a vote of confidence based interest point detector. Although the ROLG filter is robust to the abrupt variations, it still prefers the image structures which have high local contrast. Structures with low contrast are difficult to detect even if they are stable under different variations. In order to alleviate this problem, we proposed a voting algorithm instead of using the intensity contrast to enhance the response of bright and dark regions. This voting algorithm is designed to tolerate the impulsive noise and abrupt structures. We propose a grouping algorithm to determine the scales of local structures. Compared to the LoG filter based detectors, this VC response is robust to the illumination changes and clutter surroundings.

8.3 Future Work

For the future work, we plan to further analyze the properties of the proposed filters and devote further efforts in reducing the computational complexity of the detectors.

8.3.1 Probability Analysis of the Proposed Filters

The threshold decomposition method provides a powerful tool to analyze the median filter. However, there are no such tools for analyzing the ITM, WITM and ITTM filters. The statistical

properties of these filters' outputs with respect to the number of iterations are still unknown except the two extreme cases: the number of iteration equals to zero in which the filter output is the mean, and the number of iteration equals to infinity in which the filter output is the median. It is difficult to analyze these filters with the currently available tools. Further efforts are needed to develop such tools.

8.3.2 Iterative Truncated Mean Filters in Multiple Dimensions

The sorting algorithm used for the median operation is time consuming. The computational complexity becomes even worse for the vector median filter. As the ITM filter reduces the time consuming by replacing the data sorting with the truncating algorithm, it is desirable to extend the iterative truncating algorithm into multiple dimensions.

8.3.3 Trimming and Truncating Algorithm for Robust Statistic

Traditional machine learning algorithm is sensitive to the outliers in the training data. As the trimming algorithm is robust to the outliers by directly remove such samples, the trimming and truncating algorithm is desirable in robust statistic.

8.3.4 Affine Invariant Interest Point Detector

As both the ROLG and VC detectors employ circular disks to detect the local features, their performance becomes poorer when affine variation exists. Experiments in Chapters 6 and 7 show that both the ROLG and VC detectors cannot effectively deal with large affine variations. In the future, we plan to extend the ROLG and VC detectors into affine invariant detectors.

Author's Publications

Journal Papers

1. **Zhenwei Miao** and Xudong Jiang, "Further Properties and a Fast Realization of the Iterative Truncated Arithmetic Mean Filter", *IEEE Transactions on Circuits and Systems Part II: Express Briefs*, vol. 59, no. 11, pp. 810-814, November 2012.
2. **Zhenwei Miao** and Xudong Jiang, "Weighted Iterative Truncated Mean Filter", *IEEE Transactions on Signal Processing*, vol. 61, no. 16, pp. 4149-4160, August 2013.
3. **Zhenwei Miao** and Xudong Jiang, "Additive and Exclusive Noise Suppression by Iterative Trimmed and Truncated Mean Algorithm", *Signal Processing*, accept with minor revisions.
4. **Zhenwei Miao** and Xudong Jiang, "Interest Point Detection Using Rank Order LoG Filter", *Pattern Recognition*, vol. 46, no. 11, pp. 2890-2901, November 2013.
5. **Zhenwei Miao** and Xudong Jiang, "Generalized Laplacian of Gaussian Filter and ℓ_0 -LoG Interest Point detector", *IEEE Transactions on Image Processing*, under review.

Conference Papers

1. **Zhenwei Miao** and Xudong Jiang, "A Vote of Confidence based Interest Point Detector", submitted to the 38th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Vancouver, Canada, May, 2013.
2. **Zhenwei Miao** and Xudong Jiang, "A Novel Rank Order LoG Filter for Interest Point Detection", in *Proceedings of the 37th IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 937-940, Kyoto, Japan, March, 2012.

Bibliography

- [1] T. Tuytelaars and K. Mikolajczyk, “Local invariant feature detectors: a survey,” *Foundations and Trends in Computer Graphics and Vision*, vol. 3, no. 3, pp. 177–280, 2008.
- [2] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in *European Conference on Computer Vision*, vol. 3951, pp. 404–417, 2006.
- [3] J. Maver, “Self-similarity and points of interest,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1211–1226, 2010.
- [4] “vlfeat.” http://www.anefian.com/face_reco.htm.
- [5] J. Li and N. M. Allinson, “A comprehensive review of current local features for computer vision,” *Neurocomputing*, vol. 71, no. 10-12, pp. 1771–1787, 2008.
- [6] H. L. Deng, W. Zhang, E. Mortensen, T. Dietterich, and L. Shapiro, “Principal curvature-based region detector for object recognition,” in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.
- [7] X. D. Jiang, “Iterative truncated arithmetic mean filter and its properties,” *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1537–1547, 2012.
- [8] I. Pitas and A. N. Venetsanopoulos, “Order statistics in digital image processing,” *Proceedings of the IEEE*, vol. 80, no. 12, pp. 1893–1921, 1992.
- [9] G. R. Arce, *Nonlinear Signal Processing: A Statistical Approach*. New York: Wiley, 2004.
- [10] R. Yang, L. Yin, M. Gabbouj, J. Astola, and Y. Neuvo, “Optimal weighted median filtering under structural constraints,” *IEEE Transactions on Signal Processing*, vol. 43, no. 3, pp. 591–604, 1995.
- [11] S. Hoyos, Y. Li, J. Bacca, and G. R. Arce, “Weighted median filters admitting complex-valued weights and their optimization,” *IEEE Transactions on Signal Processing*, vol. 52, no. 10, pp. 2776–2787, 2004.

- [12] L. Yin, R. Yang, M. Gabbouj, and Y. Neuvo, "Weighted median filters: a tutorial," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 43, no. 3, pp. 157–192, 1996.
- [13] D. Charalampidis, "Steerable weighted median filters," *IEEE Transactions on Image Processing*, vol. 19, no. 4, pp. 882–894, 2010.
- [14] R. C. Hardie and K. E. Barner, "Rank conditioned rank selection filters for signal restoration," *IEEE Transactions on Image Processing*, vol. 3, no. 2, pp. 192–206, 1994.
- [15] K. E. Barner, Y. Nie, and W. An, "Fuzzy ordering theory and its use in filter generalization," *EURASIP Journal on Advances in Signal Processing*, no. 4, pp. 206–218, 2001.
- [16] W. Zhou and D. Zhang, "Progressive switching median filter for the removal of impulse noise from highly corrupted images," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 46, no. 1, pp. 78–80, 1999.
- [17] A. Nieminen, P. Heinonen, and Y. Neuvo, "A new class of detail-preserving filters for image processing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 74–90, 1987.
- [18] X. D. Jiang, "Image detail-preserving filter for impulsive noise attenuation," *IEEE Proceedings: Vision, Image and Signal Processing*, vol. 150, pp. 179–185, June 2003.
- [19] G. R. Arce and R. E. Foster, "Detail-preserving ranked-order based filters for image processing," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 1, pp. 83–98, 1989.
- [20] S. Akkoul, R. Ledee, R. Leconge, and R. Harba, "A new adaptive switching median filter," *IEEE Signal Processing Letters*, vol. 17, no. 6, pp. 587–590, 2010.
- [21] T. Sun and Y. Neuvo, "Detail-preserving median based filters in image processing," *Pattern Recognition Letters*, vol. 15, no. 4, pp. 341–347, 1994.
- [22] J. B. Bednar and T. L. Watt, "Alpha-trimmed means and their relationship to median filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 32, no. 1, pp. 145–153, 1984.
- [23] S. R. Peterson, Y. H. Lee, and S. A. Kassam, "Some statistical properties of alpha-trimmed mean and standard type M filters," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 5, pp. 707–713, 1988.
- [24] Y. Lee and S. Kassam, "Generalized median filtering and related nonlinear filtering techniques," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 33, no. 3, pp. 672–683, 1985.

- [25] A. B. Hamza and H. Krim, "Image denoising: a nonlinear robust statistical approach," *IEEE Transactions on Signal Processing*, vol. 49, no. 12, pp. 3045–3054, 2001.
- [26] A. Flaig, G. R. Arce, and K. E. Barner, "Affine order-statistic filters: "medianization" of linear FIR filters," *IEEE Transactions on Signal Processing*, vol. 46, no. 8, pp. 2101–2112, 1998.
- [27] T. Aysal and K. Barner, "Generalized mean-median filtering for robust frequency-selective applications," *IEEE Transactions on Signal Processing*, vol. 55, no. 3, pp. 937–948, 2007.
- [28] R. Oten and R. J. P. de Figueiredo, "Adaptive alpha-trimmed mean filters under deviations from assumed noise model," *IEEE Transactions on Image Processing*, vol. 13, pp. 627–639, may 2004.
- [29] N. Zaric, N. Lekic, and S. Stankovic, "An implementation of the l-estimate distributions for analysis of signals in heavy-tailed noise," *IEEE Transactions on Circuits and Systems II*, vol. 58, no. 7, pp. 427–431, 2011.
- [30] J. G. Gonzalez and G. R. Arce, "Optimality of the myriad filter in practical impulsive-noise environments," *IEEE Transactions on Signal Processing*, vol. 49, no. 2, pp. 438–441, 2001.
- [31] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. van Gool, "A comparison of affine region detectors," *International Journal of Computer Vision*, vol. 65, no. 1-2, pp. 43–72, 2005.
- [32] C. Schmid, R. Mohr, and C. Bauckhage, "Evaluation of interest point detectors," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 151–172, 2000.
- [33] M. Brown and D. Lowe, "Automatic panoramic image stitching using invariant features," *International Journal of Computer Vision*, vol. 74, pp. 59–73, 2007.
- [34] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 5, pp. 530–535, 1997.
- [35] G. Yang, C. Stewart, M. Sofka, and C. Tsai, "Registration of challenging image pairs: Initialization, estimation, and decision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 11, pp. 1973–1989, 2007.
- [36] S. Lazebnik, C. Schmid, and J. Ponce, "Affine-invariant local descriptors and neighborhood statistics for texture recognition," in *IEEE International Conference on Computer Vision*, vol. 1, pp. 649–655, 2003.

- [37] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman, "Learning object categories from google's image search," in *IEEE International Conference on Computer Vision*, pp. 1816–1823, 2005.
- [38] D. G. Lowe, "Object recognition from local scale-invariant features," in *IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, 1999.
- [39] H. Riemenschneider, M. Donoser, and H. Bischof, "Online object recognition by mser trajectories," in *International Conference on Pattern Recognition*, pp. 1–4, 2008.
- [40] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. II – 272–7, june 2003.
- [41] J. Sivic, F. Schaffalitzky, and A. Zisserman, "Object level grouping for video shots," *International Journal of Computer Vision*, vol. 67, no. 2, pp. 189–210, 2006.
- [42] C. Geng and X. Jiang, "Face recognition based on the multi-scale local image structures," *Pattern Recognition*, vol. 44, no. 10-11, pp. 2565–2575, 2011.
- [43] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conference*, pp. 147–151, 1988.
- [44] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *International Journal of Computer Vision*, vol. 60, no. 1, pp. 63–86, 2004.
- [45] S. M. Smith and J. M. Brady, "SUSAN - a new approach to low level image processing," *International Journal of Computer Vision*, vol. 23, no. 1, pp. 45–78, 1997.
- [46] X. Zhang, H. Wang, A. Smith, X. Ling, B. Lovell, and D. Yang, "Corner detection based on gradient correlation matrices of planar curves," *Pattern Recognition*, vol. 43, no. 4, pp. 1207–1223, 2010.
- [47] P. Beaudet, "Rotationally invariant image operators," in *International Joint Conference on Pattern Recognition*, pp. 579–583, 1978.
- [48] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
- [49] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (SURF)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.
- [50] T. Kadir and M. Brady, "Saliency, scale and image description," *International Journal of Computer Vision*, vol. 45, no. 2, pp. 83–105, 2001.

- [51] W. T. Lee and H. T. Chen, "Histogram-based interest point detectors," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1590–1596, 2009.
- [52] T. Tuytelaars and L. Van Gool, "Matching widely separated views based on affine invariant regions," *International Journal of Computer Vision*, vol. 59, no. 1, pp. 61–85, 2004.
- [53] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761–767, 2004.
- [54] Q. Li, J. Ye, and C. Kambhamettu, "Interest point detection using imbalance oriented selection," *Pattern Recognition*, vol. 41, no. 2, pp. 672–688, 2008.
- [55] B. Josef, *Vision with Direction: A Systematic Introduction to Image Processing and Computer Vision*. 2006.
- [56] G. Dorkó and C. Schmid, "Maximally stable local description for scale selection," in *European Conference on Computer Vision*, pp. 504–516, 2006.
- [57] T. Lindeberg, "Detecting salient blob-like image structures and their scales with a scale-space primal sketch - a method for focus-of-attention," *International Journal of Computer Vision*, vol. 11, no. 3, pp. 283–318, 1993.
- [58] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation theory*. Prentice-Hall PTR, 1993.
- [59] Z. W. Miao and X. D. Jiang, "Further properties and a fast realization of the iterative truncated arithmetic mean filter," *IEEE Transactions on Circuits and Systems Part II: Express Briefs*, vol. 59, pp. 810–814, November 2012.
- [60] P. J. Huber, *Robust Statistics*. John Wiley, 2005.
- [61] S. Kalluri and G. R. Arce, "Fast algorithms for weighted myriad computation by fixed-point search," *IEEE Transactions on Signal Processing*, vol. 48, no. 1, pp. 159–171, 2000.
- [62] P. Perona and J. Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, 1990.
- [63] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1-4, pp. 259 – 268, 1992.

- [64] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *International Conference on Computer Vision*, pp. 839–846, 1998.
- [65] A. Buades, B. Coll, and J. M. Morel, "Nonlocal image and movie denoising," *International Journal of Computer Vision*, vol. 76, no. 2, pp. 123–139, 2008.
- [66] M. Hans Peter, *Obstacle avoidance and navigation in the real world by a seeing robot rover*. PhD thesis, Stanford University, 1980. 909315.
- [67] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," *Image and Vision Computing*, vol. 3951, pp. 430–443, 2006.
- [68] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105–119, 2010.
- [69] K. Paler, J. Fölein, J. Illingworth, and J. Kittler, "Local ordered grey levels as an aid to corner detection," *Pattern Recognition*, vol. 17, no. 5, pp. 535–543, 1984.
- [70] J. A. Noble, "Finding corners," *Image Vision Comput.*, vol. 6, no. 2, pp. 121–128, 1988.
- [71] T. Lindeberg, "Direct estimation of affine image deformations using visual front-end operations with automatic scale selection," in *International Conference on Computer Vision*, pp. 134–141, June 1995.
- [72] E. Mair, G. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," in *European Conference on Computer Vision*, vol. 6312, pp. 183–196, Springer Berlin Heidelberg, 2010.
- [73] T. Kadir, A. Zisserman, and M. Brady, "An affine invariant salient region detector," in *European Conference on Computer Vision*, vol. 3021, pp. 228–241, 2004.
- [74] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, vol. 30, no. 2, pp. 79–116, 1998.
- [75] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide baseline stereo from maximally stable extremal regions," in *British Machine Vision Conference*, vol. 1, pp. 384–393, 2002.
- [76] T. Tuytelaars and L. V. Gool, "Wide baseline stereo matching based on local, affinity invariant regions," in *British Machine Vision Conference*, pp. 412–425, 2000.
- [77] F. Mokhtarian and F. Mohanna, "Performance evaluation of corner detectors using consistency and accuracy measures," *Computer Vision and Image Understanding*, vol. 102, no. 1, pp. 81–94, 2006.

- [78] P. M. Narendra, "A separable median filter for image noise smoothing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 3, no. 1, pp. 20–29, 1981.
- [79] T. H. Li and K. S. Song, "Estimation of the parameters of sinusoidal signals in non-Gaussian noise," *IEEE Transactions on Signal Processing*, vol. 57, no. 1, pp. 62–72, 2009.
- [80] O. J. Karst and H. Polowy, "Sampling properties of the median of a Laplace distribution," *The American Mathematical Monthly*, vol. 70, no. 6, pp. 628–636, 1963.
- [81] E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms*. Potomac, MD: Computer Science Press: W. H. Freeman & Co. New York, 1978.
- [82] A. Burian and P. Kuosmanen, "Tuning the smoothness of the recursive median filter," *IEEE Transactions on Signal Processing*, vol. 50, no. 7, pp. 1631–1639, 2002.
- [83] G. R. Arce and J. L. Paredes, "Nonlinear image processing," ch. Image enhancement and analysis with weighted medians, New York: Academic, 2000.
- [84] G. R. Arce, "A general weighted median filter structure admitting negative weights," *IEEE Transactions on Signal Processing*, vol. 46, no. 12, pp. 3195–3205, 1998.
- [85] G. R. Arce, N. A. Grabowski, and N. C. Gallagher, "Weighted median filters with sigma-delta modulation encoding," *IEEE Transactions on Signal Processing*, vol. 48, no. 2, pp. 489–498, 2000.
- [86] G. R. Arce and J. L. Paredes, "Recursive weighted median filters admitting negative weights and their optimization," *IEEE Transactions on Signal Processing*, vol. 48, no. 3, pp. 768–779, 2000.
- [87] T. C. Aysal and K. E. Barner, "Second-order heavy-tailed distributions and tail analysis," *IEEE Transactions on Signal Processing*, vol. 54, no. 7, pp. 2827–2832, 2006.
- [88] Y. Q. Dong and S. F. Xu, "A new directional weighted median filter for removal of random-valued impulse noise," *IEEE Signal Processing Letters*, vol. 14, no. 3, pp. 193–196, 2007.
- [89] M. Fischer, J. L. Paredes, and G. R. Arce, "Weighted median image sharpeners for the world wide web," *IEEE Transactions on Image Processing*, vol. 11, no. 7, pp. 717–727, 2002.
- [90] T. C. Aysal and K. E. Barner, "Quadratic weighted median filters for edge enhancement of noisy images," *IEEE Transactions on Image Processing*, vol. 15, no. 11, pp. 3294–3310, 2006.

- [91] K. E. Barner and T. C. Aysal, "Polynomial weighted median filtering," *IEEE Transactions on Signal Processing*, vol. 54, no. 2, pp. 636–650, 2006.
- [92] T. C. Aysal and K. E. Barner, "Hybrid polynomial filters for gaussian and non-gaussian noise environments," *IEEE Transactions on Signal Processing*, vol. 54, no. 12, pp. 4644–4661, 2006.
- [93] Y. B. Li, G. R. Arce, and J. Bacca, "Weighted median filters for multichannel signals," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4271–4281, 2006.
- [94] I. Shmulevich and G. R. Arce, "Spectral design of weighted median filters admitting negative weights," *IEEE Signal Processing Letters*, vol. 8, no. 12, pp. 313–316, 2001.
- [95] S. Hoyos, J. Bacca, and G. R. Arce, "Spectral design of weighted median filters: a general iterative approach," *IEEE Transactions on Signal Processing*, vol. 53, no. 3, pp. 1045–1056, 2005.
- [96] S. Kalluri and G. R. Arce, "Adaptive weighted myriad filter algorithms for robust signal processing in α -stable noise environments," *IEEE Transactions on Signal Processing*, vol. 46, no. 2, pp. 322–334, 1998.
- [97] S. Kalluri and G. R. Arce, "A general class of nonlinear normalized adaptive filtering algorithms," *IEEE Transactions on Signal Processing*, vol. 47, no. 8, pp. 2262–2272, 1999.
- [98] S. Kalluri and G. R. Arce, "Robust frequency-selective filtering using weighted myriad filters admitting real-valued weights," *IEEE Transactions on Signal Processing*, vol. 49, no. 11, pp. 2721–2733, 2001.
- [99] R. C. Nunez, J. G. Gonzalez, G. R. Arce, and J. P. Nolan, "Fast and accurate computation of the myriad filter via branch-and-bound search," *IEEE Transactions on Signal Processing*, vol. 56, no. 7, pp. 3340–3346, 2008.
- [100] K. S. Choi, A. W. Morales, and S. J. Ko, "Design of linear combination of weighted medians," *IEEE Transactions on Signal Processing*, vol. 49, no. 9, pp. 1940–1952, 2001.
- [101] O. Yli-Harja, J. Astola, and Y. Neuvo, "Analysis of the properties of median and weighted median filters using threshold logic and stack filter representation," *IEEE Transactions on Signal Processing*, vol. 39, no. 2, pp. 395–410, 1991.
- [102] C.-H. Lin, J.-S. Tsai, and C.-T. Chiu, "Switching bilateral filter with a texture/noise detector for universal noise removal," *IEEE Transactions on Image Processing*, vol. 19, no. 9, pp. 2307–2320, 2010.

- [103] Z. W. Miao and X. D. Jiang, "A novel rank order LoG filter for interest point detection," in *IEEE Conference on Acoustics, Speech and Signal Processing*, pp. 937–940, 2012.
- [104] Z. W. Miao and X. D. Jiang, "Interest point detection using rank order LoG filter," *Pattern Recognition*, vol. 46, pp. 2890–2901, November 2013.
- [105] Z. W. Miao and X. D. Jiang, "Weighted iterative truncated mean filter," *IEEE Transactions on Signal Processing*, vol. 61, pp. 4149–4160, August 2013.
- [106] G. Grimmett and D. Stirzaker, *Probability and Random Processes*. Oxford University: Oxford, Inglaterra, 2001.
- [107] J. F. Ren and X. D. Jiang, "Eye detection based on rank order filter," in *International Conference on Information, Communications and Signal Processing*, pp. 1–4, 2009.
- [108] X. D. Jiang, "Asymmetric principal component and discriminant analyses for pattern classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 931–937, 2009.
- [109] X. D. Jiang, B. Mandal, and A. Kot, "Eigenfeature regularization and extraction in face recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, no. 3, pp. 383–394, 2008.
- [110] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.
- [111] X. Jiang, "Linear subspace learning-based dimensionality reduction," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 16–26, 2011.
- [112] H. Yu and H. Yang, "A direct LDA algorithm for high-dimensional data - with application to face recognition," *Pattern Recognition*, vol. 34, no. 10, pp. 2067–2070, 2001.
- [113] A. M. Martinez, "Recognizing imprecisely localized, partially occluded, and expression variant faces from a single sample per class," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 748–763, 2002.
- [114] F. S. Samaria and A. C. Harter, "Parameterisation of a stochastic model for human face identification," in *IEEE Workshop on Applications of Computer Vision*, pp. 138–142, 1994.
- [115] "Georgia tech face database." http://www.anefian.com/face_reco.htm, 2007.

- [116] P. Phillips, H. Moon, S. Rizvi, and P. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1090 – 1104, 2000.
- [117] G. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Tech. Rep. 07-49, University of Massachusetts, Amherst, 2007.
- [118] D. L. Page, A. Koschan, Y. Sun, J. Paik, and M. A. Abidi, "Robust crease detection and curvature estimation of piecewise smooth surfaces from triangle mesh approximations using normal voting," in *Proc. Conf. Computer Vision and Pattern Recognition*, vol. 1, pp. 162–167, 2001.
- [119] Z. F. Wang and Z. G. Zheng, "A region based stereo matching algorithm using cooperative optimization," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.
- [120] M. Brown and D. Lowe, "Recognising panoramas," in *IEEE International Conference on Computer Vision*, pp. 1218 –1225, 2003.
- [121] R. Unnikrishnan and M. Hebert, "Extracting scale and illuminant invariant regions through color," in *British Machine Vision Conference*, 2006.
- [122] M. Loog and F. Lauze, "The improbability of harris interest points," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1141–1147, 2010.
- [123] R. Kimmel, C. P. Zhang, A. M. Bronstein, and M. M. Bronstein, "Are mser features really interesting?," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 11, pp. 2316–2320, 2011.
- [124] P. E. Forssen, "Maximally stable colour regions for recognition and matching," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.
- [125] J. Kim and K. Grauman, "Boundary preserving dense local regions," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1553–1560, 2011.
- [126] J. Bigun, T. Bigun, and K. Nilsson, "Recognition by symmetry derivatives and the generalized structure tensor," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 12, pp. 1590 –1605, 2004.
- [127] C. Geng and X. D. Jiang, "Face recognition using sift features," in *IEEE International Conference on Image Processing*, pp. 3313–3316, 2009.