



TESIS

# Noise Filtering Dengan Soft Weighted Median Filter untuk Meningkatkan Kualitas Segmentasi Citra

Siprianus Septian Manek  
NRP. 5116201039

DOSEN PEMBIMBING  
Prof. Ir. Handayani Tjandrasa, M.Sc Ph.D  
NIP: 19490823 197603 2 001

PROGRAM MAGISTER  
RUMPUN MATA KULIAH KOMPUTASI CERDAS DAN VISUALISASI  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
SURABAYA  
2017

*[Halaman ini sengaja dikosongkan]*

Tesis disusun untuk memenuhi salah satu syarat memperoleh gelar  
Magister Komputer (M.Kom.)

di

Institut Teknologi Sepuluh Nopember Surabaya

oleh:

Siprianus Septian Manek  
NRP. 5116201039

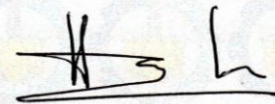
Dengan judul:

NOISE FILTERING DENGAN SOFT WEIGHTED MEDIAN FILTER UNTUK  
MENINGKATKAN KUALITAS SEGMENTASI CITRA

Tanggal ujian: 11-1-2018  
Periode wisuda: 2018 Gasal

Disetujui oleh:

Prof. Ir. Handayani Tjandrasa, M.Sc Ph.D.  
NIP. 19490823 197603 2 001



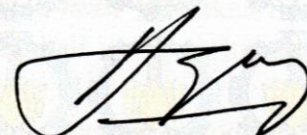
(Pembimbing 1)

Dr. Eng. Chastine Fatchah, S.Kom, M.Kom  
NIP. 19751220 200112 2 002



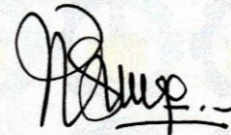
(Penguji 1)

Dr. Agus Zainal Arifin, S.Kom, M.Kom  
NIP. 19720809 199512 1 001



(Penguji 2)

Dr. Eng. Nanik Suciati, S.Kom, M.Kom  
NIP. 19710428 199412 2 001



(Penguji 3)



Dekan Fakultas Teknologi Informasi dan Komunikasi,

Dr. Agus Zainal Arifin, S.Kom, M.Kom  
NIP. 19720809 199512 1 001

# ***Noise Filtering dengan Soft Weighted Median Filter untuk Meningkatkan Kualitas Segmentasi Citra***

Nama : Siprianus Septian Manek  
NRP : 5116201039  
Pembimbing I : Handayani Tjandrasa

## **ABSTRAK**

Salah satu faktor penghambat pada proses pengolahan citra adalah *noise*. *Noise* pada citra dibedakan menjadi dua jenis yaitu *fixed-valued noise* (*salt & pepper noise*) dan *random-valued noise* (*gaussian, poisson, speckle, dan locarvar noise*). Penelitian-penelitian sebelumnya yang terkait dengan *noise filtering* lebih fokus pada *fixed-valued noise*, sedangkan untuk *random-valued noise* masih jarang dilakukan.

Penelitian ini mengusulkan metode *Soft Weighted Median Filter* (SWMF) untuk menghilangkan *fixed-valued* maupun *random-valued noise*. Untuk setiap piksel pada citra, langkah pertama yang dilakukan adalah menentukan *window* 3×3 untuk mencari piksel *center* dan piksel tetangganya. Kemudian semua nilai piksel pada *window* tersebut diurutkan dan dibagi menjadi tiga bagian, jika nilai piksel *center* berada pada bagian kedua, maka dianggap sebagai piksel bebas *noise*, sedangkan jika nilai piksel *center* berada pada bagian pertama atau bagian ketiga, maka dianggap sebagai piksel ber-*noise*. Langkah terakhir pada proses ini adalah mengganti nilai piksel ber-*noise* dengan nilai rata-rata *median* tertimbang dari semua piksel dalam *window*, sedangkan piksel bukan *noise* dibiarkan tidak berubah. Nilai piksel baru dari proses ini digunakan kembali untuk perhitungan berikutnya.

Citra hasil dari metode SWMF dibandingkan dengan metode-metode yang lain seperti; *Median Filter*, *Mean Filter*, *Wiener Filter* dan *Gaussian Filter* lewat pengukuran *Mean Squared Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR). Proses segmentasi citra dilakukan pada citra hasil *noise filtering*, terdiri dari 2 proses yaitu deteksi area (*Top-Hat Transform*) dan deteksi garis (*Sobel Edge Detection*). Analisis kinerja pada tahap ini menggunakan perhitungan *sensitivity*, *specificity*, dan *accuracy* antara citra *groundtruth* dengan citra hasil segmentasi.

Berdasarkan hasil uji coba, dapat disimpulkan bahwa metode *Soft Weighted Median Filter* berhasil meningkatkan kualitas segmentasi citra dengan cara menghilangkan *fixed-valued* maupun *random-valued noise*, metode ini memiliki rata-rata nilai PSNR paling tinggi dibandingkan metode lainnya yaitu sebesar 29,21 db.

**Kata kunci :** *median filter, noise filtering*, segmentasi citra.

# Noise Filtering with Soft Weighted Median Filter to Improve Image Segmentation Quality

Name : Siprianus Septian Manek  
Student Identity Number : 5116201039  
Supervisor : Handayani Tjandrasa

## ABSTRACT

One of the inhibiting factors in image digital processing is noise. Noise in the image is divided into two types: fixed-valued noise (salt & pepper noise) and random-valued noise (gaussian, poisson, speckle, and locarvar noise). Previous studies of noise filtering focus on fixed-valued noise, while random-valued noise is rarely done.

This research proposes a Soft Weighted Median Filter (SWMF) method to remove fixed-valued dan random-valued noise. For each pixel in the image, the first step is determine the  $3 \times 3$  *window* to search the center pixel and neighboring pixels. Then all pixel values in the *window* are sorted and divided into three parts, if the the center pixel value in the part two, it is considered as noise-free pixel, whereas if the center pixel value in part one or part three, it is considered as noise pixel. The final step in this process is replace the noise-pixel value with the average of median weighted value of all pixels in the *window*, while the noise-free pixel are left unchanged. The new pixel value from this process is reused for the next pixel calculation.

The result images of the SWMF method are compared with other methods such as: Median Filter, Mean Filter, Wiener Filter and Gaussian Filter with the measurement of Mean Squared Error (MSE) and Peak Signal to Noise Ratio (PSNR). Image segmentation process is done on the image of noise filtering result. There are two image segmentation process, firstly, area detection using Top-Hat Transform, and secondly, line detection using Sobel Edge Detection. Performance analysis at this stage using the calculation of sensitivity, specificity, and accuracy between groundtruth images with the image of the results of segmentation.

Based on the experiment results, Soft Weighted Median Filter method succeeded to improve the quality of image segmentation by eliminating fixed-valued and random-valued noise. This method has the highest average PSNR value compared to other methods of 29,21 db.

**Keywords** : median filter, noise filtering, image segmentation.

*[Halaman ini sengaja dikosongkan]*

## KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas segala berkat dan kasih karunia-Nya sehingga tesis dengan judul “Noise Filtering dengan Soft Weighted Median Filter untuk Meningkatkan Kualitas Segmentasi Citra” ini dapat diselesaikan.

Tesis ini disusun untuk memenuhi salah satu persyaratan memperoleh gelar Magister Komputer (M.Kom.) dalam bidang keahlian Komputasi Cerdas dan Visualisasi pada Program Studi Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Oleh karena itu, pada kesempatan ini penulis menyampaikan rasa hormat dan menghaturkan terima kasih yang sebesar-besarnya, kepada:

1. Ibu Prof. Ir. Handayani Tjandrasa, M.Sc Ph.D. yang telah meluangkan waktu untuk memberikan bimbingan, petunjuk, dan mendorong semangat penulis untuk menyelesaikan penulisan Tesis ini.
2. Bapak Dr. Agus Zainal Arifin, S.Kom, M.Kom., selaku Dekan Fakultas Teknologi Informasi dan Komunikasi, Dosen Wali, dan juga Dosen Penguji 2 yang telah membimbing, memberikan arahan dan waktu kepada penulis untuk berdiskusi.
3. Ibu Dr. Eng. Chastine Faticah, S.Kom, M.Kom, dan Ibu Dr. Eng. Nanik Suciati, S.Kom, M.Kom., selaku dosen penguji 1 dan 3 yang telah memberikan masukan dan saran pada saat seminar proposal dan seminar hasil tesis.
4. Ketua program studi Pascasarjana Teknik Informatika Bapak Waskitho Wibisono, S.Kom, M.Eng., Ph.D.
5. Seluruh Dosen dan Staf Program Pascasarjana Teknik Informatika yang telah memberikan arahan dan bimbingan selama penulis menjalani studi S2.
6. Orang Tua tercinta yang telah mendidik dengan penuh rasa kasih sayang dan senantiasa memberi semangat dan dorongan kepada penulis.
7. Rekan-rekan seperjuangan S2 Teknik Informatika angkatan 2016, khususnya saudara-saudari dari Timur Indonesia, Kupang-NTT, terima kasih untuk rasa persaudaraan dan kebersamaannya.



8. Kepada semua pihak yang telah membantu yang tidak dapat penulis sebutkan satu persatu.

Dengan keterbatasan pengalaman, ilmu maupun pustaka yang ditinjau, penulis menyadari bahwa tesis ini masih banyak kekurangan dan pengembangan lanjut agar benar benar bermanfaat. Oleh sebab itu, penulis sangat mengharapkan kritik dan saran agar tesis ini lebih sempurna serta sebagai masukan bagi penulis untuk penelitian dan penulisan karya ilmiah di masa yang akan datang.

Akhir kata, penulis berharap tesis ini memberikan manfaat bagi kita semua terutama untuk pengembangan ilmu pengetahuan komputer.

Surabaya, 18 Januari 2018

Penulis,

Siprianus Septian Manek

## DAFTAR ISI

<b>JUDUL</b> .....	i
<b>LEMBAR PENGESAHAN TESIS</b> .....	iii
<b>ABSTRAK</b> .....	v
<b>ABSTRACT</b> .....	vii
<b>KATA PENGANTAR</b> .....	ix
<b>DAFTAR ISI</b> .....	xi
<b>DAFTAR GAMBAR</b> .....	xiv
<b>DAFTAR TABEL</b> .....	xvi
<b>BAB 1. PENDAHULUAN</b> .....	1
1.1. Latar Belakang.....	1
1.2. Tujuan Penelitian .....	2
1.3. Perumusan Masalah .....	2
1.4. Batasan Masalah .....	3
1.5. Manfaat Penelitian .....	3
1.6. Kontribusi Penelitian.....	3
1.7. Sistematika Penulisan .....	4
<b>BAB 2. KAJIAN PUSTAKA</b> .....	5
2.1. Citra .....	5
2.1.1. Pengolahan Citra.....	5
2.1.2. <i>Window</i> .....	6
2.1.3. Histogram .....	7
2.2. <i>Noise</i> .....	7
2.2.1. <i>Noise Salt &amp; Pepper</i> .....	7
2.2.2. <i>Noise Gaussian</i> .....	8
2.2.3. <i>Noise Poisson</i> .....	8
2.2.4. <i>Noise Speckle</i> .....	9
2.2.5. <i>Noise Localvar</i> .....	9
2.3. <i>Noise Filtering</i> .....	10
2.3.1. <i>Median Filter</i> .....	11

2.3.2. <i>Mean Filter</i> .....	11
2.3.3. <i>New Adaptive Weighted Mean Filter</i> .....	12
2.3.4. <i>Weighted Mean Filter for High Density Noise</i> .....	12
2.3.5. <i>Unbiased Weighted Mean Filter</i> .....	13
2.3.6. <i>Interpolation based Noise Removing</i> .....	13
2.3.7. <i>Wiener Filter</i> .....	13
2.4. Segmentasi Citra .....	14
2.4.1. Teknik Morfologi .....	15
2.4.2. Dilasi dan Erosi .....	16
2.4.3. <i>Opening dan Closing</i> .....	16
2.4.4. Transformasi Top-Hat .....	17
2.4.5. <i>Otsu Thresholding</i> .....	18
2.4.6. Metode Sobel.....	20
<b>BAB 3. METODOLOGI PENELITIAN</b> .....	21
3.1. Tahap Penambahan <i>Noise</i> .....	22
3.2. Tahap <i>Noise Filtering</i> .....	22
3.3. <i>Soft Weighted Median Filter</i> .....	23
3.3.1. Deteksi <i>Noise</i> .....	24
3.3.2. Reduksi <i>Noise</i> .....	25
3.4. Tahap Segmentasi .....	27
3.4.1. Deteksi Area.....	27
3.4.2. Deteksi Garis .....	28
3.5. Tahap Pengujian .....	30
3.5.1. Pengujian <i>Noise Filtering</i> .....	30
3.5.2. Pengujian Segmentasi .....	31
<b>BAB 4. HASIL DAN PEMBAHASAN</b> .....	33
4.1. Lingkungan dan Data Uji Coba.....	33
4.2. Pelaksanaan dan Hasil Uji Coba .....	35
4.2.1. Pelaksanaan dan Hasil Penambahan <i>Noise</i> .....	35
4.2.2. Pelaksanaan dan Hasil <i>Noise Filtering</i> .....	37
4.2.3. Pelaksanaan dan Hasil Deteksi Area.....	41
4.2.4. Pelaksanaan dan Hasil Deteksi Garis .....	46

4.3. Evaluasi Hasil Uji Coba.....	53
4.3.1. Evaluasi Hasil <i>Noise Filtering</i> .....	53
4.3.2. Evaluasi Hasil Deteksi Area .....	53
4.3.3. Evaluasi Hasil Deteksi Garis .....	54
<b>BAB 5. PENUTUP</b> .....	<b>55</b>
5.1. Kesimpulan.....	55
5.2. Saran .....	56
<b>DAFTAR PUSTAKA</b> .....	<b>57</b>

## DAFTAR GAMBAR

Gambar 2.1. Proses pengolahan citra .....	5
Gambar 2.2. <i>Window</i> 3×3 pada citra .....	6
Gambar 2.3. <i>Local histogram window</i> 3×3 pada citra asli .....	7
Gambar 2.4. <i>Local histogram window</i> 3×3 pada citra dengan <i>noise SP</i> .....	8
Gambar 2.5. <i>Local histogram window</i> 3×3 pada citra dengan <i>noise gaussian</i> .....	8
Gambar 2.6. <i>Local histogram window</i> 3×3 pada citra dengan <i>noise poisson</i> ..	9
Gambar 2.7. <i>Local histogram window</i> 3×3 pada citra dengan <i>noise speckle</i> ..	9
Gambar 2.8. <i>Local histogram window</i> 3×3 pada citra dengan <i>noise localvar</i> ..	10
Gambar 2.9. <i>Median filter</i> .....	11
Gambar 2.10. Struktur elemen .....	15
Gambar 2.11. Dilasi dan Erosi .....	16
Gambar 2.12. Contoh Transformasi Top-Hat .....	18
Gambar 3.1. Prosedur penelitian .....	21
Gambar 3.2. Tahap-tahap penambahan <i>noise</i> .....	22
Gambar 3.3. Tahap-tahap <i>noise filtering</i> .....	23
Gambar 3.4. Ilustrasi proses pembobotan nilai piksel .....	25
Gambar 3.5. Proses SWMF.....	26
Gambar 3.6. Tahap-tahap deteksi area .....	28
Gambar 3.7. Tahap-tahap deteksi garis.....	29
Gambar 3.8. <i>Confusion matriks</i> .....	32
Gambar 4.1. Citra Uji Coba .....	34
Gambar 4.2. Hasil Penambahan Noise pada Citra Uji Coba .....	36
Gambar 4.3. Rata-Rata Keseluruhan PSNR.....	41
Gambar 4.4. Rata-Rata PSNR berdasarkan Jenis Noise .....	41
Gambar 4.5. Tahapan Proses Pengujian untuk Deteksi Area.....	42
Gambar 4.6. Hasil Filtering Noise Localvar pada Citra Rice .....	43
Gambar 4.7. Hasil Deteksi Area pada Citra Rice dengan Noise Localvar .....	44
Gambar 4.8. Rata-Rata Sensitivity, Specificity, dan Accuracy .....	45
Gambar 4.9. Tahapan Proses Pengujian untuk Deteksi Garis.....	47

Gambar 4.10. Hasil Filtering Noise Salt&Pepper pada Citra Coins .....	48
Gambar 4.11. Hasil Deteksi Garis pada Citra Coins dengan Noise SP.....	48
Gambar 4.12. Hasil Filtering Noise Gaussian pada Citra Lighting.....	49
Gambar 4.13. Hasil Deteksi Garis pada Citra Lighting dengan Gaussian .....	49
Gambar 4.14. Hasil Filtering Noise Poisson pada Citra Eyeglass.....	50
Gambar 4.15. Hasil Deteksi Garis pada Citra Eyeglas dengan Noise Poisson	50
Gambar 4.16. Hasil Filtering Noise Speckle pada Citra Woodtoys .....	51
Gambar 4.17. Hasil Deteksi Garis Citra Woodtoys dengan Noise Speckle .....	51
Gambar 4.18. Hasil Filtering Noise Localvar pada Citra Shapessm .....	52
Gambar 4.19. Hasil Deteksi Garis Citra Shapessm dengan Noise Localvar....	52

## DAFTAR TABEL

Tabel 4.1. Hasil Pengukuran PSNR (db) .....	39
Tabel 4.1. Hasil Pengukuran PSNR (lanjutan)(db).....	40
Tabel 4.2. Hasil Pengukuran Sensitivity, Specificity, dan Accuracy.....	45





# BAB 1

## PENDAHULUAN

### 1.1. Latar Belakang

Citra digital sebagai salah satu komponen multimedia memegang peranan yang sangat penting sebagai bentuk informasi visual. Sebuah citra memiliki karakteristik yang berbeda dengan teks, ada sebuah peribahasa yang mengatakan, “*a picture is more than a thousand words*“ yang memiliki arti: citra atau gambar lebih bermakna daripada seribu kata dan dapat memberi lebih banyak informasi daripada informasi disajikan dalam bentuk kata (tekstual).

Pada saat ini, pengolahan citra digital merupakan bidang yang berkembang pesat dan banyak diterapkan pada berbagai ilmu pengetahuan. Pengolahan citra didefinisikan sebagai proses pengolahan dan analisis citra yang banyak melibatkan persepsi visual. Proses pengolahan citra mempunyai ciri data masukan dan informasi keluaran berbentuk citra, sehingga pengolahan citra adalah pemrosesan citra yang telah ada untuk menghasilkan citra yang lebih tinggi kualitasnya, dalam arti lebih jelas menampilkan informasi yang diharapkan.

Terdapat beberapa gangguan pada citra yang dapat menghambat proses pengolahan citra. Gangguan tersebut dapat berupa distorsi cahaya, *noise*, maupun gangguan lainnya yang menyebabkan objek dalam citra menjadi kurang jelas ataupun kabur. Untuk mengatasi hal seperti ini, maka diperlukan sebuah teknik perbaikan citra. Salah satu teknik yang dimaksud adalah dengan menghilangkan derau (*noise filtering*) kemudian diikuti dengan proses segmentasi citra. Kedua proses ini tidak dapat dipisahkan dan selalu ada pada tahap awal pengolahan citra, hasil segmentasi citra yang bagus tergantung pada seberapa efektif teknik *noise filtering* yang digunakan (Basuki, 2005).

*Noise filtering* merupakan proses mendeteksi dan menghilangkan derau (*noise*) pada citra. *Noise* pada citra bisa diartikan sebagai titik atau nilai piksel yang sebenarnya bukan merupakan bagian dari citra tersebut dan ikut tercampur karena suatu alasan (Sutoyo, 2009). Secara teknis, *noise* terjadi karena karakteristik dari derajat keabu-abuan (*gray-level*) atau dikarenakan variabel acak dari karakteristik fungsi probabilitas kepadatan (*probability density function*).

Berdasarkan nilai penyusunnya, *noise* pada citra dapat dibagi menjadi dua yaitu; *noise* dengan nilai tetap (*fixed-valued noise*) dan *noise* dengan nilai acak (*random valued noise*) [1-3]. Contoh *fixed-valued noise* seperti; *salt & pepper noise* (SPN), sedangkan contoh *random-valued noise* seperti; *gaussian*, *poisson*, *speckle*, dan *localvar noise*.

Ada banyak penelitian untuk mendeteksi dan menghilangkan *noise* pada citra [1-10]. Namun, hampir semua penelitian tersebut fokus pada *fixed-valued noise* seperti; *New Adaptive Mean Filter* (NAWMF)[1], *Unbiased Weighted Mean Filter* (UWMF)[2], *Interpolation based Image Noise Removing* (IBINR)[3], *Weighted Mean Filter for High Density Noise* (HD-WMF)[4], *Combines Adaptive Median Filter*[5], dan yang lainnya. Jika citra yang akan diproses mengandung *random value noise*, maka metode-metode diatas memberikan hasil yang kurang optimal. Hasil *noise filtering* yang kurang optimal ini tentunya akan berdampak buruk pada kualitas segmentasi citra.

Berdasarkan penjelasan diatas, maka pada penelitian ini diusulkan metode *Soft Weighted Median Filter* (SWMF) untuk *fixed-valued noise* maupun *random-valued noise* guna meningkatkan kualitas segmentasi citra. Metode ini terdiri dari dua tahap yaitu tahap deteksi *noise* dan tahap reduksi *noise*. Proses deteksi *noise* menggunakan pendekatan histogram pada tiap-tiap piksel dalam *window*, kemudian untuk tahap reduksi *noise* menggunakan penggabungan filter median dan mean dengan proses pembobotan yang sederhana.

Citra hasil *noise filtering* dengan metode ini dibandingkan dengan metode-metode yang lain seperti: *Median Filter*, *Mean Filter*, *Wiener Filter*, dan *Gaussian Filter* lewat perhitungan penyimpangan yang terjadi antara citra asli dengan citra setelah *noise filtering* menggunakan parameter PSNR (*Peak Signal to Noise Ratio*) dan MSE (*Mean Squared Error*). Selain itu, untuk menguji kinerja dari metode ini lebih lanjut maka proses pengujian berlanjut ke proses segmentasi menggunakan Transformasi *Top-Hat* (deteksi area) dan metode Sobel (deteksi garis). Hasil segmentasi dibandingkan dengan citra *groundtruth* yang didapat dari segmentasi citra awal (citra sebelum diberi *noise*) lewat pengukuran *sensitivity*, *specificity* dan *accuracy* dari *confusion matriks*.

## 1.2. Tujuan Penelitian

Tujuan dari penelitian ini adalah mengurangi *noise* pada citra (*fixed-valued noise* dan *random-valued noise*) dengan metode *soft weighted median filter* (SWMF) untuk meningkatkan kualitas segmentasi citra.

## 1.3. Perumusan Masalah

Rumusan masalah pada penelitian ini adalah sebagai berikut:

- a. Bagaimana membangun sebuah metode yang dapat mendeteksi dan menghilangkan semua jenis *noise* pada citra.
- b. Bagaimana mengukur kinerja dari metode yang diusulkan dengan metode-metode yang lain.
- c. Bagaimana mengukur kinerja dari metode yang diusulkan dengan metode-metode yang lain pada proses segmentasi citra.

## 1.4. Batasan Masalah

Adapun batasan masalah pada penelitian ini adalah sebagai berikut:

- a. Hanya diterapkan pada objek 2 dimensi (citra digital).
- b. Menggunakan citra grayscale 8 bit dengan format .jpg
- c. Diuji pada citra yang diberi *noise salt & pepper*, *gaussian*, *speckle*, *poisson*, dan *localvar*.
- d. Data yang digunakan berupa gambar dengan satu atau beberapa objek beserta *background* didalamnya.

## 1.5. Manfaat Penelitian

Informasi pada sebuah citra hanya terdapat pada piksel yang tidak terkena *noise*, oleh karena itu diharapkan penelitian ini menghasilkan sebuah metode *filtering* baru yang secara efektif dapat mendeteksi piksel ber-*noise* pada citra dan memfilternya dengan algoritma yang tepat, selain itu diharapkan metode ini dapat digunakan pada aplikasi-aplikasi pengolahan citra untuk menghasilkan hasil segmentasi citra yang bagus.

## 1.6. Kontribusi Penelitian

Kontribusi yang diberikan dari penelitian ini adalah pada prosedur deteksi dan reduksi *noise*. Karena digunakan untuk dua tipe *noise* yang berbeda, maka perlu adanya pembaharuan algoritma untuk mendeteksi dan mereduksi *noise* dari metode-metode sebelumnya.

Kontribusi lain yang diberikan adalah pada penggunaan kembali piksel yang sudah diproses untuk perhitungan piksel berikutnya atau dengan kata lain nilai piksel yang ada pada *window* berikutnya selalu di-upgrade. Hal ini dilakukan untuk memperkecil kemungkinan kesalahan deteksi *noise* pada piksel-piksel berikutnya.

### **1.7. Sistematika Penulisan**

Sistematika penulisan pada penelitian ini adalah sebagai berikut:

#### **Bab 1. Pendahuluan**

Memaparkan tentang latar belakang masalah, rumusan masalah, batasan masalah, kontribusi penelitian, tujuan dan manfaat penelitian.

#### **Bab 2. Kajian Pustaka**

Menjelaskan tentang metode yang digunakan dan teori- teori pendukung dalam penelitian ini.

#### **Bab 3. Metodologi Penelitian**

Memaparkan dan menjelaskan tentang desain sistem serta metode yang akan diterapkan proses *noise filtering* citra.

#### **Bab 4. Hasil dan Pembahasan**

Memaparkan uji coba dan menjelaskan analisis hasil dari kinerja metode yang diusulkan pada tahap *noise filtering* dan segmentasi.

#### **Bab 5. Penutup**

Menguraikan kesimpulan yang diambil berdasarkan hasil dari penelitian yang telah diperoleh serta memberikan saran bagi pengembangan penelitian selanjutnya.

## BAB 2

### KAJIAN PUSTAKA

Pada bab ini, dasar teori dan kajian pustaka akan dijelaskan untuk memberikan gambaran yang jelas yang berkaitan dengan penelitian ini. Beberapa poin utama yang akan dijelaskan pada bab ini yaitu tentang: pengolahan citra, *noise*, *noise filtering*, segmentasi citra, dan pengukuran akhir.

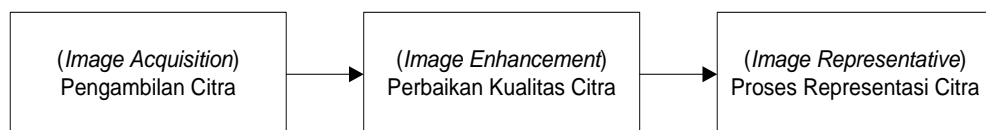
#### 2.1. Citra

Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra terbagi dua yaitu citra yang bersifat analog dan ada citra yang bersifat digital. Citra analog adalah citra yang bersifat *continue* seperti gambar pada monitor televisi, foto sinar X, dan lain-lain. Sedangkan pada citra digital adalah citra yang dapat diolah oleh komputer (Sutoyo, 2009). Gonzales dan Woods (1992) mendefinisikan citra sebagai fungsi  $f(x,y)$  berukuran  $M$  baris dan  $N$  kolom, dengan  $x$  dan  $y$  adalah koordinat spasial, dan amplitudo  $f$  di titik koordinat  $(x,y)$  dinamakan intensitas atau tingkat keabuan dari citra pada citra tersebut.

##### 2.1.1. Pengolahan Citra

Pengolahan citra (*image processing*) merupakan proses mengolah piksel-piksel di dalam citra digital untuk tujuan tertentu. Pada awalnya pengolahan citra ini dilakukan untuk memperbaiki kualitas citra, namun dengan berkembangnya dunia komputasi yang ditandai dengan semakin meningkatnya kapasitas dan kecepatan proses komputer serta munculnya ilmu-ilmu komputasi, maka memungkinkan manusia dapat mengambil informasi dari citra tersebut.

Tahapan proses pengolahan citra yang paling umum dimulai dari pengambilan citra, perbaikan kualitas citra, sampai dengan pernyataan representatif citra. Proses tersebut dapat dilihat pada Gambar 2.1.

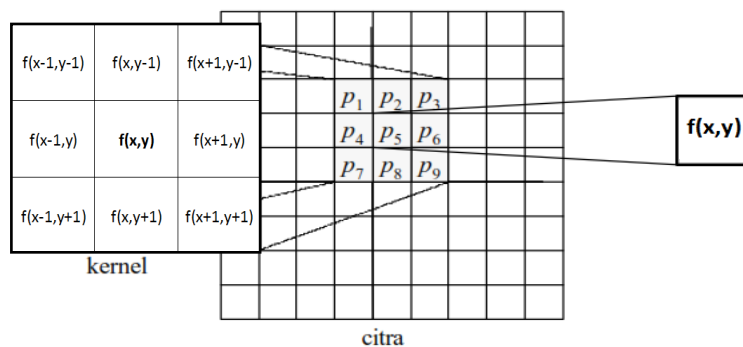


**Gambar 2.1. Proses pengolahan citra**

Dalam perkembangan lebih lanjut, *image processing* dan *computer vision* digunakan sebagai mata manusia, dengan perangkat *input image capture* seperti kamera dan *scanner* dijadikan sebagai mata dan mesin komputer (dengan program komputasinya) dijadikan sebagai otak yang mengolah informasi. Sehingga muncul beberapa pecahan bidang yang menjadi penting dalam *computer vision*, antara lain: *pattern recognition* (pengenalan pola), *biometric* pengenalan identifikasi manusia berdasarkan ciri-ciri biologis yang tampak pada badan manusia), *content based image and video retrieval* (mendapatkan kembali citra atau video dengan informasi tertentu), video editing, dan lain-lain.

### 2.1.2. Window

*Window* atau blok-piksel merupakan satu jendela ketetangaan yang mendeskripsikan satu wilayah tertentu pada citra dengan luas sesuai ukuran dari *window* tersebut. Di dalam sebuah *window* terdapat satu nilai piksel yang diproses beserta nilai piksel tetangganya. Ukuran sebuah *window* dapat ditentukan secara manual, semakin besar ukuran *window* maka semakin terlihat efek yang diberikan (Darma, 2010). Selain itu, ukuran *window* umumnya bernilai ganjil, hal ini agar nilai piksel pada pusat *window* tersebut dapat mudah ditentukan.

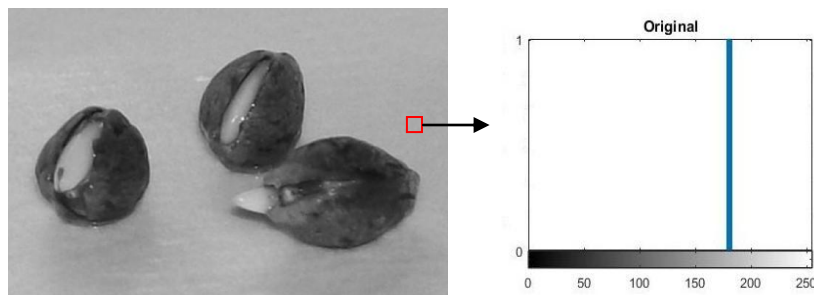


**Gambar 2.2. Window 3×3 pada citra**

Pada Gambar 2.2, dapat dilihat bahwa nilai piksel pusat *window* tersebut adalah  $f(x,y)$  dimana merupakan nilai piksel yang akan diproses, serta mempunyai 8 piksel tetangga. *Window* tersebut diletakan dari piksel pertama sebagai pusat *window*  $f(x,y)$ , kemudian digeser sampai pada piksel terakhir pada citra. Ukuran *window* yang digunakan pada penelitian ini adalah 3×3.

### 2.1.3. Histogram

Histogram citra merupakan diagram yang menggambarkan distribusi frekuensi nilai intensitas warna dalam suatu citra. Sumbu horizontal merupakan nilai intensitas warna sedangkan sumbu vertikal merupakan frekuensi atau jumlah piksel. *Global histogram* melakukan perhitungan pada semua nilai piksel dalam satu citra, sedangkan *local histogram* hanya pada satu area tertentu. Penelitian ini menggunakan histogram pada *window 3×3* (*local histogram*) untuk mengetahui karakteristik dari setiap citra (citra dengan *noise* maupun citra tanpa *noise*).



**Gambar 2.3. Local histogram window 3×3 pada citra asli**

Contoh sebuah *window 3×3* adalah area yang diberi warna merah, dijalankan pada sebuah citra untuk mengoperasikan nilai piksel didalamnya. Dalam *window* tersebut terdapat 9 nilai piksel, sehingga jika dibuat ke dalam histogram maka pada sumbu horizontal terdapat 9 nilai dengan sumbu vertikal (intensitas) adalah 1. Jika dalam *window* tersebut terdapat 2 nilai yang sama maka nilai tertinggi pada sumbu vertikal adalah 2, dan seterusnya.

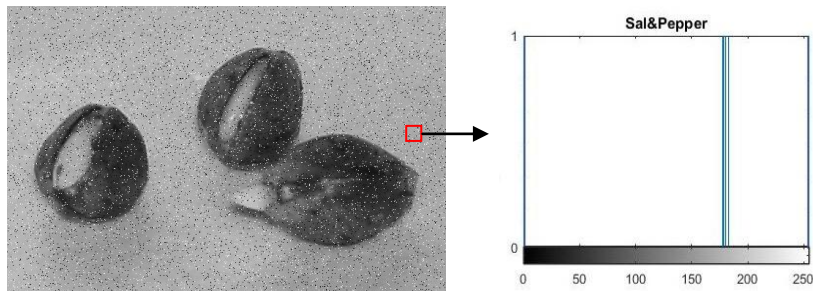
## 2.2. Noise

*Noise* adalah citra atau gambar atau piksel yang mengganggu kualitas citra. *Noise* dapat disebabkan oleh gangguan fisis (*optik*) pada alat akuisisi maupun secara disengaja akibat proses pengolahan yang tidak sesuai, selain itu *noise* juga dapat disebabkan oleh kotoran-kotoran yang terjadi pada citra. Terdapat beberapa jenis *noise* sesuai dengan bentuk dan karakteristiknya, yaitu *salt & pepper*, *gaussian*, *poisson*, *speckle*, dan *noise localvar*.

### 2.2.1. Noise Salt & Pepper

Seperti namanya *noise* jenis ini terlihat seperti *salt and pepper*. Pada citra *grayscale*, *noise* ini akan muncul dalam dua warna yakni hitam (*black*)

dan putih (*white*), sedangkan untuk citra berwarna, titik-titik muncul dalam tiga warna yakni merah (*red*), hijau (*green*) dan biru (*blue*)

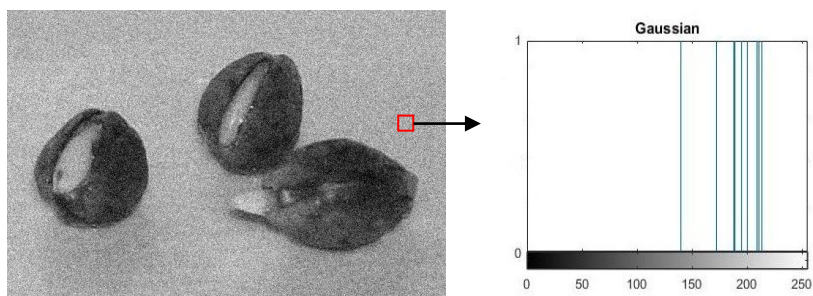


**Gambar 2.4. Local histogram window  $3 \times 3$  pada citra dengan *noise SP***

Pada Matlab dapat diatur "konstanta" *noise*. Konstanta berupa angka numerik non negatif dengan range 0 sampai dengan 1. Makin besar konstantanya citra akan semakin kabur, sebaliknya makin kecil konstantanya efek pada citra makin tidak terlihat. Nilai default untuk konstanta *noise salt & pepper* adalah 0.05.

### 2.2.2. *Noise Gaussian*

Disebut juga *gaussian white noise*. Untuk menambahkan *noise* ini pada Matlab memerlukan input tambahan berupa rata-rata dan variasi. Rata-rata dan variasi merupakan suatu konstanta real. Nilainya bisa positif maupun negatif. Makin besar rata-rata dan variasinya maka citra akan semakin kabur, sebaliknya makin kecil konstantanya efek pada citra makin tidak terlihat. Nilai default adalah 0 untuk *mean* dan 0.01 untuk *variance*.



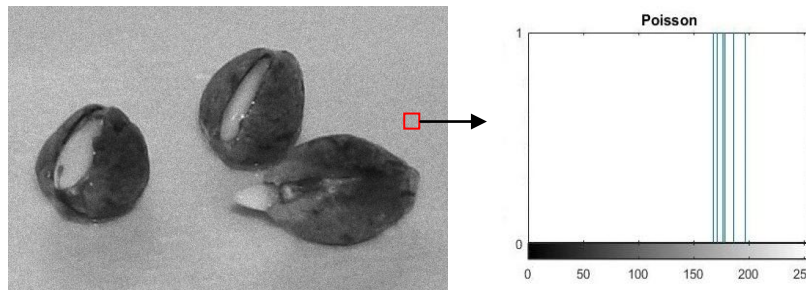
**Gambar 2.5. Local histogram window  $3 \times 3$  pada citra dengan *noise gaussian***

### 2.2.3. *Noise Poisson*

*Poisson noise* bukan merupakan *noise* buatan. *Poisson* merupakan *noise* yang ditambahkan langsung pada citra tanpa kita menambahkan parameter apapun, sehingga efeknya pada citra pun tetap, berbeda dengan tipe *noise*



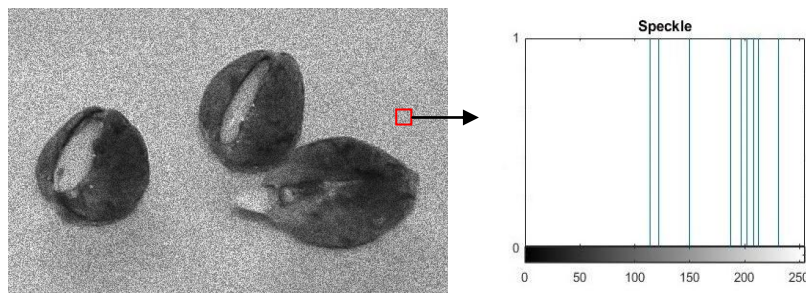
yang sudah dijelaskan sebelumnya. Pada Matlab, jika matrik citra adalah *double precision*, maka nilai piksel inputnya dianggap sebagai *mean* dari distribusi *poisson* dengan skala  $10^{12}$ .



**Gambar 2.6. Local histogram window  $3 \times 3$  pada citra dengan *noise poisson***

#### 2.2.4. Noise Speckle

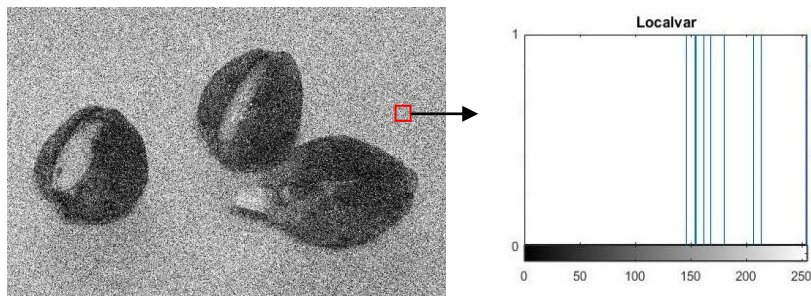
*Speckle* merupakan *noise* ganda. *Noise* ini ditambahkan pada citra menggunakan persamaan  $J=I+n*I$ , dimana  $n$  terdistribusi random seragam dengan *mean* 0 dan *variance*  $V$ .  $V$  adalah konstanta non negative yang besarnya dapat berubah-ubah. Default nilai untuk  $V$  adalah 0.04. Makin besar nilai  $V$  maka citra akan semakin kabur.



**Gambar 2.7. Local histogram window  $3 \times 3$  pada citra dengan *noise speckle***

#### 2.2.5. Noise Localvar

Pada Matlab menggunakan dua parameter untuk menambahkan *noise* ini pada citra. Dua parameter tersebut berupa vector yang ukurannya sama, dan grafik kedua parameter tersebut menggambarkan relasi fungsional antara *varians noise* dan intensitas citra. Vektor intensitas citra harus bernilai antara 0 dan 1 (normal). *Localvar* merupakan *gaussian noise* dengan *mean* 0, dengan *variance noise* adalah fungsi dari intensitas citra yang nilainya berada dalam matrik citra. Vektor intensitas citra tidak boleh bernilai sama karena citra akan nampak sebagai layar putih (*gaussian white noise*).



**Gambar 2.8. Local histogram window  $3 \times 3$  pada citra dengan *noise localvar***

Berdasarkan Gambar 2.3 sampai Gambar 2.8, jika dilihat *local histogram* dari *window  $3 \times 3$*  pada masing-masing citra maka dapat disimpulkan bahwa semua jenis *noise* mempunyai satu karakteristik yang sama yaitu memiliki histogram dengan range nilai piksel yang besar dibandingkan dengan range nilai piksel dari histogram citra asli, sebagai contoh pada Gambar 2.4, citra dengan *noise salt & pepper* memiliki range paling besar yaitu dari 0 sampai 255. Karakteristik ini juga bergantung pada parameter input pada *noise* tersebut.

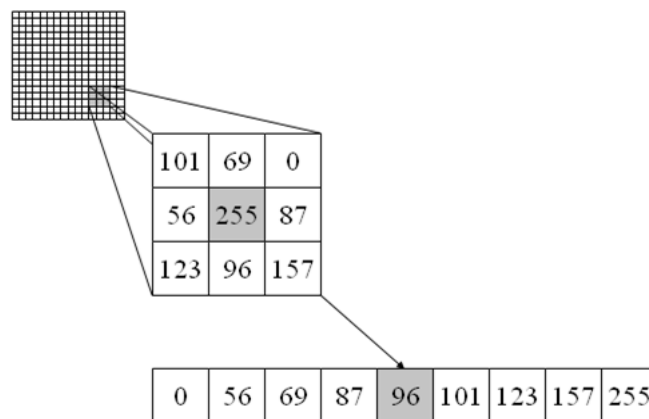
### 2.3. Noise Filtering

*Noise filtering* adalah suatu teknik yang memperhatikan bagaimana mengurangi perubahan bentuk dan penurunan kualitas citra yang diawali selama pembentukan citra tersebut (Pitas, 1993). Gonzalez dan Wood mendefinisikan *noise filtering* sebagai proses yang berusaha merekonstruksi atau mengembalikan suatu citra yang mengalami degradasi. Jadi, *noise filtering* merupakan teknik yang berorientasi pada permodelan degradasi dan menerapkan proses invers dalam rangka merekonstruksi pada citra yang original.

Terdapat banyak penelitian untuk mendeteksi dan menghilangkan *noise* pada gambar dan hampir semua penelitian tersebut fokus pada *noise* dengan tipe nilai *fixed* seperti *Adaptive Median Filter (AMF)*, *Weighted Mean Filter (WMF)*, *Interpolation based Image Noise Removing (IBINR)* dan masih banyak lagi, sedangkan untuk *noise* dengan tipe nilai *random*, banyak yang menggunakan *Median Filter*, *Mean Filter*, *Wiener Filter*, dan *Gaussian Filter*. Selain itu, metode dasar *noise filtering* seperti median dan mean filter juga masih sering digunakan pada beberapa penelitian. Pada bagian ini, akan dibahas metode-metode *noise filtering* yang mendukung penelitian ini.

### 2.3.1. Median Filter

Median filter adalah teknik penyaringan digital nonlinier, yang sering digunakan untuk menghilangkan *noise* dari gambar atau sinyal. Pengurangan *noise* semacam itu merupakan langkah awal untuk memproses hasil pengolahan selanjutnya (misalnya, deteksi tepi pada gambar). Penyaringan median sangat banyak digunakan dalam pemrosesan gambar digital karena, dalam kondisi tertentu, teknik ini mempertahankan tepi sambil menghilangkan *noise* dan juga memiliki aplikasi dalam pemrosesan sinyal.



**Gambar 2.9. Median filter**

Gagasan utama *median filter* adalah menentukan semua nilai piksel dalam sebuah *window*, mengurutkan nilai-nilai piksel tersebut, mencari nilai median, dan kemudian mengganti nilai piksel pada pusat *window* dengan nilai median yang diperoleh. Untuk sinyal 1D, *window* yang paling jelas adalah beberapa nilai sebelum dan sesudahnya, sedangkan untuk sinyal 2D (atau dimensi lebih tinggi) seperti gambar atau citra, pola atau bentuk *window* menjadi lebih kompleks (seperti pola "kotak" atau "salib").

Jika di dalam *window* memiliki jumlah piksel ganjil, maka mediannya sederhana untuk didefinisikan: nilai tengahnya hanya setelah semua entri di jendela diurutkan secara numerik. Untuk jumlah piksel genap, ada lebih dari satu kemungkinan rata-rata untuk menghitung nilai median.

### 2.3.2. Mean Filter

*Mean filter* juga merupakan teknik penyaringan digital nonlinier, yang sering digunakan untuk menghilangkan *noise* dari gambar atau sinyal. Algoritma yang digunakan sangat sederhana yaitu menggantikan nilai pusat

di *window* dengan nilai *mean* (rata-rata) dari semua nilai piksel didalam *window* tersebut. *Window* yang digunakan pada filter ini biasanya berbentuk persegi namun bisa juga berbentuk apapun. Keunggulan filter ini dibandingkan dengan filter median adalah pada efek blur yang diberikan, efek ini dapat membantu proses segmentasi karena memperkecil selisih antara nilai-nilai piksel dalam *window*.

### **2.3.3. New Adaptive Weighted Mean Filter**

*New Adaptive Weighted Mean Filter* (NA-WMF) yang diusulkan pada [1] digunakan untuk mendeteksi dan menghilangkan *salt & pepper noise*. Metode ini mengatasi masalah pada metode sebelumnya seperti kesalahan deteksi *noise* dan hilangnya detail informasi pada citra. Untuk setiap nilai piksel, pertama ditentukan ukuran *window* adaptif dengan terus memperbesar ukuran *window* sampai nilai maksimum dan minimum dari dua *window* berturut-turut sama. Nilai piksel yang sedang diproses dianggap sebagai kandidat *noise* jika sama dengan nilai maksimum atau minimum, jika tidak, piksel tersebut dianggap sebagai piksel bukan *noise*. Selanjutnya, piksel *noise* digantikan dengan nilai rata-rata terbobot, sedangkan piksel bukan *noise* dibiarkan tidak berubah.

### **2.3.4. Weighted Mean Filter for High Density Noise**

*Weighted Mean Filter for High Density Noise* (HD-WMF) yang diusulkan [4] mempunyai keunggulan pada intensitas *noise* diatas 80%. Metode ini membedakan antara piksel yang belum diproses dan yang sudah diproses, kemudian memberikan bobot yang lebih besar pada piksel yang belum diproses tersebut. Proses pembagian bobot lewat persamaan (2.1).

$$w(x,y) = \begin{cases} (1 - \alpha), & \text{if } PP \\ \alpha & , \text{if } UP \end{cases} \quad (2.1)$$

$$\alpha > (1 - \alpha), 0 < \alpha < 1$$

dimana,  $w$  adalah bobot pada piksel yang berada pada koordinat  $(x,y)$ , *PP* (*Processed pixel*) adalah piksel yang sudah diproses, dan *UP* (*Unprocessed pixel*) adalah piksel yang belum diproses. Metode HD-WMF menggunakan *window* berukuran  $3 \times 3$  dengan pembagian menjadi dua daerah untuk *UP* dan *PP* secara simetris.

### **2.3.5. Unbiased Weighted Mean Filter**

*Unbiased Weighted Mean Filter* (U-WMF) yang diusulkan pada [2] juga digunakan untuk menghilangkan *noise* dengan intensitas tinggi. Distribusi asimetris piksel yang terkena *noise* dalam sebuah *window* membuat bias spasial menuju ke piksel pada pusat *window*. U-WMF menghilangkan bias ini dengan mengkalibrasi faktor kontribusi (berat) dari setiap piksel yang tidak terkena *noise* sedemikian rupa sehingga nilai pusat bergeser kembali ke bagian tengah *window*.

Proses restorasi melibatkan tiga operasi sekuensial yaitu deteksi *noise*, mengeliminasi bias spasial, dan menghapus *noise*. Ketiga operasi ini hanya diterapkan pada *window* dengan nilai pusat yang terkena *noise*. Selanjutnya *noise* yang terdeteksi, diganti dengan nilai intensitas baru yang didapat dari proses perhitungan rata-rata dengan menggunakan bobot yang telah dikalibrasi ulang.

### **2.3.6. Interpolation based Image Noise Removing**

*Interpolation based Impulse Noise Removal* (IBINR) yang diusulkan pada [3] menghilangkan *noise* pada gambar menggunakan interpolasi non-linear. Metode ini tidak memerlukan operasi aritmatika atau sortasi kompleks apapun sehingga dapat meningkatkan efisiensi komputasi selain itu metode ini juga mampu mereduksi *noise* dengan intensitas mencapai 60%. Proses pembobotan pada setiap nilai piksel, didapat dari jarak Euclidean antara piksel tersebut dengan piksel pada pusat *window*. Metode ini memiliki kinerja puncak pada ukuran *window*  $5 \times 5$ .

### **2.3.7. Wiener Filter**

Dalam pemrosesan sinyal, wiener filter adalah filter yang digunakan untuk menghasilkan perkiraan proses acak yang diinginkan atau target dengan penyaringan waktu-invariant (LTI) linier dari proses bising yang diamati, dengan asumsi sinyal diam dan spektrum *noise* yang diketahui, dan *noise* tambahan. Filter wiener meminimalkan kesalahan kuadrat rata-rata antara proses acak yang diperkirakan dan proses yang diinginkan.

Tujuan wiener filter adalah menghitung estimasi statistik dari sinyal yang tidak diketahui dengan menggunakan sinyal terkait sebagai masukan dan

penyaringan sinyal yang diketahui untuk menghasilkan perkiraan sebagai keluaran. Karakteristik wiener filter ditandai sebagai berikut: *Asumsi*: kebisingan sinyal dan (aditif) adalah proses stokastik linear stasioner dengan karakteristik spektral yang diketahui atau autokorelasi dan korelasi silang yang diketahui, *Persyaratan*: filter harus dapat direalisasikan secara fisik atau kausal (persyaratan ini dapat dijatuhkan, sehingga menghasilkan solusi non-kausal), dan *Kriteria kinerja*: kesalahan rata-rata minimum (MMSE).

### **2.3.8. Gaussian Filter**

*Gaussian* adalah istilah matematika yang diambil dari nama seorang matematikawan Jerman, *Karl Friedrich Gauss*. Filter *Gaussian* tergolong sebagai filter lolos-rendah (*low-pass filter*) yang didasarkan pada fungsi *Gaussian*. Menurut Usman (2005), *filter Gaussian* sangat baik untuk menghilangkan *noise* yang bersifat sebaran normal, yang banyak di jumpai pada sebaran citra hasil proses digitasi menggunakan kamera karena merupakan fenomena alamiah akibat sifat pantulan cahaya dan kepekaan sensor cahaya pada kamera itu sendiri.

Filter ini menempatkan warna transisi yang signifikan dalam sebuah image, kemudian membuat warna-warna pertengahan untuk menciptakan efek lembut pada sisi-sisi sebuah *image*. *Gaussian blur* adalah salah satu *filter blur* yang menggunakan rumus matematika untuk menciptakan efek *autofocus* untuk mengurangi detail dan menciptakan efek berkabut.

## **2.4. Segmentasi Citra**

Segmentasi citra merupakan bagian dari proses pengolahan citra. Proses segmentasi citra ini lebih banyak merupakan suatu proses pra pengolahan pada sistem pengenalan objek dalam citra. Segmentasi citra (*image segmentation*) mempunyai arti membagi suatu citra menjadi wilayah-wilayah yang homogen berdasarkan kriteria keserupaan yang tertentu antara tingkat keabuan suatu piksel dengan tingkat keabuan piksel – piksel tetangganya, kemudian hasil dari proses segmentasi ini akan digunakan untuk proses tingkat tinggi lebih lanjut yang dapat dilakukan terhadap suatu citra, misalnya proses klasifikasi citra dan proses identifikasi objek.

Gonzalez dan Wintz (1987) menyatakan bahwa segmentasi adalah proses pembagian sebuah citra kedalam sejumlah bagian atau segmen. Segmentasi merupakan suatu bagian yang sangat penting dalam analisis citra secara otomatis, sebab pada prosedur ini obyek yang diinginkan akan disadap untuk proses selanjutnya, misalnya: pada pengenalan pola. Adapun dalam proses segmentasi citra itu sendiri terdapat beberapa algoritma, diantaranya : algoritma deteksi titik, deteksi garis, dan deteksi area pada citra digital.

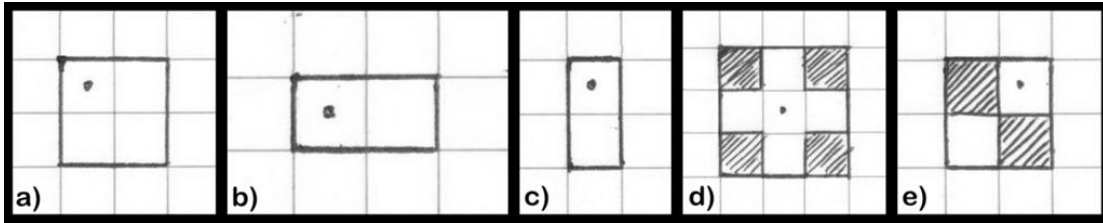
Algoritma segmentasi didasarkan pada dua karakteristik nilai piksel, yaitu: *discontinuity* dan *similarity*. Pada item pertama, citra dibagi atas dasar perubahan yang mencolok dari derajat kecerahannya. Aplikasi yang umum adalah untuk deteksi titik, garis, area, dan sisi citra. Pada kategori ke-dua, didasarkan atas *thresholding*, *region growing*, dan *region spiltting and merging*. Prinsip segmentasi citra bisa diterapkan untuk citra yang statis maupun dinamis.

Subbab ini akan menjelaskan mengenai jenis-jenis metode segmentasi yang digunakan dalam penelitian ini. Metode-metode tersebut terdiri dari teknik morfologi citra, transformasi top-hat, dan *otsu thresholding* untuk deteksi area, kemudian metode Sobel untuk deteksi garis.

#### **2.4.1. Teknik Morfologi**

Teknik morfologi merupakan teknik yang berhubungan dengan bentuk. Pada citra digital mengandung piksel-piksel yang membentuk sekumpulan data dalam struktur dua dimensi. Tujuan operasi morfologi pada citra biner adalah untuk memperbaiki bentuk objek agar menghasilkan fitur-fitur yang lebih akurat ketika dilakukan analisis terhadap objek. Operasi dasar yang terdapat pada teknik morfologi adalah dilasi dan erosi.

Operasi morfologi menggunakan bentuk *structuring element* (SE) yang dioperasikan pada citra masukan untuk memperoleh citra keluaran yang diinginkan. *Structuring element* merupakan salah satu bagian yang sangat penting dalam operasi morfologi citra dengan matriks biner yang bernilai 1 dan 0 dengan bentuk dan ukuran tertentu. Titik tengah pada *structuring element* disebut dengan *origin* yang menjadi pusat matriks *structuring element*. Berikut adalah ilustrasi *structuring element* dengan berbagai bentuk yang ditunjukkan pada Gambar 2.10.

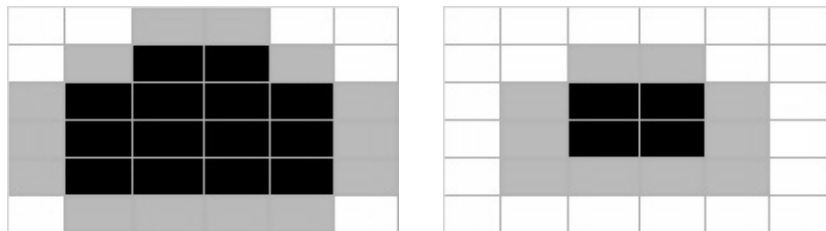


**Gambar 2.10. Struktur elemen**

### 2.4.2. Dilasi dan Erosi

Dilasi adalah operasi morfologi yang akan menambahkan piksel pada batas antar objek dalam suatu citra digital. Atau secara rinci dilasi merupakan suatu proses menambahkan piksel pada batasan dari objek dalam suatu image sehingga nantinya apabila dilakukan operasi ini maka image hasilnya lebih besar ukurannya dibandingkan dengan image aslinya.

Sedangkan, erosi merupakan kebalikkan dari dilasi. Proses ini akan membuat ukuran sebuah citra menjadi lebih kecil. Berbeda dengan dilasi, apabila erosi dilakukan maka yang dikerjakan adalah memindahkan piksel pada batasan-batasan objek yang akan di erosi. Jumlah dari piksel yang ditambah atau dihilangkan bergantung pada ukuran dan bentuk dari structuring element yang digunakan untuk memproses image tersebut. Dibawah ini adalah contoh dilasi dan erosi menggunakan struktur elemen dengan bentuk pada Gambar 2.10 (d).



**Gambar 2.11. Dilasi dan Erosi.**

### 2.4.3. Opening dan Closing

Opening merupakan kombinasi proses dimana suatu citra digital dikenai operasi erosi dilanjutkan dengan dilasi. Operasi opening pada citra mempunyai efek memperhalus batas-batas objek, memisahkan objek-objek yang sebelumnya bergandengan, dan menghilangkan objek-objek yang lebih kecil daripada ukuran structuring. Opening citra  $f$  oleh struktur elemen  $b$  dinyatakan  $f \circ b$ , dinotasikan dengan  $(f \ominus b) \oplus b$ .



Sedangkan, closing merupakan kombinasi dimana suatu citra dikenai operasi dilasi dilanjutkan dengan erosi. Operasi closing juga cenderung akan memperhalus objek pada citra, namun dengan cara menyambung pecahan-pecahan (*fuses narrow breaks and thin gulf*) dan menghilangkan lubang-lubang kecil pada objek. Sebagaimana sebelumnya, opening adalah erosi  $f$  oleh  $b$ , diikuti dilasi hasilnya dengan  $b$ . Closing  $f$  oleh  $b$  dinyatakan  $f \bullet b$  dinotasikan dengan  $(f \oplus b) \ominus b$ .

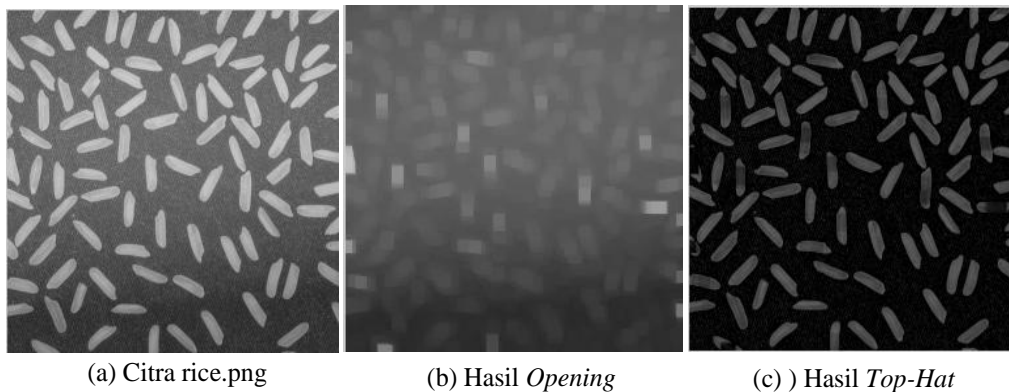
#### 2.4.4. Transformasi Top-Hat

Dengan mengkombinasikan pengurangan citra dengan opening dan closing maka akan menghasilkan transformasi *top-hat* dan *bottom-hat*. Transformasi *top-hat* citra grayscale  $f$  didefinisikan sebagai  $f$  dikurangi hasil opening  $T_{hat}(f) = f - (f \circ b)$ . Sedangkan untuk transformasi *bottom-hat*, didefinisikan sebagai closing  $f$  dikurangi dengan  $B_{hat}(f) = (f \bullet b) - f$ .

Perbedaan dari kedua operasi adalah citra dimana hanya terdapat sisa dari komponen yang dibuang. Transformasi *top-hat* digunakan untuk obyek terang pada background gelap, sedangkan transformasi *bottom-hat* digunakan untuk obyek gelap pada background terang.

Dalam morfologi matematis dan pemrosesan gambar digital, *top-hat* transform adalah operasi yang mengekstrak elemen kecil dan rincian dari gambar yang diberikan. Transformasi *top-hat* digunakan untuk berbagai tugas pemrosesan gambar, seperti ekstraksi fitur, pemerataan latar belakang, peningkatan citra, dan lain-lain seperti memperbaiki efek illumination yang tidak uniform.

Untuk lebih mudah dipahami, perhatikan Gambar 2.12 (a). Butiran beras memiliki intensitas yang tidak seragam karena pencahayaan yang tidak merata, beberapa beras memiliki nilai intensitas yang sama dengan backgroundnya hal ini tentu mempersulit proses segmentasi. Melalui *opening*, pada Gambar 2.12 (b) dan diikuti dengan transformasi *Top-Hat* ditunjukkan pada Gambar 2.12 (c). Maka intensitas beras dan background masing-masing menjadi lebih seragam dibandingkan pada citra asal. Hal ini tentunya mempermudah proses segmentasi.



**Gambar 2.12. Contoh transformasi Top-Hat**

Pada proses rekonstruksi citra, transformasi morfologi yang digunakan melibatkan dua citra dan sebuah *structuring element*. Satu citra sebagai marker merupakan titik awal untuk melakukan transformasi. Sedangkan satu citra yang lain digunakan sebagai mask, yaitu batasan dari transformasi. *Structuring element* digunakan untuk mendefinisikan hubungan antar kedua citra tersebut.

#### **2.4.5. Otsu Thresholding**

Tujuan dari metode otsu adalah membagi histogram citra *gray level* kedalam dua daerah yang berbeda secara otomatis tanpa membutuhkan bantuan *user* untuk memasukkan nilai ambang. Pendekatan yang dilakukan oleh metode otsu adalah dengan melakukan analisis diskriminan yaitu menentukan suatu variabel yang dapat membedakan antara dua atau lebih kelompok yang muncul secara alami. Analisis diskriminan tersebut diharapkan dapat memaksimalkan pemisahan objek (*foreground*) dan latar belakang (*background*).

Citra dengan intensitas 8 bit memiliki *gray level* antara rentang nilai 1 sampai  $L$ , dimana  $L=255$ . Sedangkan  $k$  merupakan nilai ambang yang berada diantara nilai tersebut. Dengan kata lain, *background* bernilai  $[1..k]$  dan *foreground* bernilai  $[k+1..L]$ . Beberapa fungsi yang digunakan dalam metode otsu seperti; probabilitas, *zeroth cumulative moment*, *first cumulative moment*, dan total nilai mean, ditunjukkan berturut turut pada penjelasan dibawah ini.

Berdasarkan penjelasan sebelumnya bahwa terdapat 2 kluster nilai piksel yaitu *background* dan *foreground*, dari kedua kluster tersebut akan dihitung nilai probabilitas ( $\omega_0, \omega_1$ ) dan rata-rata ( $\mu_0, \mu_1$ ) menggunakan persamaan (2.2).

$$\begin{aligned}
 p_i &= \frac{n_i}{N}, \quad p_i \geq 0, \quad \sum_{i=1}^L p_i = 1 \\
 \omega_0 &= \sum_{i=1}^k p_i = \omega(k) \\
 \omega_1 &= \sum_{i=k+1}^L p_i = 1 - \omega(k) \\
 \mu_0 &= \sum_{i=1}^k \frac{ip_i}{\omega_0} = \frac{\mu(k)}{\omega(k)} \\
 \mu_1 &= \sum_{i=k+1}^L \frac{ip_i}{\omega_1} = \frac{\mu_T - \mu(k)}{1 - \omega(k)}
 \end{aligned} \tag{2.2}$$

dimana *zeroth cumulative moment* ( $\omega(k)$ ), *first cumulative moment* ( $\mu(k)$ ), dan total nilai rata-rata ( $\mu_T$ ) dihitung menggunakan (2.3).

$$\begin{aligned}
 \omega(k) &= \sum_{i=1}^k p_i \\
 \mu(k) &= \sum_{i=1}^k ip_i \\
 \mu_T &= \mu(L) = \sum_{i=1}^L ip_i \\
 \omega_0\mu_0 + \omega_1\mu_1 &= \mu_T, \quad \omega_0 + \omega_1 = 1
 \end{aligned} \tag{2.3}$$

Metode otsu sangat sederhana karena pencarian threshold hanya memanfaatkan, *zeroth cumulative moment*, *first cumulative moment*, dan total nilai mean pada histogram. Selain itu, metode ini juga dapat menganalisa mean level kelas dan evaluasi kelas pemisahan.

#### 2.4.6. Metode Sobel

Deteksi tepi berfungsi untuk mengidentifikasi garis batas (*boundary*) dari suatu objek yang terdapat pada citra. Tepian dapat dipandang sebagai lokasi piksel dimana terdapat nilai perbedaan intensitas citra secara ekstrem. Sebuah *edge detector* bekerja dengan cara mengidentifikasi dan menonjolkan lokasi-lokasi piksel yang memiliki karakteristik tersebut. Pada penelitian ini *edge detector* yang digunakan adalah metode sobel.

Operator Sobel, yang sering disebut operator Sobel-Feldman atau Sobel filter, sering digunakan dalam sistem pengolahan gambar dan visi komputer, khususnya dalam algoritma deteksi tepi. Operator menggunakan dua *window* 3×3 ( $G_x$  dan  $G_y$ ) yang dikonvolusi dengan gambar asli untuk menghitung perkiraan turunan (*approximations of the derivatives*), *window*  $G_x$  digunakan untuk perubahan horisontal, dan *window*  $G_y$  untuk vertikal.

$$G_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \text{ dan } G_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Peninjauan pengaturan piksel dalam *window* 3×3 adalah sebagai berikut

$$\begin{bmatrix} f(x-1, y-1) & f(x, y-1) & f(x+1, y-1) \\ f(x-1, y) & f(x, y) & f(x+1, y) \\ f(x-1, y+1) & f(x, y+1) & f(x+1, y+1) \end{bmatrix}$$

*Window*  $G_x$  dan *window*  $G_y$  masing-masing di konvolusi dengan peninjauan pengaturan piksel dalam *window* 3×3 diatas sehingga menghasilkan aproksimasi gradien (*gradient approximations*)  $S_x$  dan  $S_y$  sebagai berikut.

$$\begin{aligned} S_x &= (f(x-1, y+1) + 2 \times f(x, y+1) + f(x+1, y+1)) - (f(x \\ &\quad -1, y-1) + 2 \times f(x, y-1) + f(x+1, y-1)) \\ S_y &= (f(x+1, y-1) + 2 \times f(x+1, y) + f(x+1, y+1)) - (f(x \\ &\quad -1, y-1) + 2 \times f(x-1, y) + f(x-1, y+1)) \end{aligned} \quad (2.4)$$

Nilai aproksimasi gradien kemudian dikombinasikan untuk menghasilkan magnitudo gradien, dengan menggunakan (2.5).

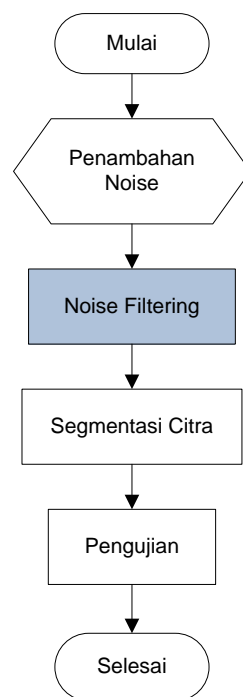
$$M = \sqrt{(S_x)^2 + (S_y)^2} \quad (2.5)$$

Nilai magnitudo yang didapat menggantikan nilai piksel  $f(x, y)$ .

## BAB 3

### METODOLOGI PENELITIAN

Metodologi penelitian adalah sekumpulan peraturan, kegiatan, dan prosedur yang digunakan oleh peneliti dalam suatu disiplin ilmu, metodologi juga merupakan analisis teoritis mengenai suatu cara atau metode. Secara garis besar prosedur dalam penelitian ini beserta kontribusi yang diberikan terdiri dari tahap-tahap sebagai berikut.

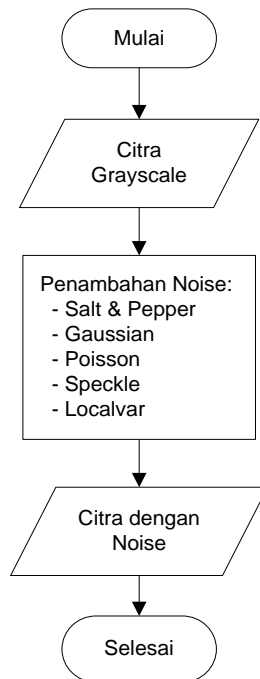


**Gambar 3.1. Prosedur penelitian**

Berdasarkan Gambar 3.1, penelitian ini terdiri dari empat tahap pemrosesan yaitu penambahan *noise*, *noise filtering*, segmentasi citra, dan tahap terakhir adalah pengujian pada citra hasil *noise filtering* dan segmentasi. Sesuai dengan judul penelitian, maka kontribusi yang diberikan ada pada tahap ke-2 yaitu pada *noise filtering*, metode *Soft Weighted Median Filter* yang diusulkan berada pada tahap ini bersama dengan metode-metode *noise filtering* yang lain. Dari tahap ke-2 kemudian dilanjutkan dengan proses segmentasi dan pengujian yang dilakukan untuk mengetahui seberapa efektif metode yang diusulkan serta mengukur tingkat keberhasilan dalam proses segmentasi.

### 3.1. Tahap Penambahan *Noise*

Tahap penambahan *noise* merupakan tahap awal dalam penelitian ini. Tahap ini dilakukan untuk mempersiapkan citra yang akan diproses sesuai dengan kebutuhan pada proses berikutnya. Ada lima jenis *noise* yang akan ditambahkan ke dalam citra yaitu *noise salt & pepper*, *gaussian*, *poisson*, *speckle*, dan *localvar*. Noise-noise tersebut telah dijelaskan pada bab sebelumnya dan nilai parameter yang digunakan untuk setiap *noise* adalah nilai standar atau *default*.



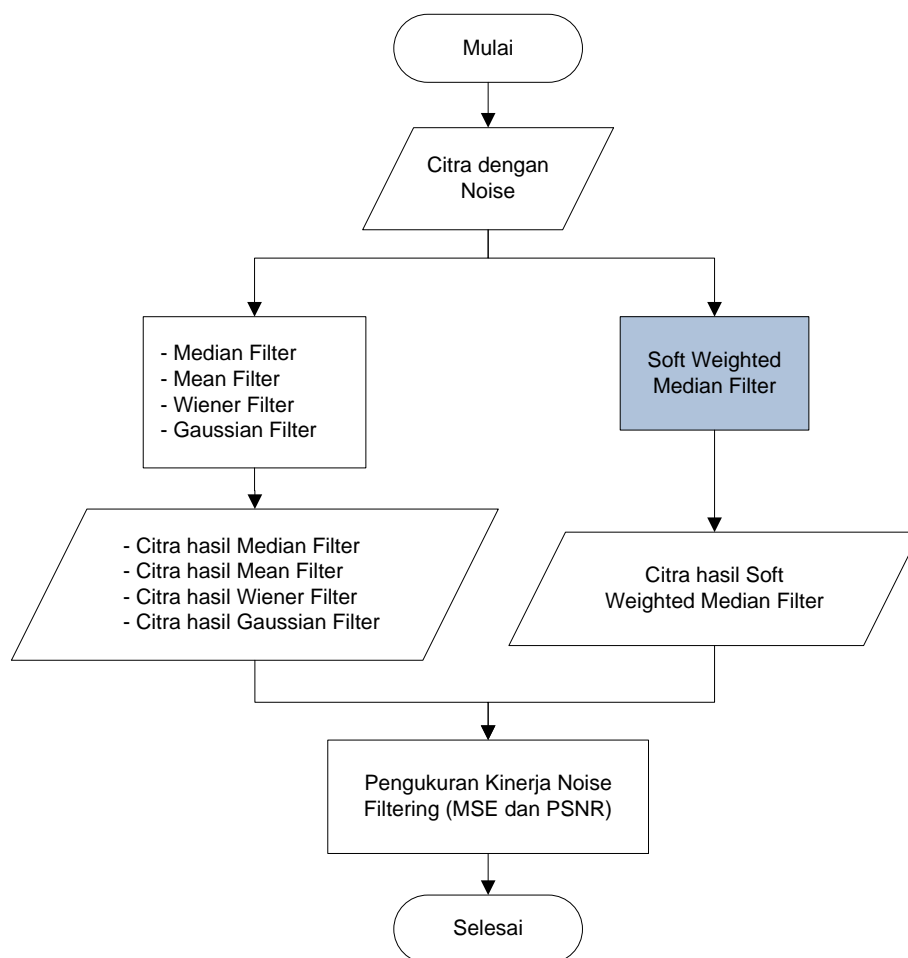
**Gambar 3.2. Tahap-tahap penambahan *noise***

Citra *input* yang digunakan pada penelitian ini adalah citra *grayscale* dengan satu atau beberapa objek beserta *background*-nya, intensitas pada citra-citra tersebut adalah 8 bit atau dengan rentang nilai piksel dari 0 sampai 255, dan format yang telah ditentukan adalah *.jpg*.

### 3.2. Tahap *Noise Filtering*

Proses penghilangan *noise* pada penelitian ini menggunakan empat metode sebagai pembanding dengan metode yang diusulkan yaitu *Median Filter*, *Mean Filter*, *Wiener Filter*, dan *Gaussian Filter*. Keempat metode tersebut merupakan yang paling sering digunakan pada aplikasi-aplikasi pengolahan citra. Selain itu, metode-metode ini dapat digunakan untuk *fixed-valued noise* maupun *random-valued noise*.

Pada proses pengujian, semua metode menggunakan ukuran *window* yang sama yaitu  $3 \times 3$  dan menggunakan nilai *default* untuk setiap parameternya. Citra hasil proses *noise filtering* dari masing-masing metode akan disimpan dengan format, intensitas, dan dimensi yang sama, kemudian digunakan kembali pada proses segmentasi bersama dengan citra aslinya (citra sebelum diberi *noise*). Pada tahap ini, pengukuran kinerja untuk setiap metode adalah menggunakan parameter MSE dan PSNR. Kedua parameter ini menghitung penyimpangan nilai piksel antara citra asli dengan citra setelah *noise filtering*.



**Gambar 3.3. Tahap-tahap *noise filtering***

### 3.3. *Soft Weighted Median Filter*

*Soft Weighted Median Filter* yang diusulkan pada penelitian ini mempunyai beberapa keunggulan salah satunya adalah dapat digunakan untuk *fixed-valued noise* maupun *random-valued noise*. Metode ini merupakan gabungan dari metode *Mean Filter* dan *Median Filter* dimana proses pembobotan untuk setiap nilai

pikselnya menggunakan fungsi perpangkatan, dengan demikian metode ini menekan pada nilai tengah namun tetap memperhitungkan nilai piksel tetangga untuk mendapatkan nilai piksel baru. Metode *Soft Weighted Median Filter* terdiri dari dua tahapan proses yaitu proses deteksi *noise* dan proses reduksi *noise*. Masing masing proses menggunakan prosedur yang berbeda dengan metode-metode sebelumnya.

### 3.3.1. Deteksi *noise*

Pada tahap ini, digunakan *local histogram* dari nilai-nilai piksel pada setiap *window* untuk mengetahui karakteristik setiap citra ber-*noise* guna membuat sebuah prosedur penentuan *noise*. Karakteristik dari setiap *noise* pada *window* dapat dilihat sebelumnya pada Gambar 2.4 sampai 2.8, gambar dengan *noise*, umumnya memiliki *local histogram* dengan rentang nilai yang besar. Sebagai contoh, pada Gambar 2.4, gambar dengan *noise salt & pepper* memiliki rentang terbesar yaitu dari 0 sampai 255. Karakteristik ini juga bergantung pada parameter masukan setiap *noise*.

Proses mendeteksi *noise* dalam metode yang diusulkan berbeda dengan metode lainnya. Untuk mendeteksi semua jenis *noise*, prosedur yang diusulkan sangat sederhana. Pertama, nilai piksel pada window  $r$  dengan ukuran  $m \times n$  dan piksel center  $I(i, j)$  diurutkan menggunakan (3.1).

$$F_k(x, y) = \text{sort}(I(i, j)), (i, j) \in r, k = (1, 2, \dots, m \times n), \quad (3.1)$$

dimana  $i = \left(x - \left\lfloor \frac{m}{2} \right\rfloor, \dots, x + \left\lfloor \frac{m}{2} \right\rfloor\right)$  dan  $j = \left(y - \left\lfloor \frac{n}{2} \right\rfloor, \dots, y + \left\lfloor \frac{n}{2} \right\rfloor\right)$ .

Kedua, nilai piksel yang telah diurutkan  $F_k(x, y)$  dibagi menjadi tiga bagian, jika nilai piksel berada pada bagian satu atau tiga maka piksel tersebut dianggap sebagai *noise*, sedangkan jika nilai piksel yang berada pada bagian dua maka dianggap bukan *noise*. Persamaan (3.2) digunakan untuk menentukan piksel bukan *noise* (*Un-Noiseed pixels*) sesuai kriteria diatas.

$$UN = \{F_a(x, y), F_{a+1}(x, y), \dots, F_b(x, y)\}, \quad (3.2)$$

dimana  $a = \left\lfloor \frac{m \times n}{3} + 1 \right\rfloor$ , dan  $b = \left\lfloor \frac{2(m \times n)}{3} \right\rfloor$ , nilai piksel diluar kriteria pada persamaan ini, dianggap sebagai *noise*.

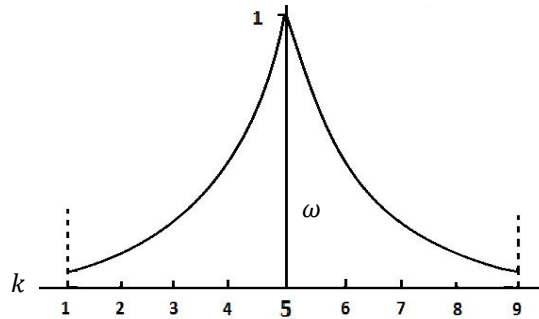


### 3.3.2. Reduksi noise

Tahap ini hanya diterapkan pada piksel yang dianggap *noise*, menggunakan pembobotan nilai median yang sederhana untuk mengganti nilai piksel yang rusak. Metode yang diusulkan menentukan nilai median berdasarkan nilai-nilai piksel dalam *window* yang sudah diurutkan. Bobot maksimum pada nilai median adalah 1.

$$\omega_k = \begin{cases} \frac{c^k}{c^{med}} & , if k < med \\ 1 & , if k = med \\ \frac{c^{(2med-k)}}{c^{med}} & , if k > med \end{cases} \quad (3.3)$$

Pada persamaan (3.3), terdapat tiga kondisi untuk menentukan bobot setiap nilai piksel pada *window*, dimana  $k$  adalah posisi piksel yang sudah diurutkan  $[1,2,3,\dots,N]$  dalam *window*,  $med$  adalah posisi piksel median dalam *window*,  $\omega_k$  adalah nilai bobot yang diberikan, dan  $c$  adalah nilai pangkat dari  $k$  dengan default adalah 2. Gambar 3.4 mengilustrasikan prosedur pembobotan berdasarkan persamaan (3.3).



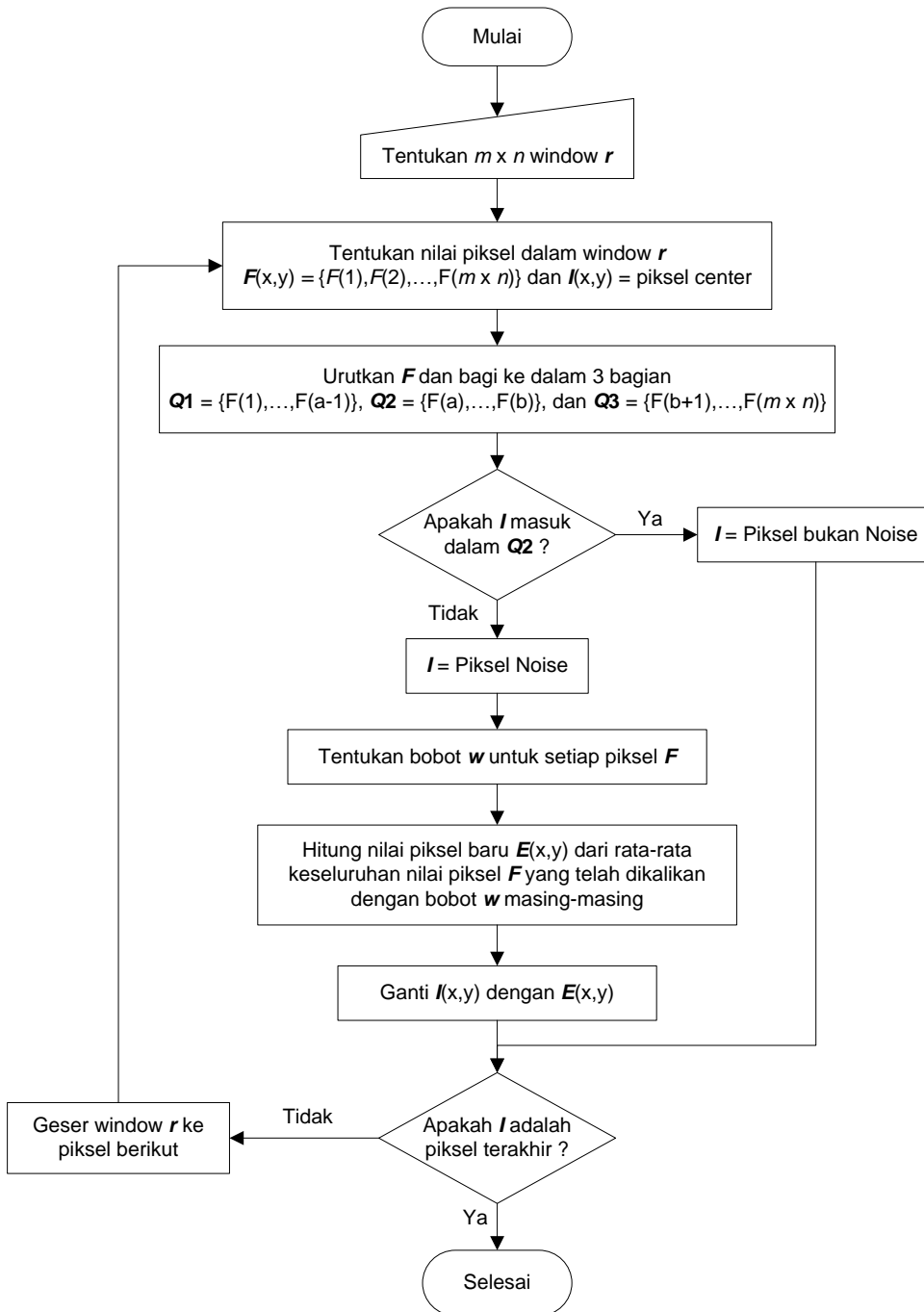
**Gambar 3.4. Ilustrasi proses pembobotan nilai piksel**

Setelah *noise* dideteksi dan diberikan bobot, maka selanjutnya adalah menghitung nilai piksel baru dengan rata keseluruhan nilai piksel pada *window* yang telah dikalikan dengan bobotnya masing-masing.

$$E(x, y) = \left[ \frac{\sum_{k=1}^{m \times n} \omega_k \times F_k(x, y)}{\sum_{k=1}^{m \times n} \omega_k} \right] \quad (3.4)$$

dimana  $E(x, y)$  adalah nilai piksel baru pada koordinat  $(x, y)$ ,  $\omega_k$  adalah bobot nilai piksel, dan  $F_k(x, y)$  adalah nilai piksel yang sudah diurutkan pada pusat *window*  $(x, y)$ .

Proses *Soft Weighted Median Filter* secara keseluruhan adalah sebagai berikut.



**Gambar 3.5. Proses SWMF**

Nilai piksel hasil reduksi *noise* digunakan kembali pada proses perhitungan piksel berikutnya. Hal ini dilakukan untuk memperkecil kemungkinan kesalahan deteksi *noise* pada piksel-piksel berikutnya.

### 3.4. Tahap Segmentasi

Segmentasi citra pada penelitian ini terdiri dari dua tahap yaitu deteksi area dan deteksi garis. Citra hasil proses segmentasi digunakan untuk mengetahui perbandingan tingkat keberhasilan antara metode yang diusulkan dengan metode yang lain dalam menghilangkan *noise*.

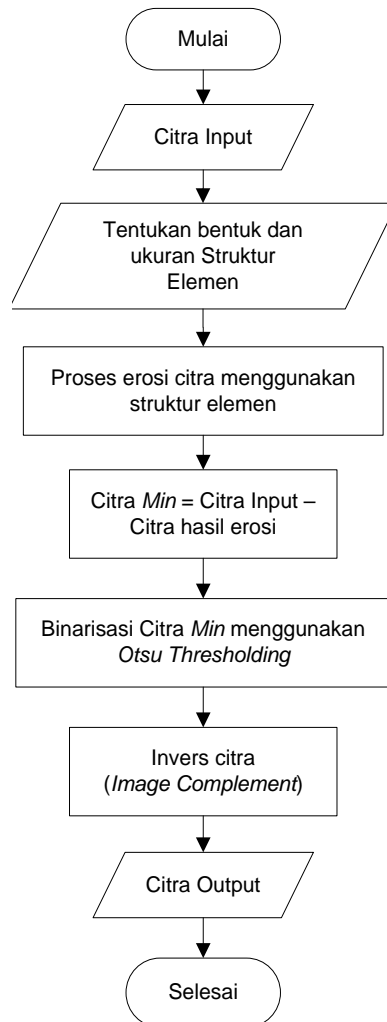
Citra *input* pada tahap ini adalah citra hasil *noise filtering* menggunakan metode-metode yang telah dijelaskan sebelumnya dan merupakan citra *grayscale*. Selain itu, digunakan juga citra asli yang akan disegmentasi dan digunakan sebagai *groundtruth*.

#### 3.4.1. Deteksi area

Tahap deteksi area menggunakan transformasi Top-Hat, penjelasan beserta contoh dari metode ini telah dijelaskan pada bab sebelumnya. Langkah-langkah deteksi area terdiri dari: penentuan struktur elemen, erosi citra, pengurangan citra dengan hasil erosi, binarisasi citra, complement citra, dan filling citra. Proses deteksi area pada penelitian ini ditampilkan pada Gambar 3.6 dan penjelasan secara detail dari tahap-tahap pada Gambar 3.6 adalah sebagai berikut:

- a. Langkah pertama adalah menentukan bentuk serta ukuran dari struktur elemen yang akan digunakan pada proses berikutnya.
- b. Langkah kedua adalah erosi citra dengan struktur elemen yang telah ditentukan. Proses ini dimaksud untuk memberikan efek blur sehingga membuat gambar yang akan diproses menjadi seperti background dengan intensitas nilai piksel yang relative sama.
- c. Langkah ketiga adalah pengurangan citra awal dengan citra hasil erosi, proses ini dilakukan untuk mendapatkan objek yang lebih kontras dengan backgroundnya.
- d. Langkah keempat adalah membagi citra ke dalam dua bagian atau binarisasi citra menggunakan *Otsu thresholding*.
- e. Langkah kelima adalah *image complement* atau pembalikan citra. Dalam gambar biner, angka nol menjadi satu dan yang satu menjadi nol atau warna hitam dan putih ditukar.

- f. Langkah terakhir adalah evaluasi hasil deteksi area menggunakan perhitungan *sensitivity*, *specificity* dan *accuracy* antara citra hasil deteksi area dengan citra *groundtruth*.

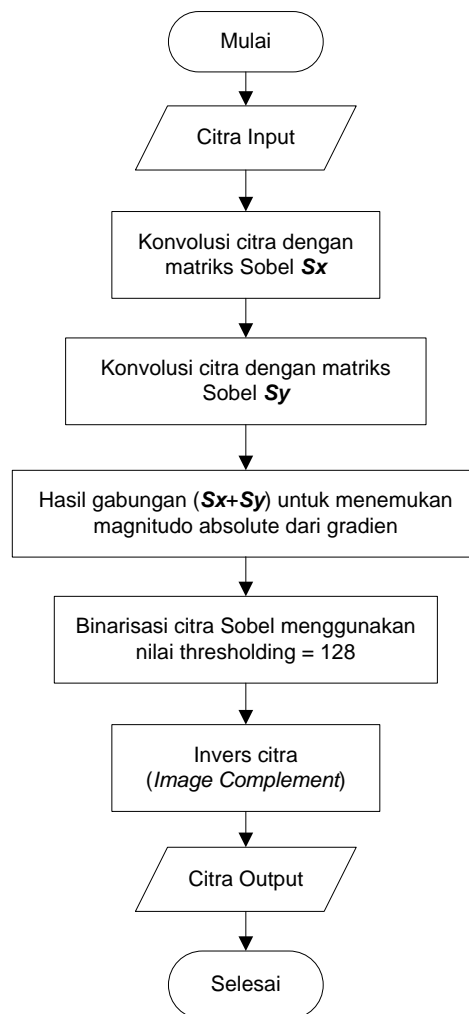


**Gambar 3.6. Tahap-tahap deteksi area**

### 3.4.2. Deteksi Garis

Tahap deteksi garis menggunakan metode Sobel, penjelasan beserta contoh dari metode ini telah dijelaskan pada bab sebelumnya. Langkah-langkah metode Sobel terdiri dari: konvolusi citra inputan dengan matriks sobel, penjumlahan citra hasil konvolusi dengan matriks sobel, binarisasi citra, dan pengukuran secara visual.

Berikut adalah gambar serta penjelasan secara detail dari tahap-tahap deteksi garis pada penelitian ini:



**Gambar 3.7. Tahap-tahap deteksi garis**

- a. Langkah pertama dan kedua, setelah input citra adalah proses konvolusi citra dengan matriks sobel  $S_x$  dan matriks sobel  $S_y$  berukuran  $3 \times 3$ . Nilai piksel pada tepian objek atau pada garis umumnya mempunyai selisih yang besar dengan nilai piksel tetangganya, langkah ini dimaksudkan untuk menemukan kondisi tersebut.  
Untuk setiap matriks  $S_x$ , nilai piksel pada sumbu horizontal bawah dikurangi dengan nilai piksel pada sumbu horizontal atas. Sedangkan untuk matriks  $S_y$ , nilai piksel pada sumbu vertikal kanan dikurangi dengan nilai piksel pada sumbu vertikal kiri.
- b. Langkah ketiga adalah menghitung nilai absolute dari magnitude gradient ( $S_x+S_y$ ). Nilai ini adalah nilai akhir untuk tahap deteksi tepi menggunakan metode sobel.

- c. Langkah keempat adalah membagi citra ke dalam dua bagian atau binarisasi citra menggunakan nilai thresholding tengah yaitu 128.
- d. Langkah kelima adalah image complement atau pembalikan citra. Dalam gambar biner, angka nol menjadi satu dan yang satu menjadi nol atau warna hitam dan putih ditukar.
- e. Langkah terakhir adalah evaluasi hasil deteksi garis menggunakan pengamatan secara visual.

### 3.5. Tahap Pengujian

Tahap pengujian pada penelitian ini terdiri dari dua bagian yaitu; pengujian pada kinerja *noise filtering* dan pengujian pada segmentasi citra (deteksi area). Pada tahap ini akan dijelaskan metode-metode yang akan digunakan pada kedua pengujian tersebut.

#### 3.5.1. Pengujian *Noise Filtering*

Penilaian kualitas citra hasil *noise removing* dilakukan secara objektif dengan menggunakan besaran MSE (*Mean Square Error*) dan PSNR (*Peak Signal to Noise Ratio*), kedua besaran tersebut membandingkan piksel-piksel pada posisi yang sama dari dua citra yang berlainan.

MSE merupakan rata-rata kuadrat nilai kesalahan, sedangkan PSNR merupakan nilai perbandingan antara harga maksimum warna pada citra hasil *filtering* dengan kuantitas gangguan (*noise*), yang dinyatakan dalam satuan desibel (dB), *noise* yang dimaksud adalah akar rata-rata kuadrat nilai kesalahan ( $\sqrt{MSE}$ ). Kedua parameter diatas secara matematis dapat dirumuskan pada (3.5) dan (3.6).

$$MSE = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} |(f(x, y) - g(x, y))|^2 \quad (3.5)$$

$$PSNR = 20 \log_{10} \left( \frac{255}{\sqrt{MSE}} \right) \quad (3.6)$$

dimana,  $M$  dan  $N$  adalah ukuran *window*,  $f(x, y)$  adalah intensitas nilai piksel citra asli, dan  $g(x, y)$  adalah intensitas nilai piksel hasil *noise filtering*.

### 3.5.2. Pengujian Segmentasi

Citra yang digunakan pada tahapan ini adalah citra biner dengan komposisi *foreground* (objek) dan *background*. Untuk membandingkan hasil segmentasi citra yang mengandung objek dilakukan evaluasi kinerja secara kuantitatif. Evaluasi hasil segmentasi menggunakan perhitungan *sensitivity*, *specificity*, dan *accuracy* terhadap citra *groundtruth*.

*Sensitivity* didefinisikan sebagai nilai sensitivitas yang mengacu pada kemampuan hasil segmentasi untuk mendeteksi area secara benar yang dinyatakan dengan observasi oleh ahli. *Sensitivity* dinyatakan dengan rasio perbandingan dari piksel *foreground* yang dikenali sebagai *foreground* sebagai *true positive (TP)* dengan piksel *foreground* yang dikenali sebagai *background* sebagai *false negative (FN)* ditunjukkan pada Gambar 3.8. *Sensitivity* dapat dirumuskan sebagai berikut:

$$sensitivity = \frac{True\ Positive\ (TP)}{True\ Positive\ (TP) + False\ Negative\ (FN)} \quad (3.7)$$

*Specificity* mengukur jumlah piksel *background* yang dikenali sebagai *background* yaitu *true negative (TN)* dengan jumlah *background* yang dikenali sebagai *foreground* yaitu *false positive (FP)* dapat dirumuskan sebagai berikut:

$$Specificity = \frac{True\ Negative\ (TN)}{True\ Negative\ (TN) + False\ Positive\ (FP)} \quad (3.8)$$

*Accuracy* sebagai ukuran ketepatan algoritma di tingkat piksel. Piksel ini berdasarkan evaluasi mempertimbangkan empat nilai, yaitu *True Positive (TP)*, jumlah piksel-piksel terdeteksi dengan benar, *False Positive (FP)*, jumlah piksel-piksel bukan objek yang terdeteksi secara salah sebagai objek, *False Negative (FN)*, jumlah piksel-piksel objek yang tidak terdeteksi dan *True Negative (TN)*, jumlah piksel-piksel bukan objek yang secara benar diidentifikasi sebagai piksel-piksel bukan objek. Dirumuskan sebagai berikut:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \quad (3.9)$$

		Predicted class	
		$P$	$N$
Actual Class	$P$	True Positives (TP)	False Negatives (FN)
	$N$	False Positives (FP)	True Negatives (TN)

**Gambar 3.8. Confusion Matriks**

*Confusion matriks* adalah tata letak tabel yang spesifik yang memungkinkan visualisasi kinerja algoritma. Masing-masing baris matriks mewakili instance dalam kelas yang diprediksi sementara masing-masing kolom mewakili instance di kelas sebenarnya atau dengan kata lain proses ini membentuk tabel kontingensi khusus, dengan dua dimensi ("aktual" dan "perkiraan"), dan rangkaian kelas identik di kedua dimensi.



## **BAB 4**

### **HASIL DAN PEMBAHASAN**

Pada bab ini, dijelaskan tahap-tahap yang dilakukan untuk mendapatkan hasil beserta analisis yang digunakan dalam menyelesaikan penelitian ini. Selanjutnya secara terperinci tahap-tahap tersebut dibahas dan dianalisis secara detail dan tajam, dengan menggunakan metode yang telah dijelaskan dalam metodologi penelitian, sampai diperoleh suatu hasil penelitian. Analisis dan pembahasan ini, dilakukan untuk tujuan yang telah ditetapkan pada tujuan penelitian.

Tahapan uji coba, hasil, beserta penjelasan pada penelitian ini terdiri dari tiga yaitu lingkungan dan data uji coba, pelaksanaan, dan yang terakhir adalah evaluasi hasil. Berikut adalah penjelasan detail dari tahap-tahap tersebut.

#### **4.1. Lingkungan dan Data Uji Coba**

Adapun lingkungan perangkat lunak yang digunakan pada uji coba ini adalah sebagai berikut:

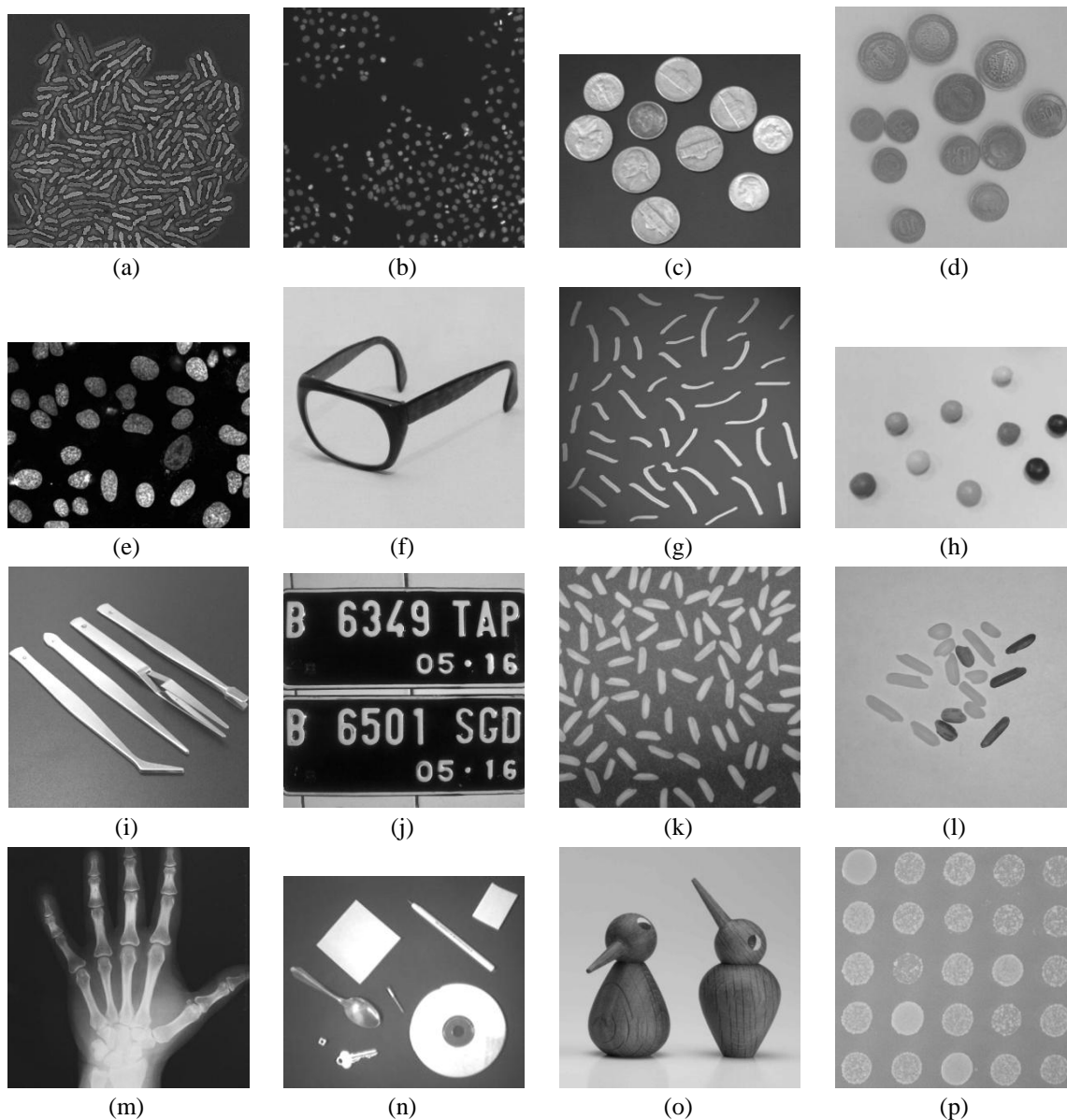
- a. Sistem operasi *Microsoft Window 10 Pro*
- b. Aplikasi *Matrix Laboratory MATLAB R2016b, MathWorks Inc.*

Lingkungan perangkat keras yang digunakan adalah komputer personal (*stand-alone*) yang memiliki spesifikasi sebagai berikut:

- a. *Processor* : Intel ® Core™ i5-6200U CPU @ 2.40 GHz
- b. *Installed memory* : 4.00 GB
- c. *System type* : 64-bit *Operating System*

Data uji coba yang digunakan pada penelitian ini adalah dataset citra *grayscale* dengan satu atau beberapa objek beserta *background* di dalamnya, jumlah data yang digunakan adalah 16 citra, dengan spesifikasi yang telah dijelaskan pada bab sebelumnya.

Sebagian besar data uji coba merupakan dataset citra yang sering digunakan pada beberapa penelitian sebelumnya yang terkait dengan proses *noise filtering* dan segmentasi citra, baik itu yang merupakan dataset citra dari *The Mathworks Inc*, maupun yang berasal dari data public (*free*). Gambar 4.1 menampilkan citra uji coba secara lengkap.



**Gambar 4.1. Citra Uji Coba**

Keterangan dari Gambar 4.1 adalah sebagai berikut.

- Gambar 4.1(a): Cell.jpg, (460×460)
- Gambar 4.1(b): Cell2.jpg, (500×500)
- Gambar 4.1(c): Coins.jpg, (500×400)
- Gambar 4.1(d): Coins2.jpg, (460×460)
- Gambar 4.1(e): Contour.jpg, (520×400)
- Gambar 4.1(f): Eyeglass.jpg, (400×400)
- Gambar 4.1(g): Lighting.jpg, (500×500)
- Gambar 4.1(h): Marbels.jpg, (480×360)

- Gambar 4.1(i): Pincer.jpg, (400×400)
- Gambar 4.1(j): Plats.jpg, (410×400)
- Gambar 4.1(k): Rice.jpg, (400×400)
- Gambar 4.1(l): Rice2.jpg, (460×460)
- Gambar 4.1(m): Rontgen.jpg, (400×400)
- Gambar 4.1(n): Shapessm.jpg, (410×360)
- Gambar 4.1(o): Woodtoys.jpg, (410×410)
- Gambar 4.1(p): Yeast.jpg, (500×500)

Semua citra yang digunakan pada uji coba penelitian ini merupakan citra *grayscale*, dengan intensitas 8 bit. Rata-rata ukuran dimensi untuk semua citra uji coba adalah  $\leq 500 \times 500$ , hal ini dimaksud untuk mempercepat proses komputasi.

## 4.2. Pelaksanaan dan Hasil Uji Coba

### 4.2.1. Pelaksanaan dan Hasil Penambahan *Noise*

Pembahasan tentang *noise* sebelumnya telah disampaikan secara lengkap pada Bab 2. Pada pelaksanaannya, proses ini menggunakan bahasa pemrograman Matlab yaitu dengan menggunakan fungsi (*imnoise*). Fungsi ini merupakan yang paling umum digunakan untuk menambahkan *impulse noise* pada citra. Citra hasil penambahan *noise* dapat disimpan dan digunakan kembali untuk proses berikutnya.

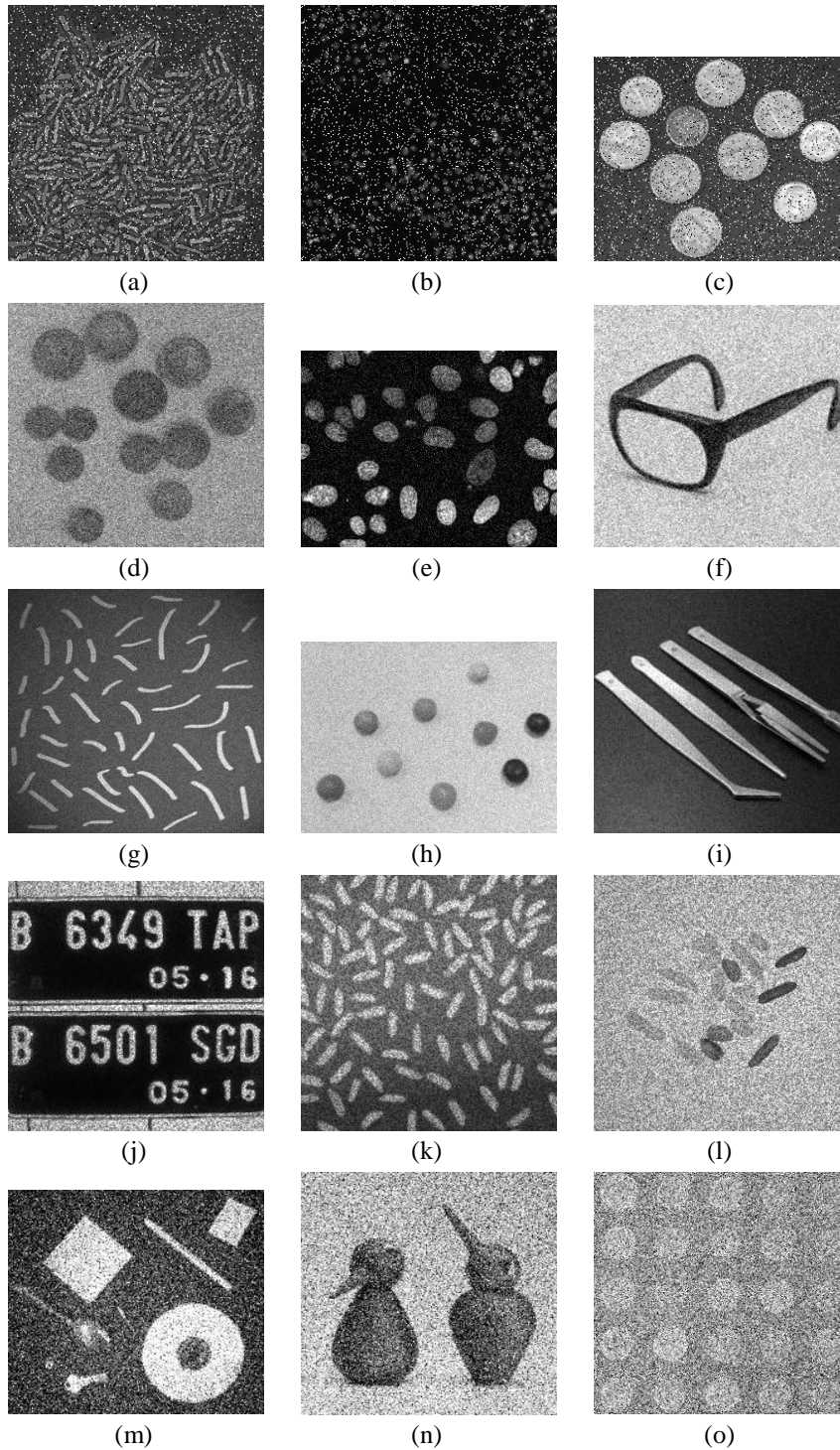
Terdapat 3 masukan yang harus ada pada fungsi (*imnoise*), jika ditulis dalam satu baris perintah Matlab, maka fungsi tersebut adalah sebagai berikut (`I=imnoise(input1,input2,input3)`).

*Input* pertama merupakan nilai intensitas piksel dari citra masukan. Untuk beberapa tipe *noise*, tipe citra masukan yang digunakan dapat berupa: *uint8*, *uint16*, *int16*, *single*, ataupun *double*. Pada penelitian ini, digunakan tipe gambar *uint8* dengan format *.jpg*

*Input* kedua merupakan tipe *noise* yang hendak ditambahkan pada citra masukan. Nilai untuk masukan ini dapat diisi dengan: (*salt&pepper*, *gaussian*, *poisson*, *speckle*, atau *localvar*). Untuk tipe *noise* selain yang telah disebutkan, maka fungsi *imnoise* pada matlab ini tidak dapat digunakan.

*Input* ketiga adalah nilai parameter dari *noise* yang ditambahkan. Pada beberapa *noise* nilai parameter ini harus

diisi. Jika nilai parameter *noise* tidak diisi, maka sistem akan secara otomatis mengisinya dengan nilai *default*. Gambar 4.2 menunjukkan hasil penambahan noise pada citra uji coba.



**Gambar 4.2. Hasil Penambahan Noise pada Citra Uji Coba**

Keterangan dari Gambar 4.2 adalah sebagai berikut.

- Gambar 4.2(a), (b), dan (c) merupakan hasil penambahan noise *salt & pepper* pada masing-masing citra: Cell.jpg, Cell2.jpg, dan Coins.jpg.
- Gambar 4.2(d), (e), dan (f) merupakan hasil penambahan noise *gaussian* pada masing-masing citra: Coins2.jpg, Contour.jpg, dan Eyeglass.jpg.
- Gambar 4.2(g), (h), dan (i) merupakan hasil penambahan noise *poisson* pada masing-masing citra: Lighting.jpg, Marbels.jpg, dan Pincer.jpg.
- Gambar 4.2(j), (k), dan (l) merupakan hasil penambahan noise *speckle* pada masing-masing citra: Plats.jpg, Rice.jpg, dan Rice2.jpg.
- Gambar 4.2(m), (n), dan (o) merupakan hasil penambahan noise *localvar* pada masing-masing citra: Shapessm.jpg, Woodtoys.jpg, dan Yeast.jpg.

#### 4.2.2. Pelaksanaan dan Hasil *Noise Filtering*

Seperti yang telah dijelaskan sebelumnya, proses *noise filtering* pada penelitian ini menggunakan empat metode sebagai pembandingan dengan metode yang diusulkan. Metode-metode tersebut antara lain; Median Filter, Mean Filter, Wiener Filter, dan Gaussian Filter.

Proses implementasi metode menggunakan bahasa pemrograman Matlab lewat fungsi-fungsi yang telah disediakan, selain itu digunakan nilai default untuk setiap parameternya.

Berikut ini adalah fungsi implementasi untuk setiap metode pada bahasa pemrograman Matlab:

- a. *Median Filter*: menggunakan fungsi  $B = \text{medfilt2}(A, [m \ n]);$   
dimana  $A$  adalah intensitas nilai piksel citra input,  $B$  adalah citra output,  $m$  dan  $n$  adalah ukuran *window* yang akan digunakan. Nilai *default* untuk parameter  $m$  dan  $n$  adalah 3. Citra output mempunyai kelas dan ukuran dan sama dengan citra input.
- b. *Mean Filter*: fungsi untuk *mean filter* tidak ada pada Matlab. Oleh karena itu, untuk menggunakan fungsi ini perlu dibuat sebuah matriks berukuran  $m \times n$ , yang nantinya akan dikonvolusi dengan citra masukan.

Nilai dari matriks tersebut adalah rata-rata dari jumlah keseluruhan nilai yang ada didalamnya. Contoh matriks berukuran  $3 \times 3$  untuk metode *mean filter* adalah sebagai berikut.

$$C = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}, \text{ proses konvolusi citra dengan matriks yang telah}$$

dibuat menggunakan fungsi  $B = A * C$ ; dimana  $A$  adalah intensitas nilai piksel dari citra *input*.

- c. *Wiener Filter*: menggunakan fungsi  $J = \text{wiener2}(I, [m \ n], \text{noise})$ ; dimana  $I$  adalah intensitas nilai piksel citra input, menggunakan *window* berukuran  $m$  dan  $n$  mencari *neighboring pixel* dan menghitung standar deviasi. Nilai *default* untuk parameter  $m$  dan  $n$  adalah 3. Noise aditif (*gaussian white noise*) diasumsikan sebagai nilai *default* untuk noise. Citra output mempunyai kelas dan ukuran dan sama dengan citra input.
- d. *Gaussian Filter*: menggunakan fungsi  $B = \text{imgaussfilt}(A, C)$ ; dimana  $A$  adalah intensitas dari nilai piksel citra input,  $B$  adalah citra output, dan  $C$  adalah nilai standar deviasi, nilai *default* untuk standar deviasi yang digunakan adalah 0.5.

Tahap-tahap pelaksanaan pengujian kinerja *noise filtering* telah dijelaskan pada bab sebelumnya. Tahap pertama ada mempersiapkan gambar hasil penambahan *noise*, kemudian gambar dengan *noise* tersebut difilter menggunakan 5 metode. Gambar hasil *noise filtering* dari masing-masing metode akan dibandingkan dengan gambar awal dengan menggunakan parameter MSE dan PSNR untuk mengetahui tingkat kinerja yang diberikan. Semakin besar nilai PSNR yang dihasilkan, maka performa atau kinerja dari metode tersebut semakin baik.

Hasil nilai PSNR untuk kinerja *noise filtering* dari masing-masing metode ditampilkan secara lengkap pada Tabel 4.1, sedangkan untuk rata-rata kinerja dari setiap metode ditampilkan pada grafik di Gambar 4.3.

Hasil visual untuk *filtering noise salt & pepper, gaussian, poisson, speckle*, dan *localvar* dengan metode-metode yang telah dijelaskan, masing-masing ditampilkan pada Gambar 4.6, 4.10, 4.12, 4.14, 4.16, dan 4.18

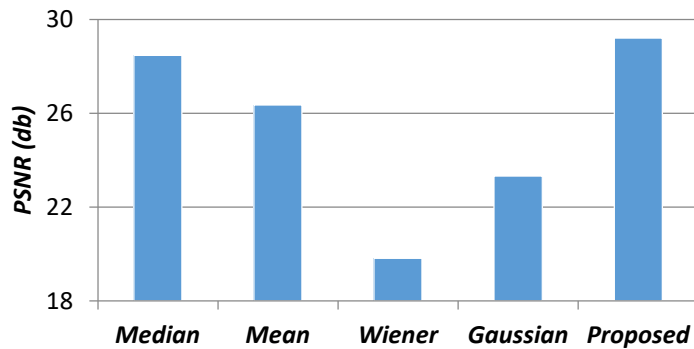
**Tabel 4.1. Hasil Pengukuran PSNR (db)**

No.	Citra	Impulse Noise	Filter				
			Median	Mean	Wiener	Gaussian	SWMF
1	Cell.jpg (460x460)	SP	<b>21,74</b>	18,67	14,94	18,03	20,97
		Gaussian	21,08	19,83	20,21	<b>22,40</b>	20,87
		Poisson	22,77	20,25	<b>29,52</b>	26,73	21,48
		Speckle	21,42	20,06	22,89	<b>24,19</b>	21,05
		Localvar	18,43	18,56	14,26	17,47	<b>19,45</b>
2	Cell2.jpg (500x500)	SP	<b>38,37</b>	21,76	13,56	17,14	34,08
		Gaussian	28,50	29,21	22,09	25,48	<b>30,80</b>
		Poisson	<b>39,18</b>	38,84	35,78	39,15	38,96
		Speckle	36,39	<b>37,93</b>	32,93	36,47	36,19
		Localvar	23,90	21,85	15,68	18,71	<b>25,56</b>
3	Coins.jpg (500x400)	SP	<b>38,10</b>	23,82	15,12	18,83	34,40
		Gaussian	27,29	28,86	20,08	23,88	<b>29,53</b>
		Poisson	34,22	34,41	28,02	31,75	<b>34,65</b>
		Speckle	25,31	<b>28,59</b>	20,03	23,79	28,30
		Localvar	22,04	23,22	14,13	17,91	<b>25,06</b>
4	Coins2.jpg (460x460)	SP	<b>36,99</b>	24,67	15,85	19,66	33,92
		Gaussian	27,53	28,21	20,01	23,84	<b>30,13</b>
		Poisson	33,74	31,98	26,87	30,65	<b>34,76</b>
		Speckle	23,94	27,04	18,48	22,30	<b>27,53</b>
		Localvar	22,08	22,82	13,61	17,44	<b>25,08</b>
5	Contour.jpg (520x400)	SP	<b>33,60</b>	21,67	13,70	17,26	31,18
		Gaussian	28,10	27,36	21,88	24,97	<b>28,51</b>
		Poisson	34,15	33,44	33,01	<b>36,10</b>	33,15
		Speckle	29,71	<b>31,25</b>	25,57	29,16	29,86
		Localvar	23,48	21,21	15,30	18,30	<b>24,18</b>
6	Eyeglass.jpg (400x400)	SP	<b>33,60</b>	22,56	14,69	18,37	31,15
		Gaussian	27,32	27,18	20,17	23,99	<b>29,36</b>
		Poisson	32,39	29,36	25,47	29,25	<b>33,14</b>
		Speckle	21,07	24,36	16,15	19,96	<b>24,68</b>
		Localvar	22,11	22,81	14,53	18,28	<b>24,81</b>
7	Lighting.jpg (500x500)	SP	<b>34,71</b>	23,84	15,26	19,09	32,66
		Gaussian	26,91	27,98	20,03	23,81	<b>28,96</b>
		Poisson	33,32	31,94	28,60	32,03	<b>33,67</b>
		Speckle	26,48	<b>28,98</b>	21,73	25,47	28,58
		Localvar	21,76	22,83	13,92	17,70	<b>24,53</b>
8	Marbels.jpg (480x360)	SP	<b>35,89</b>	23,03	14,93	18,63	31,99
		Gaussian	27,55	27,55	20,11	23,94	<b>30,06</b>
		Poisson	32,72	29,97	25,32	29,14	<b>33,93</b>
		Speckle	21,02	24,42	15,89	19,72	<b>24,91</b>
		Localvar	22,11	22,86	14,33	18,12	<b>25,17</b>

**Tabel 4.1. Hasil Pengukuran PSNR (db)(lanjutan)**

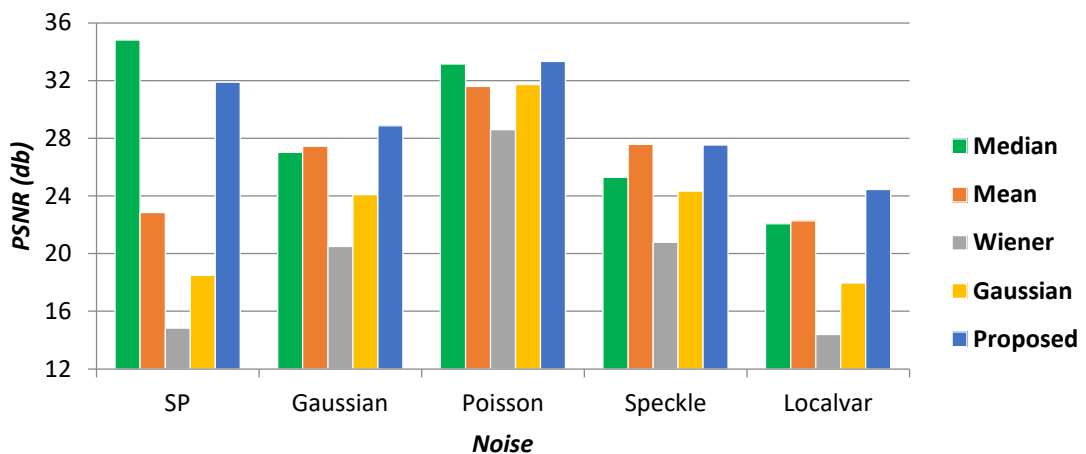
9	Pincer.jpg (400x400)	SP	<b>34,99</b>	22,93	14,70	18,34	32,23
		Gaussian	27,03	28,29	20,36	24,12	<b>28,97</b>
		Poisson	<b>34,30</b>	32,65	29,74	33,09	33,80
		Speckle	27,54	<b>29,58</b>	22,92	26,52	29,21
		Localvar	22,04	22,87	14,51	18,17	<b>24,77</b>
10	Plats.jpg (410x400)	SP	<b>29,37</b>	21,23	14,20	17,74	27,95
		Gaussian	25,78	25,35	20,95	24,42	<b>26,00</b>
		Poisson	30,60	27,07	29,60	<b>32,00</b>	29,32
		Speckle	24,82	<b>25,62</b>	21,19	24,72	25,35
		Localvar	21,68	21,34	14,93	18,33	<b>22,65</b>
11	Rice.jpg (400x400)	SP	<b>34,91</b>	24,09	15,53	19,27	32,64
		Gaussian	27,05	28,09	20,03	23,84	<b>28,59</b>
		Poisson	<b>32,91</b>	31,95	27,66	31,34	32,57
		Speckle	25,05	<b>27,87</b>	19,68	23,50	27,33
		Localvar	21,82	22,79	13,80	17,59	<b>24,32</b>
12	Rice2.jpg (460x460)	SP	<b>36,62</b>	23,97	15,48	19,27	33,18
		Gaussian	27,42	27,81	19,99	23,82	<b>29,97</b>
		Poisson	32,89	30,79	25,82	29,60	<b>34,27</b>
		Speckle	21,98	25,18	16,47	20,30	<b>25,76</b>
		Localvar	21,96	22,81	13,84	17,67	<b>24,98</b>
13	Rontgen.jpg (400x400)	SP	<b>38,76</b>	22,30	13,94	17,56	34,29
		Gaussian	28,33	29,00	21,47	24,97	<b>30,83</b>
		Poisson	37,18	36,99	31,46	35,15	<b>37,36</b>
		Speckle	29,10	<b>32,30</b>	23,79	27,60	31,99
		Localvar	23,43	22,05	15,08	18,35	<b>25,58</b>
14	Shapessm.jpg (410x360)	SP	<b>37,86</b>	23,22	14,59	18,35	34,47
		Gaussian	27,52	28,78	20,38	24,10	<b>29,76</b>
		Poisson	34,52	34,21	28,44	31,98	<b>35,20</b>
		Speckle	25,63	26,91	20,57	23,76	<b>27,69</b>
		Localvar	22,34	22,92	14,44	18,09	<b>25,14</b>
15	Woodtoys.jpg (410x410)	SP	<b>35,59</b>	23,34	15,13	18,93	31,79
		Gaussian	27,39	27,41	20,10	23,92	<b>29,48</b>
		Poisson	32,79	29,99	26,06	29,81	<b>33,27</b>
		Speckle	22,21	25,20	16,98	20,79	<b>25,69</b>
		Localvar	22,01	22,73	14,11	17,91	<b>24,72</b>
16	Yeast.jpg (500x400)	SP	<b>36,06</b>	24,50	15,75	19,53	33,37
		Gaussian	27,38	28,23	20,03	23,85	<b>29,87</b>
		Poisson	32,72	31,80	26,21	29,98	<b>33,68</b>
		Speckle	22,79	26,05	17,29	21,12	<b>26,39</b>
		Localvar	22,00	22,78	13,64	17,46	<b>24,97</b>
<b>Average</b>			<b>28,47</b>	<b>26,35</b>	<b>19,82</b>	<b>23,32</b>	<b>29,21</b>





**Gambar 4.3. Rata-Rata Keseluruhan PSNR**

Berdasarkan Tabel 4.1, dapat dihitung juga kinerja *noise filtering* berdasarkan jenis noise. Gambar 4.4 menampilkan grafik perbandingan nilai PSNR masing-masing metode untuk setiap jenis *noise*.

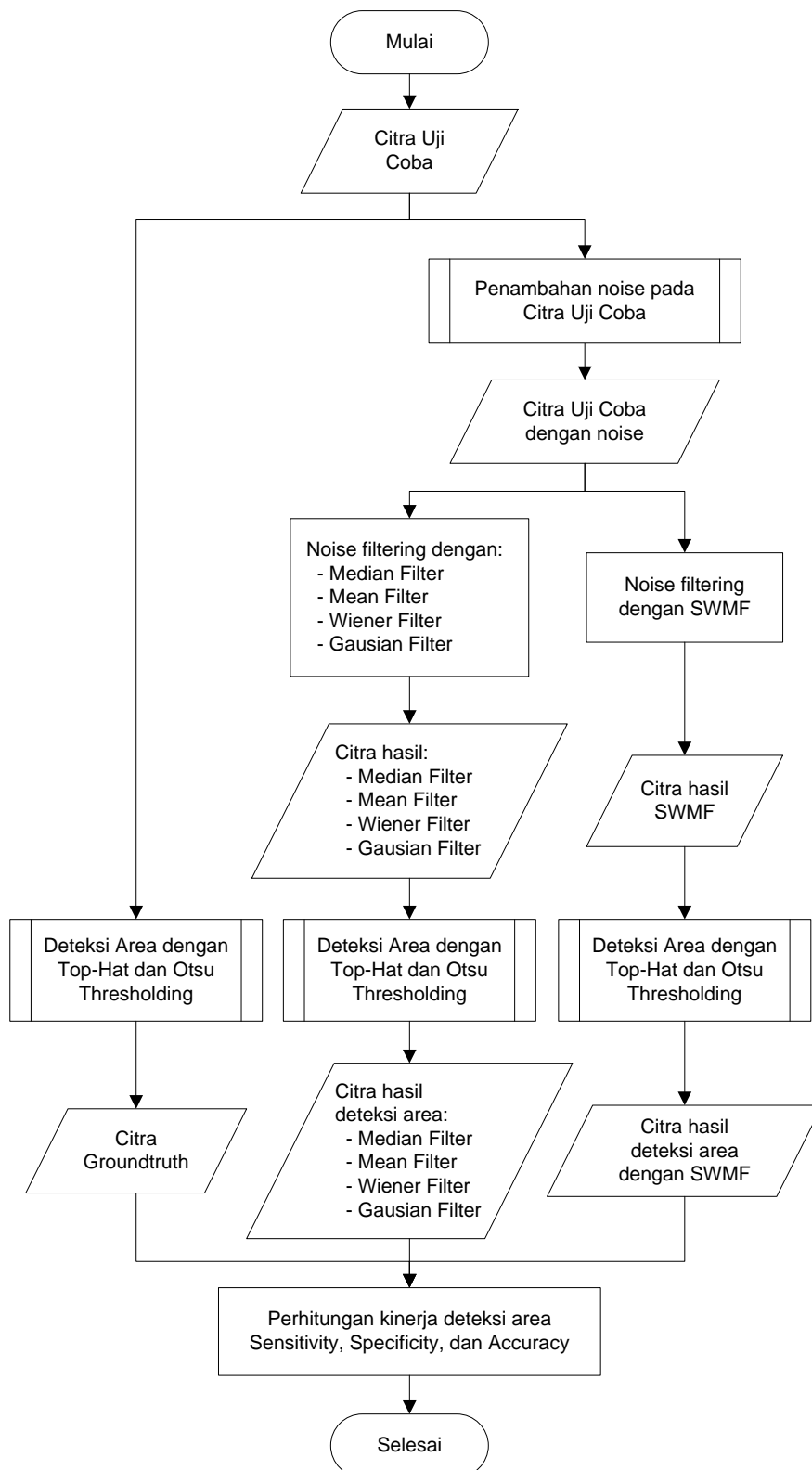


**Gambar 4.4. Rata-Rata PSNR berdasarkan Jenis Noise**

#### 4.2.3. Pelaksanaan dan Hasil Deteksi Area

Pada pelaksanaan dan hasil, proses segmentasi untuk deteksi area menggunakan algoritma Top-Hat dan citra yang digunakan adalah citra hasil *noise filtering* dari proses sebelumnya. Hasil deteksi area dari citra hasil *noise filtering* tersebut masing-masing akan dibandingkan dengan citra *groundtruth* untuk menghitung *sensitivity*, *specificity*, dan *accuracy* untuk masing-masing metode. Citra *groundtruth* merupakan hasil deteksi area pada citra awal (citra sebelum diberi *noise*).

Untuk lebih mudah dipahami, alur kerja, proses serta data-data yang digunakan pada deteksi area ditampilkan secara detail pada Gambar 4.5.

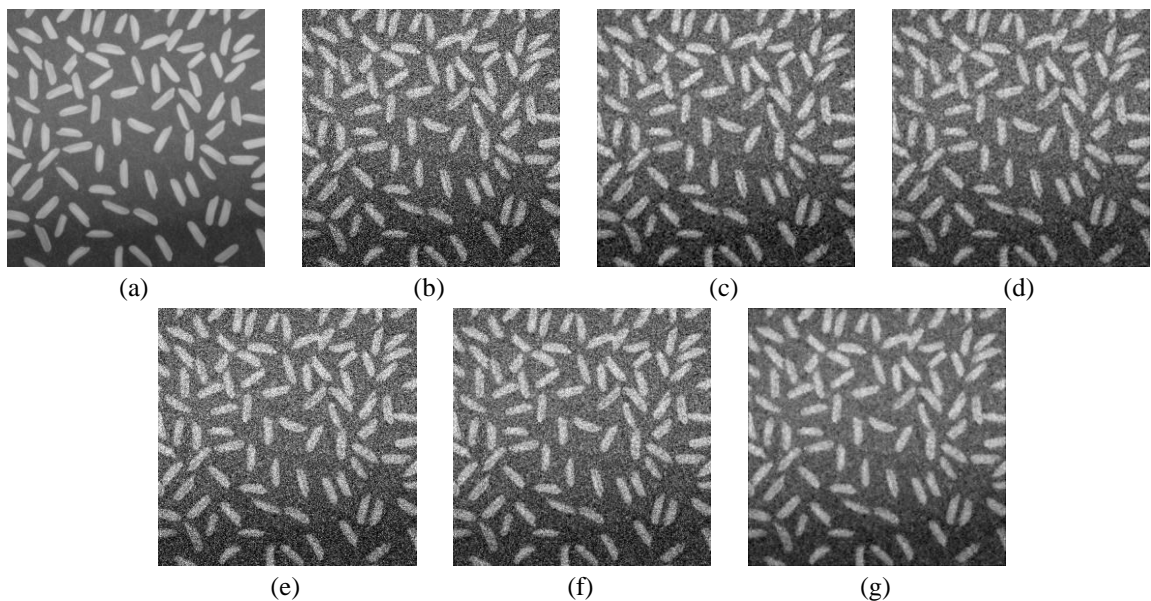


**Gambar 4.5. Tahapan Proses Pengujian untuk Deteksi Area**

Proses implementasi algoritma Top-Hat pada bahasa pemrograman Matlab adalah melalui tahap-tahap berikut:

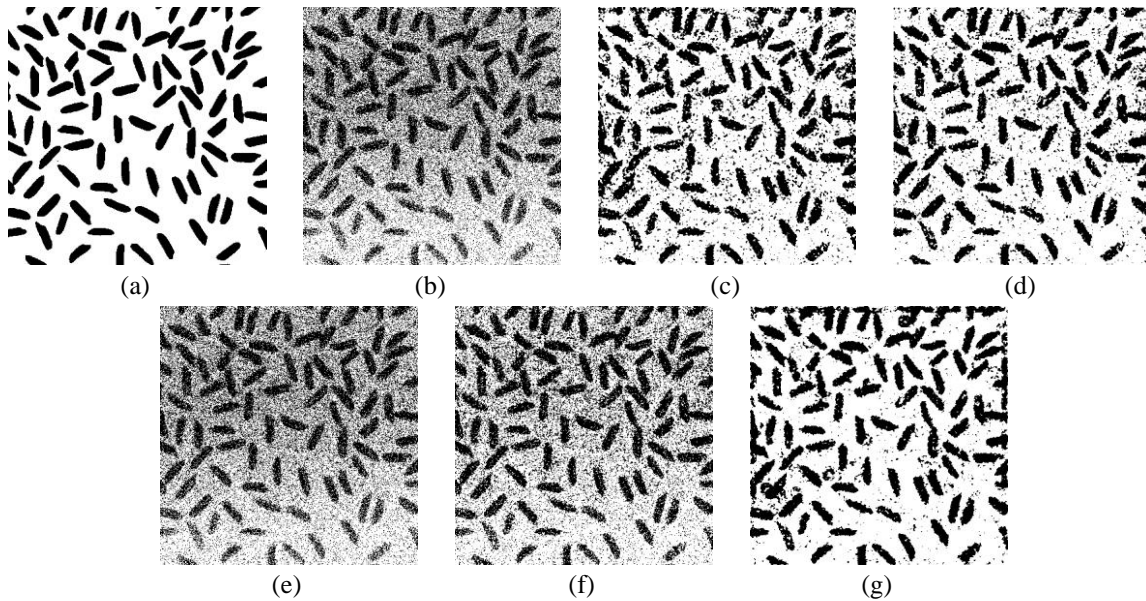
- a. Menentukan struktur elemen dengan fungsi `se = strel('disk',10);` dimana *strel* adalah fungsi matlab untuk membuat struktur elemen, input pertama adalah bentuk struktur elemen (lingkaran), dan input kedua adalah ukuran dimensi *se* (10×10).
- b. Erosi citra awal *im* menggunakan fungsi `ero = imerode(im,se);` dimana *ero* adalah citra hasil erosi menggunakan struktur elemen *se*.
- c. Mengurangi citra awal *im* dengan citra hasil erosi *ero* menggunakan fungsi `min = im - ero;` dimana *min* adalah citra hasil pengurangan.
- d. Binarisasi citra hasil pengurangan *min* dengan menggunakan fungsi `bin = imbinarize(min,graythresh(min));` dimana *bin* adalah citra hasil binarisasi, dan *graythresh* adalah fungsi untuk menentukan nilai tresholding secara otomatis menggunakan metode Otsu.
- e. Invers citra hasil binarisasi *bin* menggunakan fungsi `in = imcomplement(bin);` dimana *in* adalah citra hasil inverse.

Gambar 4.6 dan Gambar 4.7 menampilkan contoh visual untuk hasil deteksi area yang diberikan pada berbagai metode. Sebagai contoh, digunakan gambar Rice.jpg dengan *noise localvar*.



**Gambar 4.6. Hasil *Filtering Noise Localvar* pada Citra Rice**

Gambar 4.6(a) adalah citra awal, (b) adalah citra setelah ditambahkan *noise localvar*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan. Hasil deteksi area dari masing-masing Gambar 4.6 adalah sebagai berikut



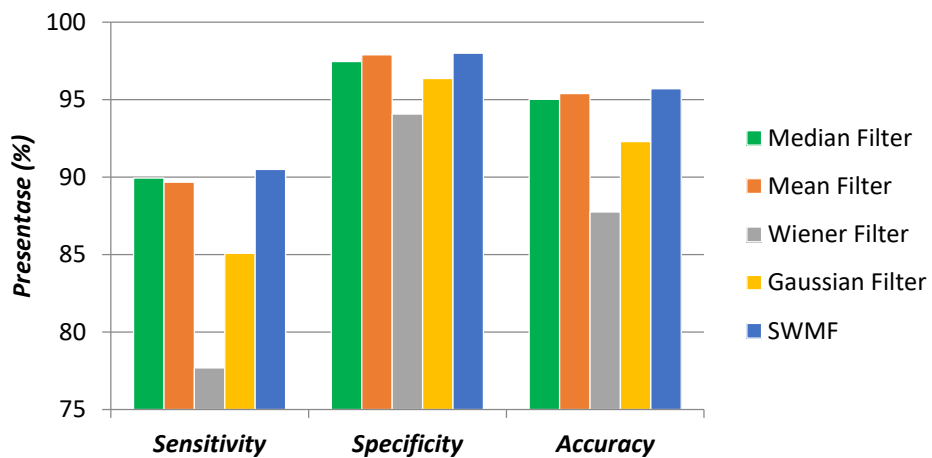
**Gambar 4.7. Hasil Deteksi Area pada Citra Rice dengan *Noise Localvar***

Gambar 4.7(a) adalah hasil deteksi area citra awal, (b) adalah hasil deteksi area citra setelah ditambahkan *noise localvar*, (c) hasil deteksi area pada citra hasil Median Filter, (d) hasil deteksi area pada citra hasil Mean Filter, (e) hasil deteksi area pada citra Wiener Filter, (f) hasil deteksi area pada citra hasil Gaussian Filter, dan (g) hasil deteksi area pada citra hasil metode SWMF yang diusulkan.

Tabel 4.2 merupakan hasil pengukuran deteksi area untuk masing-masing metode. Nilai pada tabel tersebut merupakan nilai rata-rata presentase tingkat *sensitivity*, *specificity*, dan *accuracy* dari 16 citra yang diuji. Rata-rata kinerja dari Tabel 4.2 kemudian dibuat kedalam diagram batang pada Gambar 4.8 untuk melihat perbandingan kinerja dari keseluruhan metode.

**Tabel 4.2. Hasil Pengukuran Sensitivity, Specificity, dan Accuracy (%)**

No.	Metode	Impulse Noise	Sensitivity	Specificity	Accuracy
1.	Median Filter	SP	95,10	99,44	98,22
		Gaussian	94,25	98,24	97,15
		Poisson	97,42	99,14	98,68
		Speckle	95,17	96,20	95,93
		Localvar	67,74	94,15	85,08
	Average	<b>89,94</b>	<b>97,44</b>	<b>95,01</b>	
2.	Mean Filter	SP	89,64	98,14	95,69
		Gaussian	94,78	98,53	97,50
		Poisson	96,19	99,18	98,35
		Speckle	94,32	98,25	97,17
		Localvar	73,35	95,38	88,17
	Average	<b>89,66</b>	<b>97,89</b>	<b>95,38</b>	
3.	Wiener Filter	SP	89,31	95,38	93,76
		Gaussian	66,75	95,96	85,22
		Poisson	96,48	98,68	98,09
		Speckle	86,24	90,01	89,16
		Localvar	49,61	90,33	72,54
	Average	<b>77,68</b>	<b>94,07</b>	<b>87,75</b>	
4.	Gaussian Filter	SP	89,12	97,04	94,82
		Gaussian	87,97	97,72	94,90
		Poisson	97,91	99,19	98,84
		Speckle	94,18	95,35	95,05
		Localvar	56,16	92,44	77,84
	Average	<b>85,07</b>	<b>96,35</b>	<b>92,29</b>	
5.	SWMF	SP	94,75	99,47	98,13
		Gaussian	91,63	98,37	96,46
		Poisson	96,87	99,00	98,42
		Speckle	94,30	97,42	96,59
		Localvar	74,83	95,64	88,92
	Average	<b>90,48</b>	<b>97,98</b>	<b>95,70</b>	



**Gambar 4.8. Rata-Rata Sensitivity, Specificity, dan Accuracy**

#### 4.2.4. Pelaksanaan dan Hasil Deteksi Garis

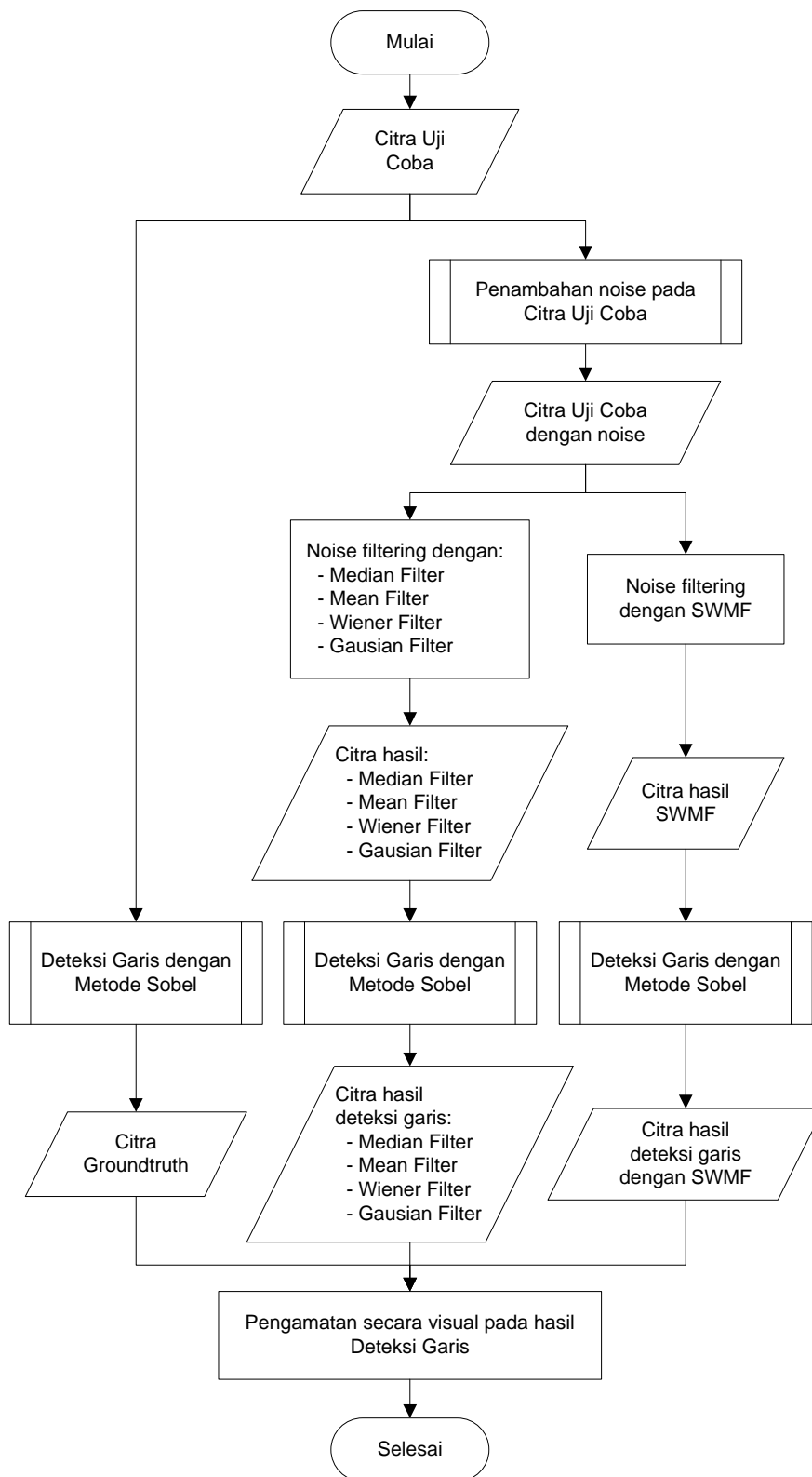
Pada pelaksanaannya, proses segmentasi untuk deteksi garis menggunakan metode Sobel, dan evaluasi untuk hasil deteksi garis dilakukan dengan pengamatan secara visual. Proses implementasi metode Sobel pada bahasa pemrograman Matlab adalah melalui langkah-langkah sebagai berikut.

- a. Menerapkan algoritma deteksi garis Sobel pada citra awal *im* dengan menggunakan fungsi `sob = edge(im,'sobel')`; dimana *sob* adalah citra hasil deteksi garis dengan metode Sobel.
- b. Binarisasi citra hasil deteksi garis *sob* dengan menggunakan fungsi `bin = imbinarize(sob,0.5)`; dimana *bin* adalah citra hasil binarisasi, dan nilai thresholding ditentukan secara manual menggunakan nilai tengah yaitu 128.
- c. Invers citra hasil binarisasi *bin* menggunakan fungsi `in = imcomplement(bin)`; dimana *in* adalah citra hasil inverse.

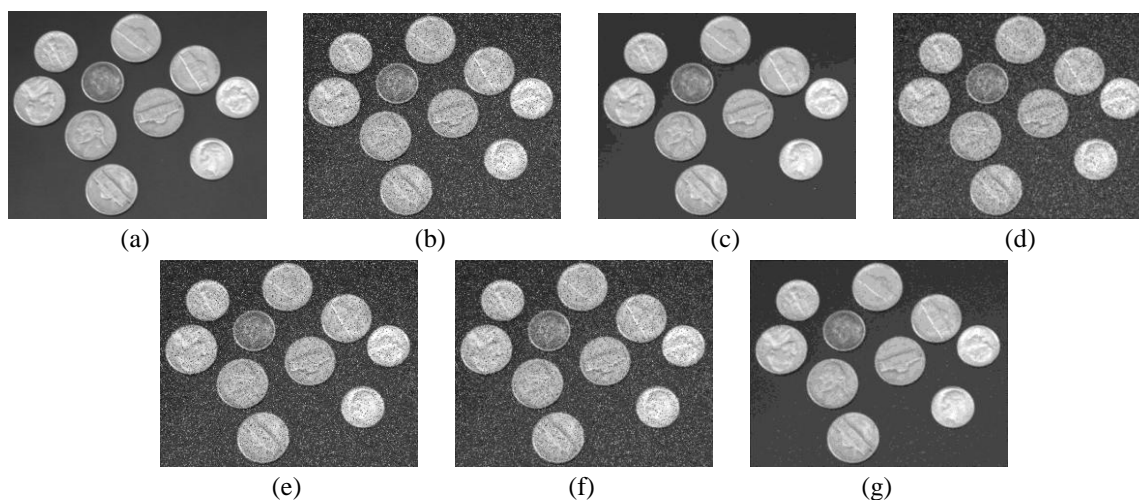
Mengingat jumlah data uji yang banyak dan masing-masing data tersebut harus diberi *noise* lalu difilter dengan masing-masing metode (jumlah data uji  $16 \times$  jumlah *noise*  $5 \times$  jumlah metode  $5 = 400$  hasil), maka untuk menampilkan hasil deteksi garis secara visual, hanya digunakan beberapa gambar yang memiliki karakteristik yang berbeda dan mewakili gambar yang lainnya.

Gambar yang digunakan pada tahap deteksi garis seperti Coins.jpg, Lighting.jpg, Eyeglass.jpg, Woodtoys.jpg, dan Rice.jpg. Gambar-gambar ini memiliki tepian objek (garis) yang jelas.

Tahap pelaksanaan untuk deteksi garis hampir sama dengan deteksi area. Metode sobel diterapkan pada citra awal (citra sebelum penambahan *noise*), citra hasil penambahan *noise*, dan citra hasil *noise filtering* (menggunakan 5 metode yang telah dijelaskan). Untuk lebih mudah dipahami, alur kerja, proses serta data-data yang digunakan pada deteksi area ditampilkan secara detail pada Gambar 4.9

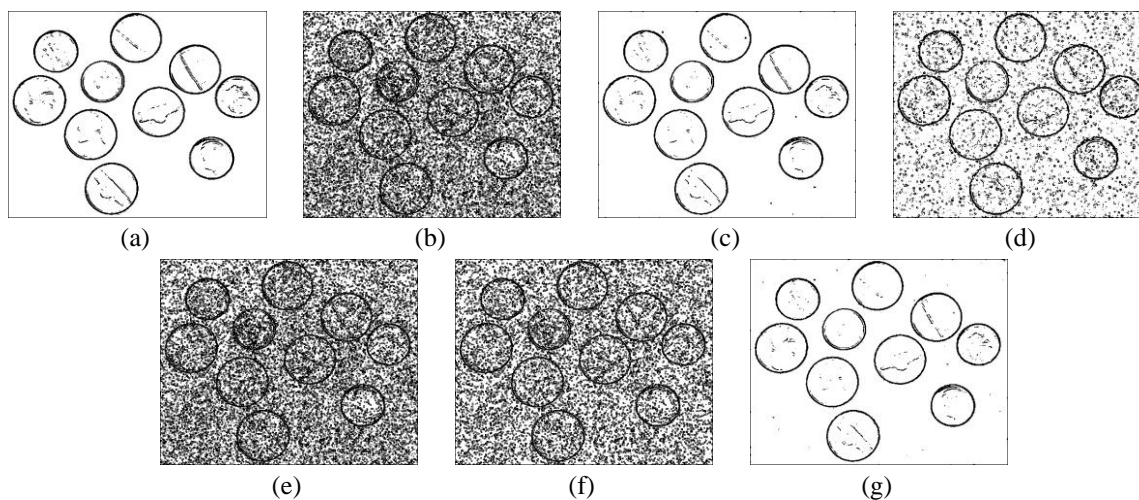


**Gambar 4.9. Tahapan Proses Pengujian untuk Deteksi Garis**



**Gambar 4.10. Hasil *Filtering Noise Salt & Pepper* pada Citra Coins**

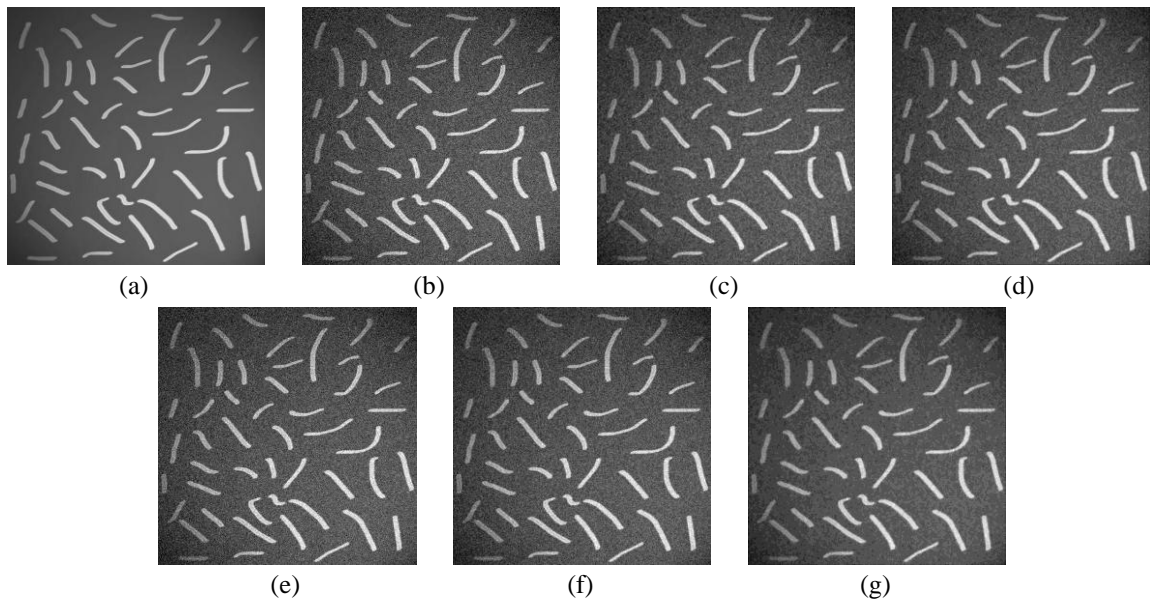
Gambar 4.10(a) adalah citra awal, (b) adalah citra setelah ditambahkan *noise salt & pepper*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan. Hasil deteksi garis dari masing-masing Gambar 4.10 adalah sebagai berikut



**Gambar 4.11. Hasil Deteksi Garis pada Citra Coins dengan *Noise Salt & Pepper***

Gambar 4.11(a) adalah hasil deteksi garis citra awal, (b) adalah hasil deteksi garis citra setelah ditambahkan *noise salt & pepper*, (c) hasil deteksi garis pada citra hasil Median Filter, (d) hasil deteksi garis pada citra hasil Mean Filter, (e) Wiener Filter, (f) Gaussian Filter, dan (g) hasil deteksi area pada citra hasil metode SWMF yang diusulkan.

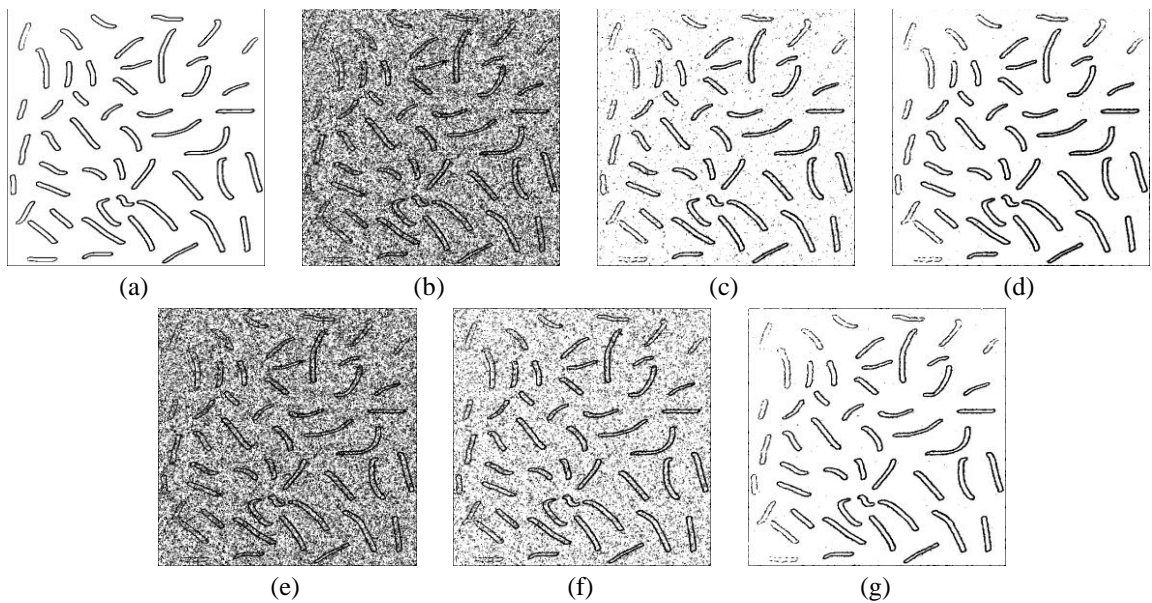




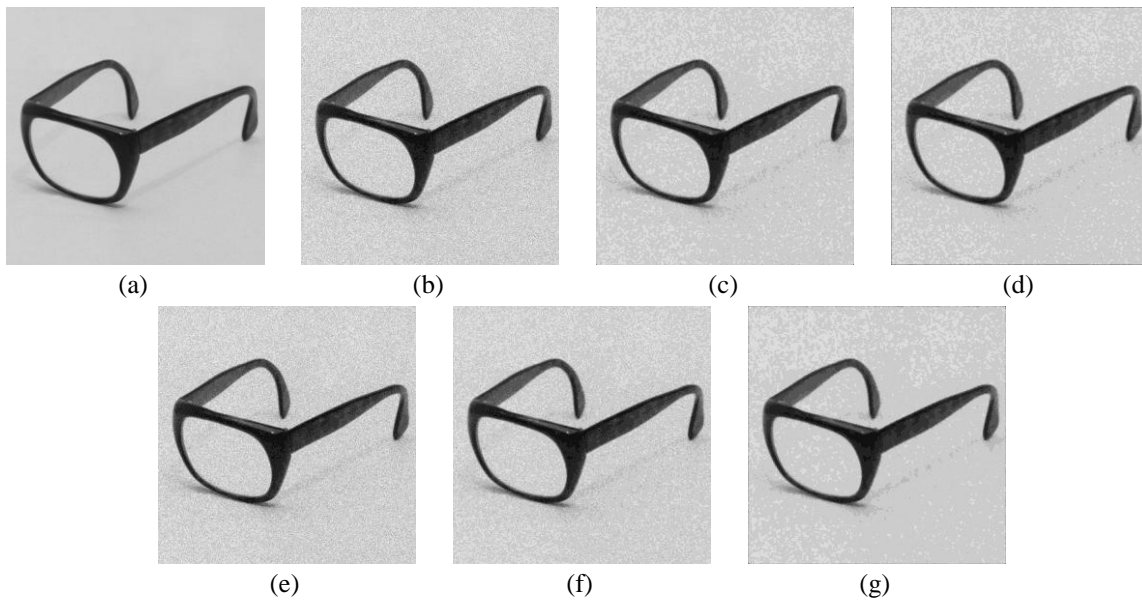
**Gambar 4.12. Hasil *Filtering Noise Gaussian* pada Citra *Lighting***

Gambar 4.12(a) adalah citra awal, (b) citra setelah ditambahkan *noise gaussian*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan.

Hasil deteksi garis dari masing-masing Gambar 4.12 adalah sebagai berikut.



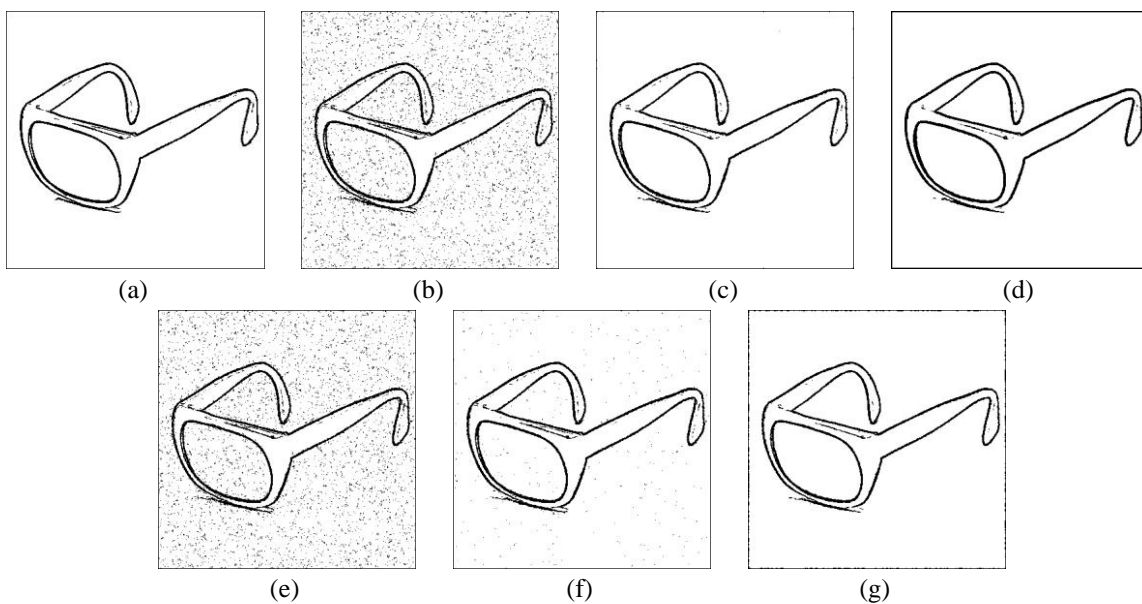
**Gambar 4.13. Hasil Deteksi Garis pada Citra *Lighting* dengan *Noise Gaussian***



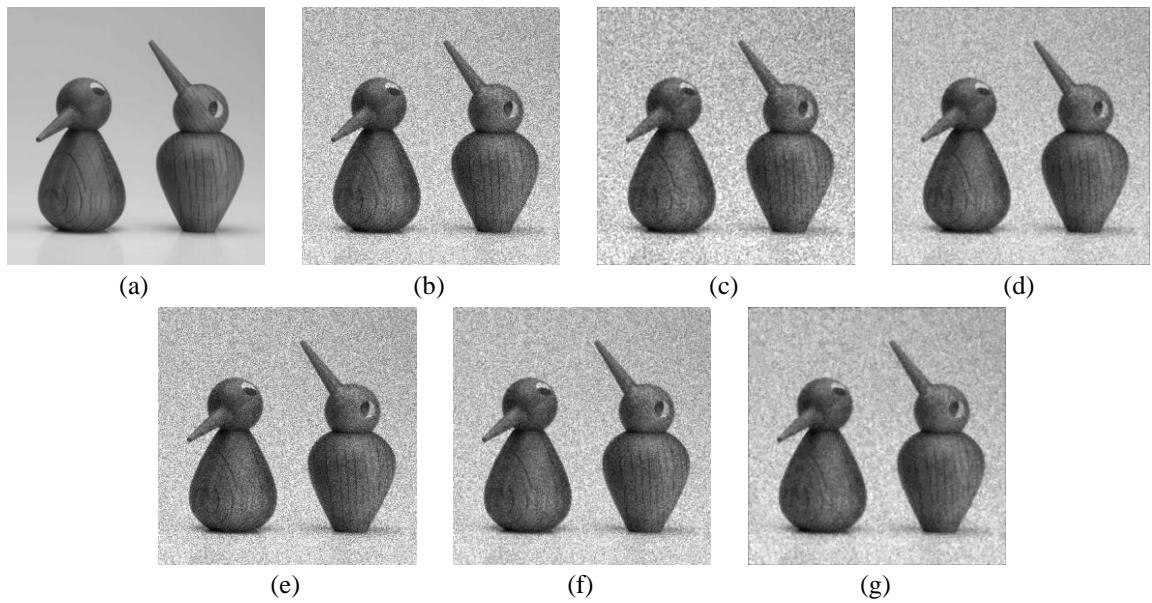
**Gambar 4.14. Hasil *Filtering Noise Poisson* pada Citra Eyeglass**

Gambar 4.14(a) adalah citra awal, (b) citra setelah ditambahkan *noise poisson*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan.

Hasil deteksi garis dari masing-masing Gambar 4.14 adalah sebagai berikut.



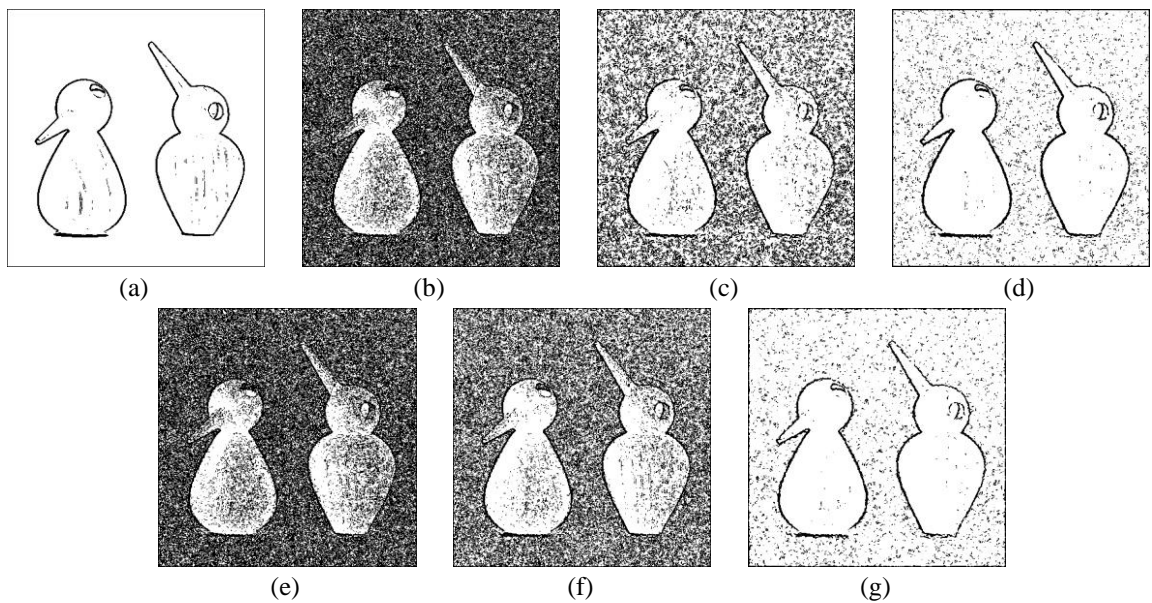
**Gambar 4.15. Hasil Deteksi Garis pada Citra Eyeglass dengan *Noise Poisson***



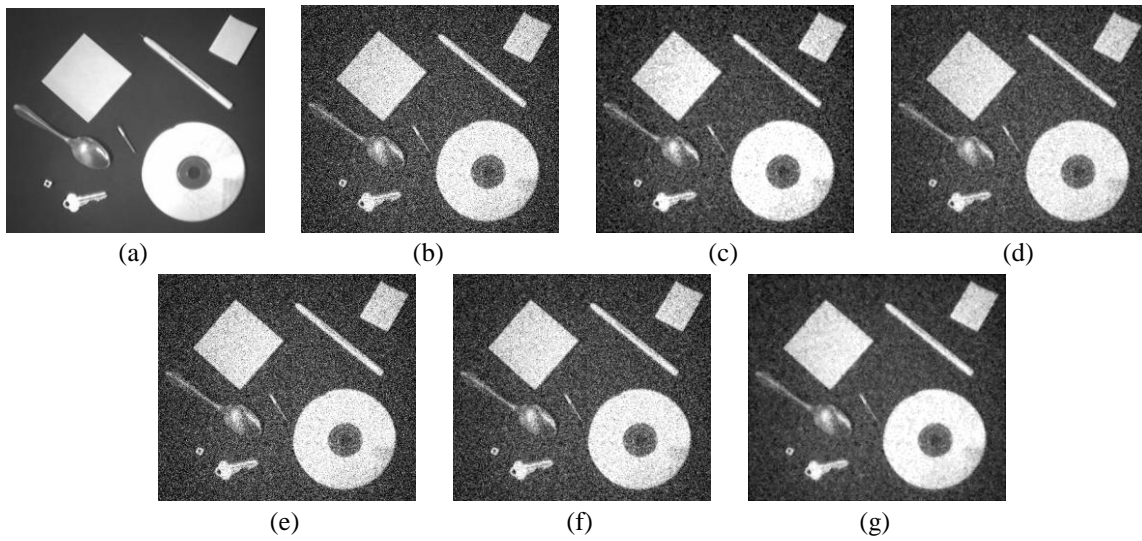
**Gambar 4.16. Hasil *Filtering Noise Speckle* pada Citra Woodtoys**

Gambar 4.16(a) adalah citra awal, (b) citra setelah ditambahkan *noise speckle*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan.

Hasil deteksi garis dari masing-masing Gambar 4.16 adalah sebagai berikut.

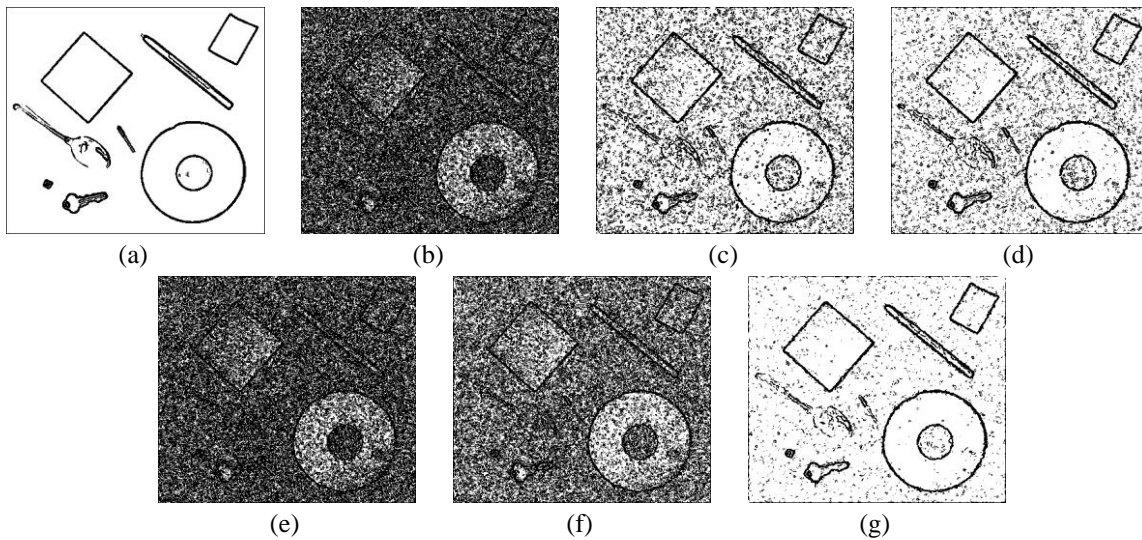


**Gambar 4.17. Hasil Deteksi Garis pada Citra Woodtoys dengan *Noise Speckle***



**Gambar 4.18. Hasil *Filtering Noise Localvar* pada Citra Shapessm**

Gambar 4.18(a) adalah citra awal, (b) citra setelah ditambahkan *noise localvar*, (c) citra hasil Median Filter, (d) citra hasil Mean Filter, (e) citra hasil Wiener Filter, (f) citra hasil Gaussian Filter, dan (g) citra hasil metode SWMF yang diusulkan. Hasil deteksi garis dari masing-masing Gambar 4.18 adalah sebagai berikut.



**Gambar 4.19. Hasil Deteksi Garis pada Citra Shapessm dengan *Noise Localvar***

Gambar 4.19(a) adalah hasil deteksi garis citra awal, (b) hasil deteksi garis citra dengan *noise localvar*, (c) hasil deteksi garis pada citra hasil Median Filter, (d) Mean Filter, (e) Wiener Filter, (f) Gaussian Filter, dan (g) hasil deteksi area pada citra hasil metode SWMF yang diusulkan.

### 4.3. Evaluasi Hasil Uji Coba

#### 4.3.1. Evaluasi Hasil *Noise Filtering*

Berdasarkan pelaksanaan dan hasil uji coba untuk *noise filtering*, hasil evaluasi adalah sebagai berikut.

- a. Metode *Soft Weighted Median Filter* yang diusulkan pada penelitian memiliki rata-rata nilai PSNR paling tinggi dibandingkan metode lainnya yaitu sebesar 29,21
- b. Metode *Median Filter* dan *Mean Filter* masing-masing berada pada urutan 2 dan 3, dengan rata-rata nilai PSNR 28,47 untuk metode *Median Filter*, dan 26,35 untuk metode *Mean Filter*.
- c. Dari 16 citra yang diuji, metode *Soft Weighted Median Filter* selalu memiliki nilai PSNR yang paling tinggi pada citra dengan *noise gaussian* dan *noise localvar*.
- d. Dari 16 citra yang diuji, metode *Median Filter* selalu memiliki nilai PSNR yang paling tinggi pada citra dengan *noise salt & pepper*.
- e. Berdasarkan jenis *noise*, grafik rata-rata kinerja *noise filtering* menunjukkan bahwa metode *Soft Weighted Median Filter* yang diusulkan memiliki kinerja paling baik pada *noise gaussian*, *poisson*, dan *localvar*.
- f. Berdasarkan jenis *noise*, grafik rata-rata kinerja *noise filtering* menunjukkan bahwa metode *Median Filter* memiliki kinerja paling baik pada *noise salt & pepper*.
- g. Berdasarkan jenis *noise*, grafik rata-rata kinerja *noise filtering* menunjukkan bahwa metode *Mean Filter* memiliki kinerja paling baik pada *noise speckle*.

#### 4.3.2. Evaluasi Hasil Deteksi Area

Berdasarkan pelaksanaan dan hasil uji coba untuk deteksi area, hasil evaluasi adalah sebagai berikut.

- a. Untuk tingkat *sensitivity*, metode *Soft Weighted Median Filter* yang diusulkan pada penelitian memiliki rata-rata nilai presentase paling tinggi yaitu sebesar 90,48%, diikuti metode *Median Filter* pada urutan kedua dengan nilai presentase sebesar 89,94%, dan metode *Mean Filter* di urutan ketiga dengan nilai presentase sebesar 89,66%.

- b. Untuk tingkat *specificity*, metode *Soft Weighted Median Filter* juga memiliki rata-rata nilai presentase paling tinggi yaitu sebesar 97,98%, diikuti metode *Mean Filter* pada urutan kedua dengan nilai presentase sebesar 97,89%, dan metode Median Filter di urutan ketiga dengan nilai presentase sebesar 97,44%
- c. Untuk tingkat *accuracy*, metode *Soft Weighted Median Filter* juga tetap memiliki rata-rata nilai presentase paling tinggi yaitu sebesar 95,70%, diikuti metode *Mean Filter* pada urutan kedua dengan nilai presentase sebesar 95,38%, dan metode Median Filter di urutan ketiga dengan nilai presentase sebesar 95,01%.

#### **4.3.3. Evaluasi Hasil Deteksi Garis**

Berdasarkan pengamatan secara visual untuk hasil uji coba pada deteksi garis pada Gambar 4.9 sampai Gambar 4.13, hasil evaluasi adalah sebagai berikut.

- a. Pada Gambar dengan *noise salt & pepper*, metode *Median Filter* memberikan hasil yang cukup efektif. Terlihat bahwa pada background gambar bebas dari titik atau noda hitam. Hasil serupa juga ada pada gambar hasil metode SWMF, metode ini juga mampu mendeteksi garis dan mampu mengurangi bintik hitam pada background gambar.
- b. Pada Gambar dengan *noise gaussian*, metode SWMF memberikan hasil yang cukup efektif. Dari semua hasil yang didapatkan, background gambar hasil metode SWMF mempunyai tingkat kerusakan yang paling sedikit. Hasil serupa juga terjadi pada Gambar dengan *noise poisson. speckle* dan *localvar*.

## **BAB 5**

### **PENUTUP**

#### **5.1. Kesimpulan**

Berdasarkan hasil uji coba, pembahasan, dan temuan yang sebelumnya sudah disesuaikan dengan tujuan penelitian, maka secara keseluruhan penelitian ini dapat disimpulkan sebagai berikut.

1. Proses penghilangan *noise* pada tahap awal pengolahan citra sangat berpengaruh terhadap hasil segmentasi dan proses-proses selanjutnya. Metode yang akan digunakan untuk menghilangkan *noise* harus sesuai dan tepat sehingga dapat memberikan hasil segmentasi sesuai yang diharapkan.
2. Berdasarkan pengamatan pada *local histogram* dari *window 3×3*, *noise* pada citra menghasilkan rentang atau jarak antara nilai piksel dalam *window* tersebut yang sebelumnya kecil (berdekatan) menjadi besar (berjauhan).
3. Metode *Soft Weighted Median Filter* mampu meningkatkan kualitas segmentasi citra dengan mengurangi *fixed value noise* maupun *random valued noise* pada citra.
4. Berdasarkan hasil uji coba pada tahap *noise filtering*, metode *Soft Weighted Median Filter* memberikan hasil yang lebih baik dibandingkan dengan metode lainnya dengan nilai PSNR sebesar 29,21 db. Metode ini bekerja efektif pada gambar dengan *random value noised*, namun untuk *fixed valued noise* seperti *noise salt & pepper*, metode ini masih kurang dengan *Median Filter*.
5. Berdasarkan hasil uji coba pada tahap deteksi area menggunakan Transformasi Top-Hat, metode *Soft Weighted Median Filter* juga memberikan hasil yang cukup efektif. Metode ini menghasilkan rata-rata tingkat *accuracy* yang lebih tinggi dibandingkan metode lainnya, hal yang sama juga terjadi pada tingkat *sensitivity* dan *specificity* yang dihasilkan.
6. Berdasarkan hasil uji coba pada tahap deteksi garis menggunakan metode Sobel, metode *Soft Weighted Median Filter* mampu meningkatkan hasil deteksi garis dari citra yang sebelumnya diberi *noise*. Untuk *noise* dengan tipe nilai *random*, hasil deteksi garis dengan metode ini selalu lebih baik dibandingkan metode lainnya.

## 5.2. Saran

Berdasarkan hasil yang diperoleh dari penelitian ini, saran yang diperlukan untuk perbaikan sistem maupun untuk penelitian selanjutnya adalah sebagai berikut.

1. Diperlukan algoritma untuk membedakan antara *fixed valued noise* dengan *random valued noise*.
2. Ukuran window yang akan digunakan diharapkan dapat fleksibel sesuai dengan kondisi citra dan jenis *noise* yang dideteksi.



## DAFTAR PUSTAKA

- [1] P. Zhang and F. Li, "A New Adaptive Weighted Mean Filter for Removing Salt-and-Pepper Noise," *IEEE Signal Process. Lett.*, vol. 21, no. 10, pp. 1280–1283, 2014.
- [2] Ö. T. Cengiz Kandemir, Cem Kalyoncu, "A Weighted Mean Filter with Spatial-Bias Elimination for Impulse Noise Removal," *Digit. Signal Process.*, vol. 46, pp. 164–174, 2015.
- [3] C. Kalyoncu, Ö. Toygar, and H. Demirel, "Interpolation-based Impulse Noise Removal," *IET Image Process.*, vol. 7, no. 8, pp. 777–785, 2013.
- [4] N. Chithirala, A. Radhakrishnan, and P. O. Amritanagar, "Weighted Mean Filter for Removal of High Density Salt and Pepper Noise," in *Advanced Computing and Communication Systems (ICACCS)*, 2016, pp. 1–4.
- [5] H. Sun, L. Zhang, and X. Jin, "An Image Denoising Method which Combines Adaptive Median Filter with Weighting Mean Filter," in *Measurement, Information and Control (MIC)*, 2012, no. Mic, pp. 392–396.
- [6] L. Liu, C. L. P. Chen, Y. Zhou, and X. You, "A New Weighted Mean Filter with A Two-Phase Detector for Removing Impulse Noise," *Inf. Sci. (Ny)*, vol. 315, pp. 1–16, 2015.
- [7] Y. Wang, J. Wang, X. Song, and L. Han, "An Efficient Adaptive Fuzzy Switching Weighted Mean Filter for Salt-and-Pepper Noise Removal," *IEEE Signal Process. Lett.*, vol. 23, no. 11, pp. 1582–1586, 2016.
- [8] C. Lee, "Adaptive Weighted Fuzzy Mean Filter," *Fuzzy Syst.*, vol. 2, pp. 2110–2116, 1996.
- [9] C. Lu, Y. Chen, L. Wang, and C. Chang, "Removal of Salt and Pepper Noise in Corrupted Image using Three Values Weighted Approach with Variable Size Window," *Pattern Recognit. Lett.*, vol. 80, pp. 188–199, 2016.
- [10] G. Xu and Y. Lin, "An Efficient Restoration Algorithm for Images Corrupted with Salt and Pepper Noise," in *BioMedical Engineering and Informatics*, 2016, vol. 9, pp. 184–188.