ANALISIS PERBANDINGAN SPATIAL MEDIAN FILTER DAN ADAPTIVE NOISE UNTUK MENGHILANGKAN SALT AND PEPPER PADA CITRA DIGITAL

SKRIPSI

Disusun dan Diajukan Untuk Memenuhi Persyaratan Ujian Akhir Memperoleh
Gelar Sarjana Komputer pada Fakultas Sains dan Teknologi
Universitas Pembangunan Panca Budi
Medan

OLEH :

NAMA	: RENDY WIDANA
NPM	: 1514370384
PRODI	: SISTEM KOMPUTER

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS PEMBANGUNAN PANCA BUDI
MEDAN
2020

ABSTRAK

Noise pada citra tidak hanya terjadi karena ketidaksempurnaan dalam proses pengambilan gambar, tetapi bisa juga disebabkan oleh kotoran-kotoran yang terjadi pada citra. Dalam tugas akhir ini *noise* yang digunakan adalah *impulse noise* (*salt and pepper*). *Impulse noise* biasanya terjadi selama transisi citra. *Noise* ini tampak sebagai impuls-impuls hitam atau putih diatas citra. *Impulse noise* ini dapat terjadi karena *error bit* acak pada saluran komunikasi. Dalam literatur, dapat ditemukan berbagai algoritma yang dapat digunakan untuk menghilangkan *salt and pepper noise*, beberapa diantaranya adalah *spatial median filter* dan *adaptive noise reduction*.

Perangkat lunak yang dirancang ini melakukan beberapa tahapan proses yaitu dimulai dari proses menentukan sebuah piksel adalah *noise* atau tidak berdasarkan pada nilai *threshold* yang diberikan. Apabila piksel tersebut merupakan *noise* maka nilai baru akan dihitung dan di-set pada tahapan *noise reduction*. Terakhir, fase *image enhancement* akan dilakukan untuk menghasilkan citra digital dengan kualitas yang lebih bagus.

Perangkat lunak ini akan menyisipkan *noise* ke dalam citra input. Setelah itu, *noise* tersebut akan dihapus dengan menggunakan metode *adaptive noise reduction*. Terakhir, akan dilakukan pengujian terhadap citra hasil reduksi *noise* dengan menggunakan metode MSE dan PSNR.

Kata kunci: pengolahan citra digital, reduksi *noise*, *spatial median filter*, *adaptive noise reduction*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Citra sering memiliki resolusi yang buruk ataupun mengalami penurunan mutu atau kualitas yang diakibatkan gambar yang kurang tajam, kabur, munculnya derau atau *noise*. (Munir, 2008). Berdasarkan bentuk dan karakteristiknya *noise* pada citra dibedakan menjadi beberapa macam yaitu: *Gaussian noise*, *Speckle noise* dan *Salt and Pepper noise*. Model *Gaussian noise* sering disebut juga sebagai *noise* elektronik karena *noise* ini muncul pada *amplifier* atau *detector*. (Boyat, A.K. dan Brijendra Kumar Joshi, 2015) Efek dari *gaussian noise* ini, pada gambar muncul titik-titik berwarna yang jumlahnya sama dengan persentase *noise*. *Noise speckle* merupakan model *noise* yang memberikan warna hitam pada titik yang terkena *noise*. Sedangkan *noise salt* dan *pepper* seperti halnya taburan garam, akan memberikan warna putih pada titik yang terkena *noise*. Dalam tugas akhir ini *noise* yang digunakan adalah *impulse noise* (*salt and pepper*). *Impulse noise* biasanya terjadi selama transisi citra. *Noise* ini tampak sebagai impuls-impuls hitam atau putih diatas citra. *Impulse noise* ini dapat terjadi karena *error bit* acak pada saluran komunikasi. Citra yang mengalami gangguan (*noise*) perlu diperbaiki agar dapat meningkatkan kualitas citra. (Gebreyohannes, T. and D.Y. Kim, 2012)

Metode yang dapat digunakan untuk menghilangkan *salt and pepper noise* adalah algoritma *Fuzzy based Median Filtering*, namun kekurangan dari metode

tersebut adalah reduksi hanya pada gambar *grayscale*, persentase PSNR masih rendah dan kecepatan dalam penyelesaian lama. (Deshpande, B., H. K. Verma dan P. Deshpande) Metode lainnya yang dapat digunakan untuk menghilangkan *salt and pepper noise* adalah *spatial median filter* dan *adaptive noise reduction*. *Spatial Median Filter* adalah sebuah *noise removal filter* yang menerapkan sebuah algoritma *smoothing* yang teratur dengan tujuan untuk menghilangkan *noise* pada data citra dengan tetap menjaga sisi pada objek di citra. Sasaran dari sebuah *noise removal filter* adalah untuk mengambil sebuah citra *corrupted* sebagai *input* dan menghasilkan sebuah estimasi dari citra asli dengan tidak mengurangi kualitas dari citra asli. Sementara itu, *adaptive noise reduction* yang dipublikasikan oleh Tina Gebreyohannes dan Dong-Yoon Kim pada tahun 2012 ini fokus pada proses pendeteksian *salt and pepper noise* secara efektif dan mengembalikan citra digital secara efisien. Algoritma ini menggunakan tiga tahapan atau mekanisme yaitu deteksi piksel *noise* pada citra berdasarkan pada nilai *threshold* yang diberikan. Apabila piksel tersebut merupakan *noise* maka nilai baru akan dihitung dan di-set pada tahapan *noise reduction*. Terakhir, fase *image enhancement* akan dilakukan untuk menghasilkan citra digital dengan kualitas yang lebih bagus. (Gebreyohannes, T. and D.Y. Kim, 2012)

Metode *adaptive noise reduction* merupakan sebuah metode reduksi *salt and pepper noise* baru yang dapat melakukan pendeteksian dan reduksi *noise* secara efektif dan efisien, sehingga cocok untuk dipelajari (Gebreyohannes, T. and D.Y. Kim, 2012). Sementara itu, metode *Spatial Median Filter* merupakan metode reduksi *noise* yang mampu untuk menjaga sisi pada objek di citra

sehingga tidak terjadi perubahan bentuk pada objek citra. Kedua metode memiliki kelebihan dan kelemahan masing-masing. Untuk mengetahui kinerja dan performansi kedua metode dalam melakukan proses reduksi *noise* pada citra, maka perlu dirancang sebuah aplikasi yang mampu untuk membandingkan proses kerja dari kedua metode. Oleh karena itu, penulis tertarik untuk mengangkat masalah ini sebagai topik dengan judul "**Analisis Perbandingan Metode *Spatial Median Filter* dan *Adaptive Noise Reduction* untuk Menghilangkan *Salt and Pepper Noise Removal* pada Citra Digital**"

1.2. Rumusan Masalah

Dari uraian diatas, adapun perumusan masalah dalam tugas akhir ini dapat dirumuskan sebagai berikut :

1. Perlu sebuah aplikasi reduksi *noise* yang mampu untuk mereduksi *salt and pepper noise* terhadap citra *input* dengan menggunakan metode *Spatial Median Filter* dan *Adaptive Noise Reduction*.
2. Perlu dilakukan pengujian terhadap metode *Spatial Median Filter* dan *Adaptive Noise Reduction* dalam mereduksi *salt and pepper noise* agar dapat diketahui kinerja dari kedua metode tersebut.

1.3. Batasan Masalah

Batasan masalah dalam tugas akhir ini mencakup :

1. *Input* program adalah citra berwarna dengan format JPEG, BMP dan PNG.

2. Pengukuran kemiripan dua buah citra hasil reduksi *noise* dari citra asli menggunakan rumusan *Mean Square Error (MSE)* dan *Peak to Signal Noise Ratio (PSNR).*
3. *File* gambar yang akan dilakukan perbaikan kualitas adalah citra persegi dengan ukuran minimal 100 x 100 piksel dan maksimal 1024 x 1024 piksel.
4. *Salt and pepper noise* dapat dimasukkan sendiri ke dalam sebuah citra *input* melalui fitur yang disediakan oleh program.

1.4. Tujuan dan Manfaat Penelitian

A. Tujuan Penelitian

Tujuan penyusunan tugas akhir ini, yaitu:

1. Membuat sebuah aplikasi *noise removal* dengan menggunakan algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction*.
2. Membandingkan algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction* dalam mereduksi *salt and pepper noise*.

B. Manfaat Penelitian

Manfaat dari penyusunan tugas akhir ini, yaitu:

1. Membantu pemahaman terhadap cara kerja dari algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction* dalam mereduksi *salt and pepper noise*.
2. Mengetahui kinerja dari algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction* dalam mereduksi *salt and pepper noise* berdasarkan pada MSE dan PSNR yang dihasilkan.

3. Aplikasi dapat digunakan untuk melakukan proses reduksi citra *noise* sehingga dapat meningkatkan kualitas citra.

BAB II

LANDASAN TEORI

2.1 Pengenalan Citra

Citra didefinisikan sebagai fungsi dua dimensi, *f* (*x* ,*y*), dimana *x* dan *y* adalah koordinat spasial, dan lebar ayunan dari fungsi*f* pada pasangan koordinat (*x*, *y*) disebut sebagai *intensitas* atau *tingkat keabuan* dari gambar pada titik tersebut.Sistem perekaman data akan memberikan keluaran citra yang dapat bersifat optikal dalam bentuk foto, bersifat *analog* dalam bentuk sinyal-sinyal video, atau bersifat *digital* yang dapat langsung disimpan pada suatu *media* penyimpanan (Gonzales & Woods, 2002).

Menurut Kusumanto dan Alan Novi Tompunu (2011), secara matematis, citra merupakan fungsi kontinyu (*continue*) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Repersentasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi citra. Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi f(x,y) yang terdiri dari M kolom dan N baris, dimana perpotongan antara kolom dan baris disebut piksel (*pixel* = *picture element*) atau elemen terkecil dari sebuah citra.

2.1.1 Pembagian Citra

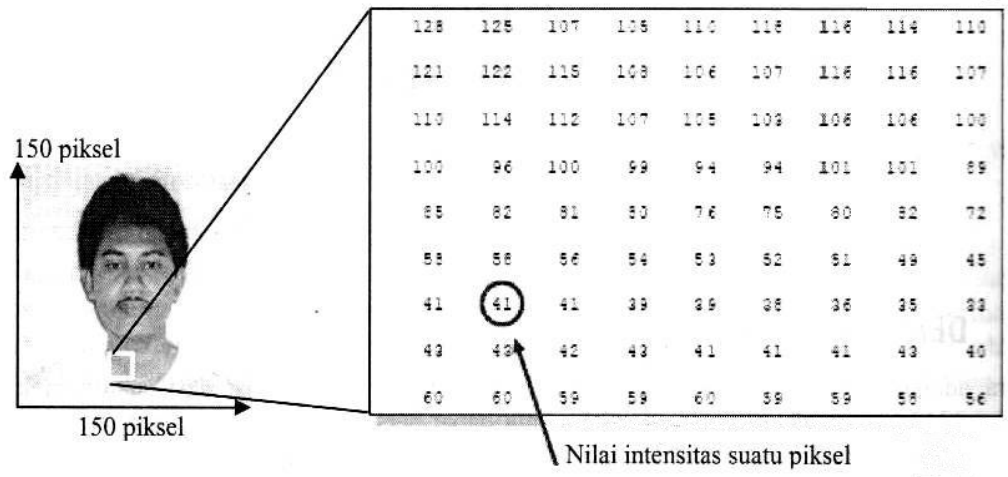
Menurut (Sutoyo, et al., 2009) citra dapat terbagi 2 yaitu citra yang bersifat analog dan citra yang bersifat digital.

1. Citra Analog

Citra analog adalah citra yang bersifat kontinu. Contoh citra analog adalah gambar pada monitor televisi, foto sinar X, lukisan, pemadangan alam, hasil *CT scan*, gambar-gambar yang terekam dalam pita kaset, dan lain sebagainya. Citra analog tidak bisa direpresentasikan dalam komputer sehingga tidak dapat diproses pada komputer secara langsung. Agar dapat diproses di komputer, citra harus terlebih dahulu dikonversi dari analog ke digital. Citra analog dihasilkan dari alat-alat analog, seperti video kamera analog, kamera foto analog, *WebCam*, *CT Scan*, sensor rontgen untuk foto thorax, sensor gelombang pendek pada sistem radar, sensor *ultrasound*, pada sistem USG, dan lain-lain(Sutoyo, et al., 2009).

2. Citra Digital

Citra digital adalah citra yang dapat diolah secara langsung oleh komputer (Sutoyo, T. et al, 2009:9). Jika sebuah citra *grayscale* dengan ukuran 150x150 piksel (elemen terkecil dari sebuah citra) diambil ukuran 9x9 piksel. Maka, bentuk potongan tersebut di monitor adalah kotak kecil seperti terlihat pada gambar 2.1. Nilai dari kotak kecil tersebut disimpan dimemori komputer dalam bentuk angka-angka yang menunjukkan besar intensitas dari setiap piksel pada gambar.



Gambar 2.1 Citra *grayscale* dengan ukuran 150x150 piksel

(Sumber : Sutoyo, T. et al, 2009:10)

Nilai pada suatu irisan antara baris dan kolom (pada posisi x,y) disebut dengan *picture element, image elements, pels* atau pixels. Istilah *pixel* paling sering digunakan pada citra digital (Putra, 2010).

Secara matematis, citra merupakan fungsi kontinyu (*continue*) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Representasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi citra (RD. Kusumanto dan Alan Novi Tompunu, 2011).

2.1.2 Elemen – Elemen Citra Digital

Citra digital mengandung sejumlah elemen-elemen dasar yang dimanipulasi dalam pengolahan citra. Elemen-elemen dasar yang penting diantaranya adalah (Munir, 2004):

1. Kecerahan (*brightness*)

   Kecerahan adalah kata lain untuk intensitas cahaya. Kecerahan pada sebuah titik (*pixel*) di dalam citra bukanlah intensitas yang rill, tetapi sebenarnya adalah intensitas rata-rata dari suatu area yang melingkupinya. Sistem visual manusia mampu menyesuaikan dirinya dengan tingkat kecerahan (*brightness level*) mulai dari yang paling rendah sampai dengan yang paling tinggi dengan jangkauan sebesar 10^{10}.

2. Kontras (*contrast*)

   Kontras menyatakan sebaran terang (*lightness*) dan gelap (*darkness*) di dalam sebuah gambar. Citra dengan kontras rendah dicirikan oleh sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Pada citra dengan kontras yang baik, komposisi gelap dan terang tersebar secara merata.

3. Kontur (*contour*)

   Kontur adalah keadaan yang ditimbulkan oleh perubahan intensitas pada *pixel-pixel* yang bertetangga. Karena adanya perubahan intensitas inilah maka kita mampu mendeteksi tepi-tepi (edge) objek di dalam citra.

4. Warna (color)

   Warna-warna yang diterima oleh mata (system visual manusia) merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Penelitian memperlihatkan bahwa kombinasi warna yang memberikan rentang warna yang paling lebar adalah *red* (R), *green* (G), dan *blue* (B).

Persepsi sistem visual manusia terhadap warna sangat relative sebab dipengaruhi oleh banyak kriteria, salah satunya disebabkan oleh adaptasi yang menimbulkan distorsi. Misalnya bercak abu-abu di sekitar warna hijau akan tampak keungu-unguan (distorsi terhadap ruang), atau jika mata melihat warna hijau lalu langsung dengan cepat melihat warna abu-abu, maka mata menangkap kesan warna abu-abu tersebut sebagai warna ungu (distorsi terhadap waktu).

5. Bentuk (*shape*)

   Shape adalah property intrinsic dari objek tiga dimensi, dengan pengertian bahwa *shape* merupakan property intrinsic utama untuk system visual manusia. Manusia lebih sering mengasosiasikan objek denan bentuknya ketimbang elemen lainnya (warna misalnya). Pada umumnya, citra yang dibentuk oleh mata merupakan citra dwimatra (2 dimensi), sedangkan objek yang dilihat umumnya berbentuk trimatra (3 dimensi). Informasi bentuk objek dapat diekstraksi dari citra pada pemulaan pra-pengolahan dan segmentasi citra.

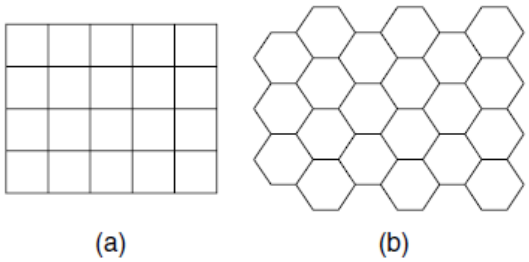
6. Tekstur (*texture*)

   Tekstur dicirikan sebagai distribusi spasial dari derajat keabuan di dalam sekumpulan *pixel-pixel* yang bertetangga. Sistem visual manusia pada hakikatnya tidak menerima informasi citra secara independen pada setiap *pixel*, melainkan suatu citra dianggap sebagai suatu kesatuan. Resolusi citra yang diamati ditentukan oleh skala pada mana tekstur tersebut dipersepsi.

2.1.3 Representasi Citra

Sebuah gambar dimodelkan dengan fungsi kontinu dari dua variabel *f* (*x*, *y*) dimana variabel *x* dan *y* merupakan koordinat pada bidang gambar (Cristobal G. et al). Sebelum gambar direpresentasikan, fungsi kontinu terlebih dahulu diubah menjadi fungsi digital dengan menerapkan *discretization* dan *quantization* pada gambar. Dalam tahap *discretization*, koordinat *x*, *y* dari gambar di-*sampling* pada *domain spatial* dalam sebuah *grid* terpilih. Pada dasarnya, lebih dekat jarak sampel, lebih baik juga perkiraan dari fungsi kontinu pada gambar asli. Namun, *sampling* yang padat membutuhkan lebih banyak memori untuk menyimpan gambar. Sebagai penyelesaian, *sampling* akan menggunakan teorema *sampling Shannon* untuk menghasilkan frekuensi *sampling* yang tepat.

Sampling pada umumnya menggunakan dua macam *grid* yaitu, *square grid* dan *hexagonal grid*. *Hexagonal grid* memiliki prioritas perhitungan yang lebih jelas daripada *square grid*. Keuntungan dari penggunaan *hexagonal grid* adalah sebuah piksel memiliki *Euclidean distance* yang sama terhadap seluruh tetangganya. Keuntungan penggunaan *rectangular* (*square*) *grid*, yaitu untuk menggambarkan sebuah hubungan gambar dalam matriks. Koordinat baris, kolom sama dengan koordinat spasial *x*, *y* pada gambar dan nilai dari elemen matriks memberikan nilai dari fungsi gambar, berupa tingkat kecerahan. Setiap elemen dari sebuah matriks mewakili piksel, yang merupakan elemen dari gambar (*picture element*). Dari keuntungan tersebut, hampir seluruh *sampling grid* citra menggunakan *square grid*.



Gambar 2.2 Konfigurasi *sampling grid*: (a) *square* dan (b) *hexagonal*

(Sumber: Cristobal G. et al)

Setelah *discretization*, kembalian yang dihasilkan oleh fungsi *sampling* berupa intensitas jarak yang berkelanjutan harus diubah (*quantized*) menjadi satuan diskrit sehingga dapat digunakan untuk membentuk fungsi digital (Gonzales, R. C. dan Woods, R. E., 2007). Sebagai contoh representasi citra digital, akan dilakukan proses *sampling* terhadap gambar yang bersifat kontinu ke sebuah *array* 2D, *f*(*x*, *y*), dengan baris sebanyak M dan kolom sebanyak N, dimana (*x*, *y*) adalah koordinat diskrit. Koordinat diskrit *x* diisi dengan nilai bilangan bulatyakni 0, 1, 2, ..., M-1, dan *y* diisi dengan nilai 0, 1, 2, …, N-1, atau jika dituliskan:

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & \vdots & & \vdots \\ f(M-1,0) & f(M-1,1) & \cdots & f(M-1,N-1) \end{bmatrix} \quad (2\text{-}3)$$

Dimana setiap elemen dalam matriks ini disebut sebagai *image element*, *picture element*, *pixel* atau *pel*.

Dalam proses digitalisasi (*sampling* dan *quantization*), nilai dari M, N, dan L sebagai intensitas dari koordinat diskrit harus ditentukan, dan biasanya nilai L merupakan perpangkatan dua atau dapat dituliskan sebagai berikut:

$$L = 2^k \quad \text{.....} \quad (2\text{-}4)$$

Sehingga, jika *b* merupakan bit yang dibutuhkan untuk menyimpan sebuah citra digital, maka *b* dapat dihitung dengan persamaan berikut:

$$b = M \times N \times k \quad \text{.....} \quad (2\text{-}5)$$

Meskipun sebuah citra kaya informasi, namun seringkali citra yang kita miliki mengalami penurunan mutu (degradasi), misalnya mengandung cacat atau derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang.

2.1.4 Jenis Citra

Untuk menyimpan foto dan citra digunakan format citra layar kuadratis (berbentuk kotak) yang terdiri atas titik-titik citra kecil yang disebut dengan piksel (*pixel*). *Pixel* (*picture element*) adalah sebuah titik yang merupakan elemen paling kecil pada citra satelit. Angka numerik (1 byte) dari pixel disebut *digital number* (DN). DN bisa ditampilkan dalam warna kelabu, berkisar antara putih dan hitam (*grayscale*), tergantung level energi yang terdeteksi. *Pixel* yang disusun dalam order yang benar akan membentuk sebuah citra. Kebanyakan citra satelit yang belum diproses disimpan dalam bentuk gray scale, yang merupakan skala warna dari hitam ke putih dengan derajat keabuan yang bervariasi. Untuk PJ, skala yang dipakai adalah 256 shade gray scale, dimana nilai 0 menggambarkan hitam, nilai 255 putih. Dua gambar di bawah ini menunjukkan derajat keabuan dan hubungan antara DN dan derajat keabuan yang menyusun sebuah citra. Untuk citra

multispectral, masing masing pixel mempunyai beberapa DN, sesuai dengan jumlah band yang dimiliki (Harvei Desmon Hutahaean, 2013).

Resolusi dari sebuah citra adalah karakteristik yang menunjukkan level kedetailan yang dimiliki oleh sebuah citra. Resolusi didefinisikan sebagai area dari permukaan bumi yang diwakili oleh sebuah *pixel* sebagai elemen terkecil dari sebuah citra. Pada citra satelit pemantau cuaca yang mempunyai resolusi 1 km, masing-masing *pixel* mewakili rata-rata nilai *brightness* dari sebuah area berukuran 1×1 km. Bentuk yang lebih kecil dari 1 km susah dikenali melalui image dengan resolusi 1 km. Landsat 7 menghasilkan citra dengan resolusi 30 meter, sehingga jauh lebih banyak detail yang bisa dilihat dibandingkan pada citra satelit dengan resolusi 1 km. Resolusi adalah hal penting yang perlu dipertimbangkan dalam rangka pemilihan citra yang akan digunakan terutama dalam hal aplikasi, waktu, biaya, ketersediaan citra dan fasilitas komputasi (Harvei Desmon Hutahaean, 2013).

Warna adalah persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Setiap warna mempunyai panjang gelombang yang berbeda. Warna merah mempunyai panjang gelombang paling tinggi, sedangkan warna ungu mempunyai panjang gelombang paling rendah. Warna-warna yang diterima oleh mata merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Kombinasi warna yang memberikan rentang warna yang paling lebar adalah *red*(R), *green*(G) dan *blue*(B) dan warna bukan merupakan besaran fisik tetapi warna merupakan suatu sensasi yang dihubungkan dengan sistem saraf kita, seperti halnya rasa maupun bau. Sensasi

warna diperoleh dengan adanya interaksi antara warna dengan sistem saraf sensitive warna kita (Harvei Desmon Hutahaean, 2013).

Nilai suatu pixel memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung dari jenis warnanya. Namum secara umum jangkauannya adalah 0 -255. Citra dengan penggambaran seperti ini digolongkan ke dalam citra integer. Berikut adalah jenis-jenis citra berdasarkan nilai pixelnya(Putra, 2010).

1. Citra Biner

Citra biner adalah citra digital yang hanya memiliki dua kemungkinan nilai piksel yaitu hitam dan putih. Citra biner juga disebut sebagai citra B&W (*black and white*) atau citra monokrom. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap piksel dari citra biner (Putra, 2010).

Citra biner sering kali muncul sebagai hasil dari proses pengolahan seperti segmentasi, pengambangan, morfologi, ataupun dithering (Putra, 2010).

Gambar 2.5 Citra Biner (Sumber: Putra, 2010)

2. Citra Grayscale

Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, dengan kata lain nilai bagian RED = GREEN = BLUE. Nilai tersebut digunakan untuk menunjukkan tinggi intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan, dan putih. Tingkatan keabuan di sini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih. Citra *grayscale* berikut memiliki kedalaman warna 8 bit (256 kombinasi warna keabuan) (Putra, 2010).

Gambar 2.6 Citra Grayscale (Sumber: Putra, 2010)

3. Citra Warna (8 bit)

Setiap piksel dari citra warna (8 bit) hanya diwakili oleh 8 bit dengan jumlah warna maksimum yang dapat digunakan adalah 256 warna. Ada dua jenis citra warna 8 bit, yaitu yang pertama citra warna 8 bit dengan menggunakan palet warna 256 dengan setiap paletnya memiliki pemetaan nilai (*colormap*) RGB tertentu. Model ini lebih sering digunakan dan kedua setiap piksel memiliki format 8 bit sebagai berikut (Putra, 2010).

Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	G	G	G	B	B

Gambar 2.7 Citra warna 8 bit dengan palet (Sumber: Putra, 2010)

4. Citra Warna (16 bit)

Citra warna 16 bit biasanya disebut sebagai citra *highcolor* dengan setiap pikselnya diwakili dengan 2 byte memory (16 bit).

Warna 16 bit memiliki 65.536 warna. Dalam formasi bitnya, nilai merah dan biru mengambil tempat di 5 bit di kanan dan kiri. Komponen hijau memiliki 5 bit ditambah 1 bit ekstra. Pemilihan komponen hijau dengan deret 6 bit dikarenakan penglihatan manusia lebih sensitive terhadap warna hijau (Putra, 2010).

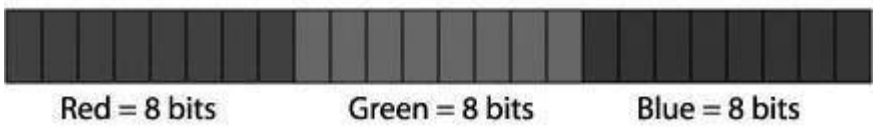
Bit-15	Bit-14	Bit-13	Bit-12	Bit-11	Bit-10	Bit-9	Bit-8	Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0
R	R	R	R	R	G	G	G	G	G	G	B	B	B	B	B

Gambar 2.8 Contoh citra warna 16 bit (Sumber: Putra, 2010)

5. Citra Warna (24 bit)

Setiap piksel dari citra warna 24 bit diwakili dengan 24 bit sehingga total 16.777.216 variasai warna. Variasi ini sudah lebih dari cukup untuk memvisualisasikan seluruh warna yang dapat dilihat oleh penglihatan manusia. Penglihatan manusia dipercaya hanya dapat membedakan hingga 10 juta warna saja (Putra, 2010).

Setiap poin informasi *pixel* (RGB) disimpan ke dalam 1 *byte* data. 8 bit pertama menyimpan nilai biru, kemudian diikuti dengan nilai hijau pada 8 bit kedua dan 8 bit terakhir merupakan warna merah (Putra, 2010).



Gambar 2.9Contoh citra warna 24 bit (Sumber: Putra, 2010)

2.1.5 Format File Citra

Sebuah format file citra harus dapat menyatukan kualitas citra, ukuran file dan kompabilitas dengan berbagai aplikasi. Format file citra standar yang digunakan saat ini terdiri dari beberapa jenis. Format- format ini digunakan untuk menyimpan citra dalam sebuah file. Setiap format memiliki karakteristik masing-masing. Ini adalah contoh format umum, yaitu : Bitmap (.bmp), Graphics Interchange Format (.gif), Portable Network Graphics (.png), JPEG (.jpg) (Putra, 2010).

Bahkan menurut (Sutoyo, et al., 2009), ada dua jenis format file citra yang sering digunakan dalam pengolahan citra, yaitu citra bitmap dan citra vektor. Pada citra bitmap ini sering disebut juga citra raster. Citra bitmap ini menyimpan data kode citra secara digital dan lengkap (cara penyimpanannya adalah per piksel). Citra bitmap ini dipresentasikan dalam bentuk matriks atau dipetakan dengan menggunakan bilangan biner atau sistem bilangan yang lain. Citra ini memiliki

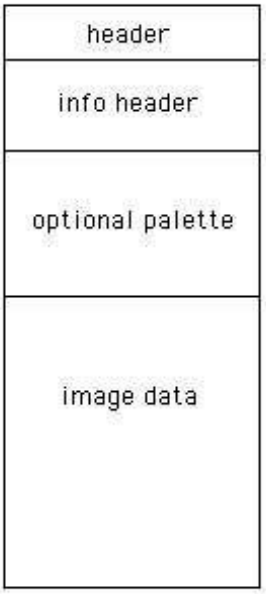
kelebihan untuk memanipulasi warna, tetapi untuk mengubah objek lebih sulit. Tampilan bitmap mampu menunjukkan kehalusan gradasi bayangan dan warna dari sebuah gambar. Tetapi bila tampilan diperbesar maka tampilan di monitor akan tampak pecah-pecah (kualitas citra menurun). Contoh format file citra antara lain adalah BMP, GIFF, TIF, WPG, IMG, dll. Sedangkan pada format file citra vektor merupakan citra vektor yang dihasilkan dari perhitungan matematis dan tidak terdapat piksel, yaitu data yang tersimpan dalam bentuk vektor posisi, dimana yang tersimpan hanya informasi vektor posisi dengan bentuk sebuah fungsi. Pada citra vektor, mengubah warna lebih sulit dilakukan, tetapi membentuk objek dengan cara mengubah nilai lebih mudah. Oleh karena itu, bila citra diperbesar atau diperkecil, kualitas citra relatif tetap baik dan tidak berubah. Citra vektor biasanya dibuat menggunakan aplikasi- aplikasi citra vektor seperti CorelDRAW, Adobe Illustrator, Macromedia Freehand, Autocad.

1. Bitmap (.bmp)

Format .bmp adalah format penyimpanan standar tanpa kompresi yang umum dapat digunakan untuk menyimpan citra biner hingga citra warna. Format ini terdiri dari beberapa jenis yang setiap jenisnya ditentukan dengan jumlah bit yang digunakan untuk menyimpan sebuah nilai pixel (Putra, 2010).

File BMP adalah format file bersejarah (tapi masih umum digunakan) untuk sistem operasi yang disebut "Windows". Gambar BMP dapat berkisar dari hitam dan putih (1 bit per pixel) hingga 24 bit warna (16,7 juta warna). Sementara gambar dapat dikompresi (Bourke, 1998).

Sebuah file BMP terdiri dari 3 atau 4 bagian seperti yang ditunjukkan dalam diagram di bawah. Bagian pertama adalah header, ini diikuti oleh bagian informasi, jika gambar diindeks warna kemudian palet berikut, dan terakhir dari semua adalah data pixel. Posisi data gambar sehubungan dengan sart file yang terkandung dalam header. Informasi seperti lebar gambar dan tinggi, jenis kompresi, jumlah warna yang terkandung dalam header informasi (Bourke, 1998).



Gambar 2.10 Diagram format file BMP (Sumber: Bourke, 1998)

File header

Header terdiri dari bidang-bidang berikut. Perhatikan bahwa kita mengasumsikan int pendek 2 byte, int dari 4 byte, dan panjang int dari 8 byte. Selanjutnya kita mengasumsikan byte pemesanan seperti untuk khas (Intel) mesin. Header adalah 14 byte panjangnya (Bourke, 1998).

typedef struct {

```
unsigned short int type;
unsigned int size;
unsigned short int reserved1, reserved2;
unsigned int offset;
} HEADER;
```

Information

Info gambar data yang berikut adalah 40 byte panjangnya, hal ini dijelaskan dalam struct diberikan di bawah. Bidang yang paling menarik di bawah ini adalah lebar gambar dan tinggi, jumlah bit per pixel (harus 1, 4, 8 atau 24), jumlah pesawat (diasumsikan 1 di sini), dan jenis kompresi (diasumsikan menjadi 0 di sini) (Bourke, 1998).

```
typedef struct {
unsigned int size;
int width,height;
unsigned short int planes;
unsigned short int bits;
unsigned int compression;
unsigned int imagesize;
int xresolution,yresolution;
unsigned int ncolours;
unsigned int importantcolours;
} INFOHEADER;
```

24 bit Image Data

Data yang paling sederhana untuk membaca adalah 24 bit gambar true color. Dalam hal ini data gambar berikut segera setelah header informasi, yaitu, tidak ada palet warna. Ini terdiri dari tiga byte per pixel di b, g, agar r. Setiap byte memberikan saturasi untuk itu komponen warna, 0 untuk hitam dan 1 untuk putih (sepenuhnya jenuh) (Bourke, 1998).

Indexed Colour Data

Jika gambar diindeks warna kemudian segera setelah header informasi akan ada tabel infoheader.ncolours warna, masing-masing 4 byte. Tiga byte pertama sesuai dengan b, g, komponen r, byte terakhir dicadangkan / tidak terpakai tapi jelas bisa mewakili channel alpha. Untuk 8 bit gambar greyscale indeks warna ini umumnya hanya menjadi jalan abu-abu. Jika Anda melakukan penjumlahan, maka panjang header ditambah panjang blok informasi ditambah 4 kali jumlah warna palet harus sama dengan data gambar offset (Bourke, 1998).

2. Portable Network Graphics (.png)

Format .png adalah format penyimpanan citra terkompresi. Format ini dapat digunakan pada citra *grayscale,* citra dengan palet warna, dan juga citra *fullcolor.* Format .png juga mampu menyimpan informasi hinggal kanal alpha dengan penyimpanan sebesar 1 hingga 16 bit per kanal (Putra, 2010).

PNG singkatan dari Portable Network Graphics. PNG adalah format citra bitmap yang menggunakan kompresi data lossless. PNG diciptakan untuk

meningkatkan dan menggantikan format GIF. Format file PNG dianggap, dan dibuat sebagai penerus gratis dan open source untuk format file GIF. Format file PNG mendukung true color (16 juta warna), sedangkan format file GIF hanya memungkinkan 256 warna. PNG unggul ketika citra memiliki area besar warna seragam. The lossless PNG paling cocok untuk mengedit citra, sedangkan format lossy seperti JPG yang terbaik untuk distribusi final fotografi-jenis citra karena ukuran file yang lebih kecil. Namun banyak browser sebelumnya tidak mendukung format file PNG; Namun dengan rilis Internet Explorer 7 browser modern semua populer sepenuhnya mendukung PNG. Fitur khusus dari file PNG termasuk dukungan untuk hingga 48 bit informasi warna. Format PNG memanfaatkan jalinan skema 2D, yang semakin menampilkan citra lebih cepat daripada sebuah file citra GIF. Gamma koreksi memungkinkan nilai-nilai yang ditampilkan pada platform apapun menjadi sama dengan yang asli (Jayaraman, et al., 2009).

Fitur penting dari gambar PNG adalah sebagai berikut (Jayaraman, et al., 2009):

(i) citra PNG menggunakan skema kompresi lossless.

(ii) citra PNG yang interlaced.

(iii) citra PNG mendukung 8-bit transparansi.

3. JPEG (.jpg)

Format .jpg adalah format yang sangat umum digunakan saat ini khususnya untuk transmisi citra. Format ini digunakan untuk menyimpan citra hasil kompresi dengan metode JPEG (Putra, 2010).

JPEG sebenarnya bukan jenis file. JPEG adalah standar saat yang paling penting untuk kompresi citra. Standar JPEG diciptakan oleh kelompok kerja Organisasi Internasional untuk Standardisasi (ISO). Format ini menyediakan pilihan kompresi paling dramatis untuk citra fotografi. Kompresi JPEG digunakan dalam format file JFIF yang menggunakan ekstensi file (.jpg). Format ini berguna ketika ruang penyimpanan adalah pada premium. JPEG menyimpan citra raster tunggal dalam warna 24-bit. JPEG adalah format platform-independen yang mendukung tingkat tertinggi kompresi; Namun, kompresi ini lossy. File JPEG progresif mendukung interlacing (Jayaraman, et al., 2009).

Fitur penting dari format file JPEG adalah sebagai berikut (Jayaraman, et al., 2009):

1. JPEG menggunakan skema kompresi lossy.
2. citra JPEG tidak interlaced; Namun, citra JPEG progresif dapat interlaced.

File Format Kekuatan format file JPEG adalah kemampuannya untuk kompres file citra yang lebih besar. Karena kompresi ini, data citra dapat disimpan secara efektif dan efisien menular dari satu tempat ke tempat lain (Jayaraman, et al., 2009).

File Format JPEG dalam versi basis tidak mendukung beberapa lapisan, kisaran dinamis tinggi. Oleh karena itu JPEG tidak akan menjadi pilihan bijak jika

ada yang tertarik untuk mempertahankan citra berkualitas tinggi (Jayaraman, et al., 2009).

2.2 Pengolahan Citra

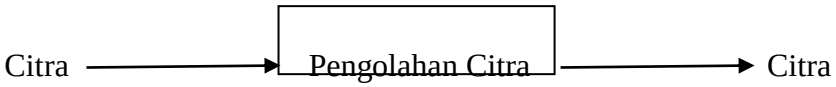
Pengolahan citra digital adalah sebuah disiplin ilmu yang mempelajari hal-hal yang berkaitan dengan perbaikan kualitas gambar (peningkatan kontras, transformasi warna, restorasi citra), transformasi gambar (rotasi, translasi, skala, transformasi geometrik), melakukan pemilihan citra ciri (*feature images)* yang optimal untuk tujuan analisis, melakukan proses penarikan informasi atau deskripsi objek atau pengenalan objek yang terkandung pada citra, melakukan kompresi atau reduksi data untuk tujuan penyimpanan data, transmisi data, dan waktu proses data. Input dari pengoalahan citra adalah citra, sedangkan outputnya adalah citra hasil pengolahan (Sutoyo, et al., 2009)

Pengolahan citra pada dasarnya mencakup tiga langkah berikut (Munir, 2004):

a. Mengimpor gambar melalui alat akuisisi gambar.
b. Menganalisis dan memanipulasi gambar.
c. Output yang hasilnya dapat diubah gambar atau laporan yang didasarkan pada analisis citra.

Pengolahan Citrabertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas

lebih baik daripada citra masukan. Termasuk ke dalam bidang ini juga adalah pemampatan citra (*image compression*)(Munir, 2004).



Gambar 2.11 Skema Pengolahan Citra

(Sumber: Munir, 2008)

Perbaikan citra bertujuan meningkatkan kualitas tampilan citra untuk pandangan manusia atau untuk mengkonversi suatu citra agar memiliki format yang lebih baik sehingga citra tersebut menjadi lebih mudah diolah dengan mesin (komputer). Perbaikan terhadap suatu citra dapat dilakukan dengan operasi titik (*point operation*), operasi spasial (*spatial operation*), operasi geometri (*geometric operation*), dan operasi aritmatik (*arithmetic operation*) (Putra, 2010).

Perubahan bentuk suatu citra dapat berupa perubahan geometri pixel seperti perputaran (rotasi), pergeseran (translasi), penskalaan, dan lain sebagainya atau dapat juga berupa perubahan ruang (domain) citra ke domain lainnya, sepertin transformasi Fourier yang mengubah suatu citra dari domain spasial menjadi domain frekuensi(Putra, 2010).

Melalui proses transformasi, suatu citra dapat dinyatakan sebagai kombinasi linier dair sinyal dasar (*basis signals*) yang sering disebut dengan fungsi basis (basis *function*). Suatu citra yang telah mengalami transformasi dapat diperoleh kembali dengan menggunakan transformasi balik (*invers transformation*). Tujuan diterapkannya transformasi citra adalah untuk

memperoleh informasi (*feature extraction*) yang lebih jelas yang terkandung dalam suatu citra(Putra, 2010).

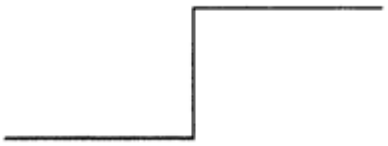
Analisis citra digunakan untuk menghitung besaran kuantitif dari citra untuk menghasilkan deskripsinya. Teknik analisis citra mengekstrasi ciri-ciri tertentu yang membantu dalam identifikasi objek, contoh operasi dalam analisis citra misalnya pendeteksian tepi objek (*edge detection*)(Munir, 2004).

Tepian citra dapat merepresentasikan objek-objek yang terkandung dalam citra tersebut, bentuk, dan ukurannya serta terkadang juga informasi tentang teksturnya. Tepian citra adalah posisi di mana intensitas pixel dari citra berubah dari nilai rendah ke nilai tinggi atau sebaliknya. Deteksi tepi umumnya adalah langkah awal melakukan segmentasi citra (Putra, 2010).

Tepian citra dapat dilihat melalui perubahan intensitas pixel pada suatu area. Berdasarkan perbedaan perubahaan intensitas tersebut, tepian dapat dibagi menjadi 4 jenis, antara lain adalah (Putra, 2010):

1. Step

   Tepian jenis step merupakan tepian citra yang terbentuk dari perubahan intensitas citra secara signifikan dari tinggi ke rendah ataupun sebaliknya.



Gambar 2.13 Tepian Step (Sumber: Putra, 2010)

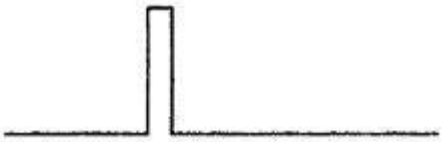
2. Ramp

   Tepian jenis ini terbentuk dari perubahan intensitas citra secara perlahan. Perubahan secara perlahan dapat dilihat pada bentuk kurva yang semakin tinggi dengan perubahan kontinu.

Gambar 2.14 Tepian Ramp (Sumber: Putra, 2010)

3. Line

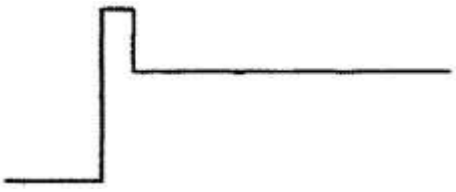
   Tepian jenis ini ditandai dengan perubahan intensitas secara drastis dari intensitas rendah-tinggi-rendah atau sebaliknya.



Gambar 2.15 Tepian Line (Sumber: Putra, 2010)

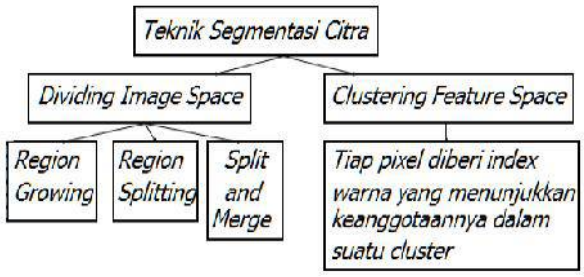
4. Step-line

   Tepian step-line merupakan gabungan dari tepian jenis step dan line. Tepian jenis ini ditandai dengan peningkatan intensitas yang tajam dalam interval tertentu dan kemudian ditandai dengan penurunan yang tidak signifikan, sehingga perubahan intensitas selanjutnya berlangsung stabil.



Gambar 2.16 Tepian Step-line (Sumber: Putra, 2010)

Segmentasi merupakan teknik untuk membagi suatu citra menjadi beberapa daerah (*region*) di mana setiap daerah memiliki kemiripan atribut. Menurut (Jain, 1989), segmentasi citra dapat dibagi dalam beberapa jenis yaitu dividing image space dan clustering feature space. Jenis yang pertama adalah teknik segmentasi dengan membagi image menjadi beberapa bagian untuk mengetahui batasannya, sedangkan teknik yang kedua dilakukan dengan cara memberi index warna pada tiap piksel yang menunjukkan keanggotaan dalam suatu segmentasi. Teknik segmentasi citra menurut (Jain, 1989) dapat dilihat pada diagram berikut:

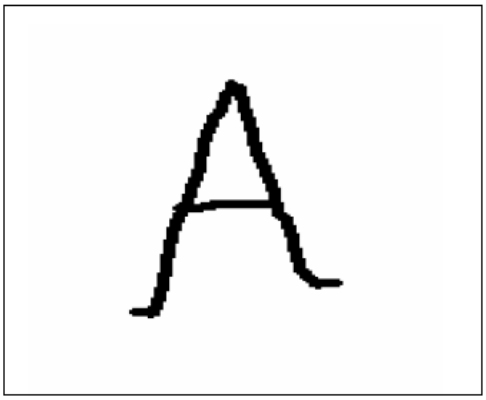


Gambar 2.17 Teknik segmentasi citra (Sumber: Jain, 1989)

Pengenalan Polamengelompokkan data numerik dan simbolik (termasuk citra) secara otomatis oleh mesin (dalam hal ini komputer). Tujuan pengelompokan adalah untuk mengenali suatu objek di dalam citra. Manusia bisa mengenali objek yang dilihatnya karena otak manusia telah belajar mengklasifikasi objek-objek di alam sehingga mampu membedakan suatu objek dengan objek lainnya. Kemampuan sistem visual manusia inilah yang dicoba ditiru oleh mesin. Komputer menerima masukan berupa citra objek yang akan

diidentifikasi, memproses citra tersebut, dan memberikan keluaran berupa deskripsi objek di dalam citra(Munir, 2004).

Contoh pengenalan pola misalnya citra pada Gambar 2.18adalah tulisan tangan yang digunakan sebagai data masukan untuk mengenali karakter '*A*'. Dengan menggunakan suatu algoritma pengenalan pola, diharapkan komputer dapat mengenali bahwa karakter tersebut adalah '*A*'.



Gambar 2.18Citra karakter 'A' yang digunakan sebagai masukan untuk pengenalan huruf. (Sumber: Munir, 2004).

Kompresi data citra merupakan proses mereduksi ukuran suatu data untuk menghasilkan representasi dgital yang padat atau mampat namun tetap dapat mewakili kuantitas informasi yang terkandung pada data tersebut. Pada citra kompresi mengarah pada minimisasi jumlah bit rate untuk representasi digital. Tujuan daripada kompresi data tiada lain adalah untuk mengurangi data berlebihan tersebut sehingga ukuran data menjadi lebih kecil dan lebih ringan dalam proses transmisi (Putra, 2010).

2.2.1 Perbaikan Kualitas Citra (*image enhancement*)

Umumnya, operasi-operasi pada pengolahan citra diterapkan pada citra bila (Munir, 2004):

a. perbaikan atau memodifikasi citra perlu dilakukan untuk meningkatkan kualitas penampakan atau untuk menonjolkan beberapa aspek informasi yang terkandung di dalam citra.
b. Elemen di dalam citra perlu dikelompokkan, dicocokkan, atau diukur,
c. Sebagian citra perlu digabung dengan bagian citra yang lain.

Perbaikan kualitas citra (image enhancement) merupakan salah satu proses awal dalam pengolahan citra (image preprocessing). Perbaikan kualitas diperlukan karena seringkali citra yang dijadikan objek pembahasan mempunyai kualitas yang buruk, misalnya citra mengalami derau (noise) pada saat pengiriman melalui saluran transmisi, citra terlalu terang/gelap, citra kurang tajam, kabur, dan sebagainya. Melalui operasi pemrosesan awal iniah kualitas citra diperbaiki sehingga citra dapat digunakan untuk aplikasi lebih lanjut, misalnya untuk aplikasi pengenalan (recognition) objek di dalam citra.

Proses-proses yang termasuk ke dalam perbaikan kualitas citra (Sutoyo, et al., 2009):

1. Pengubahan kecerahan gambar (*image brightness*)
2. Peregangan kontras (*contrast stretching*)
3. Pengubahan histogram citra – Perataan histogram
4. Pelembutan citra (*image smoothing*)
5. Penajaman tepi (*sharpening edge*)

1. Kecerahan Gambar (*image brightness*)

Untuk membuat citra lebih terang atau lebih gelap, perlu melakukan pengubahan kecerahan gambar. Kecerahan/kecemerlangan gambar dapat diperbaiki dengan menambahkan (atau mengurangkan) sebuah konstanta kepada (atau dari) setiap pixel di dalam citra. Akibat dari operasi ini, histogram citra mengalami pergeseran(Munir, 2004).

Secara matematis operasi ini ditulis sebagai

$$f(x,y)' = f(x,y) + b \quad \text{.....} \quad (2\text{-}6)$$

Jika b positif, kecerahan gambar bertambah, sebaliknya jika b negatif kecerahan gambar berkurang.

Nilai pixel hasil pengubahan mungkin £ derajat keabuan minimum (0) atau ³ derajat keabuan maksimum (255). Karena itu, pixel tersebut perlu dilakukan clipping ke nilai keabuan minimum atau ke nilai keabuan maksimum. Sebagai contoh, Gambar 2.19(a) adalah citra Zelda (beserta histogramnya) yang tampak gelap, sedangkan Gambar 2.19(b) adalah citra Zelda (beserta histogramnya) yang lebih terang (nilai b = 80). Perhatikan histogramnya. Sebelum operasi penambahan kecerahan, histogramnya menumpuk di bagian kiri. Setelah penambahan kecerahan, histogramnya bergeser ke bagian kanan (Munir, 2004).

(a) Citra Zelda (orisinil)

(b) Histogram citra Zelda (orisinil)

(c) Citra Zelda setelah penambahan kecerahan dengan $b = 80$

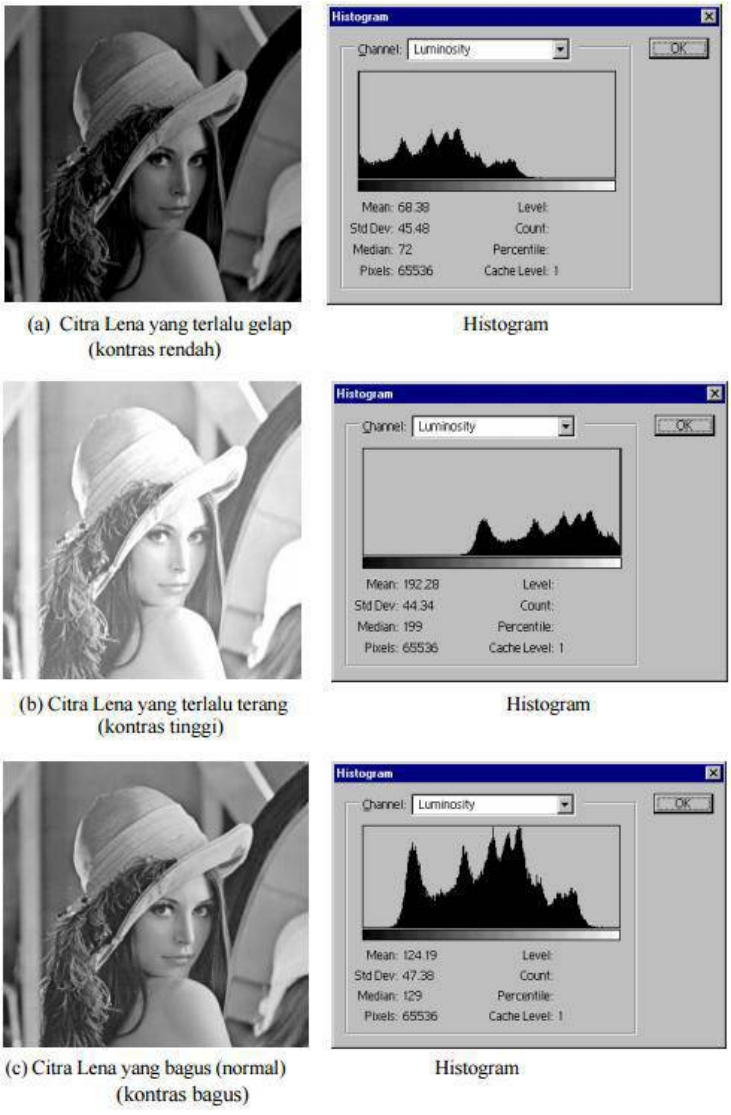
(d) Histogram citra Zelda setelah penambahan kecerahan

Gambar 2.19 Citra Zelda; (a,b): sebelum operasi penambahan kecerahan terlihat agak gelap; (c,d) citra Zelda setelah operasi penambahan kecerahan dengan b = 80(Sumber: Munir, 2004).

2. Peregangan kontras (*contrast stretching*)

Kontras menyatakan sebaran terang (lightness) dan gelap (darkness) di dalam sebuah gambar. Citra dapat dikelompokkan ke dalam tiga kategori kontras: citra kontras-rendah (low contrast), citra kontras-bagus (good contrast atau normal contrast), dan citra kontras-tinggi (high contrast). Ketiga kategori ini umumnya dibedakan secara intuitif (Munir, 2004).

Gambar 2.20 memperlihatkan tiga buah citra Lena yang masing-masing memiliki kontras-rendah, kontras-tinggi, dan kontras-bagus. Ketiga histogram ini dihasilkan dengan program Adobe Photoshop (Munir, 2004).



Gambar 2.20 Tiga buah citra lena dengan tiga macam kontras (Sumber: Munir, 2004).

2.3 *Impulse Noise (Salt and Pepper)*

Setiap gangguan pada citra dinamakan dengan *noise*. *Noise* pada citra tidak hanya terjadi karena ketidaksempurnaan dalam proses pengambilan gambar, tetapi bisa juga disebabkan oleh kotoran-kotoran yang terjadi pada citra. Berdasarkan bentuk dan karakteristiknya *noise* pada citra dibedakan menjadi beberapa macam yaitu: *Gaussian noise, Speckle noise dan Salt and Pepper noise*. Dalam tugas akhir ini *noise* yang digunakan adalah *impulse noise (salt and pepper*).*Impulsenoise* biasanya terjadi selama transisi citra. *Noise* ini tampak sebagai impuls-impuls hitam atau putih diatas citra. *Impulsenoise* ini dapat terjadi karena error bit acak pada saluran komunikasi. *Noise impulse* ini dapat dimodelkan sebagai :

$$g(x,y)\begin{cases} z(x,y) & dengan\ peluang\ p \\ f(x,y) & dengan\ peluang\ 1-p \end{cases} \quad \text{..... (1)}$$

Persamaan*ImpulseNoise*

Dimana :

g(x,y) = citra aslinya

z(x,y) dan f(x,y) = *noiseimpulse*-nya

Pada *noise impulsepixelnoise* sangat berbeda dari nilai *pixel* di sekelilingnya. Karakteristik dari *noise impulse* ini adalah nilai dari *pixelnoise* tidak memiliki hubungan dengan *pixel* citra aslinya.

2.3.1 Skema *Adaptive Noise Reduction*

Untuk (i, j) ∈ X = {1, …, N} x {1, …, N}, anggap x_{ij} adalah bentuk *gray* dari citra asli X. Level *gray* pada lokasi piksel (i, j) adalah sebagai berikut:

$$f(x_{ij}) = \begin{cases} 0 & dengan\ p/2 \\ x_{ij} & dengan 1-p \\ 255 & dengan\ p/2 \end{cases} \quad \text{..... (3)}$$

Dimana p adalah rasio *noise* yang terkontaminasi pada citra

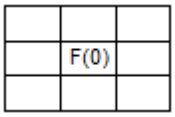
Skema *adaptive noise reduction* untuk *salt and pepper noise* yang diperkenalkan oleh Tina Gebreyohannes dan Dong Yoon Kim terdiri dari 3 tahapan yang dapat dijabarkan sebagai berikut:

1. Skema Pendeteksian *Noise*

   Untuk mengklasifikasikan piksel korup atau tidak, akan digunakan metode *Mean Absolute Gradient* (MAG). MAG yang kecil biasanya mengindikasikan sebuah daerah datar dan MAG yang besar biasanya mengindikasikan daerah kompleks atau daerah *impulse noise*. MAG dapat didefinisikan sebagai berikut:

   $$MAG = \frac{1}{N-1}\sum_{i=0}^{n-1} F(0) - F(i) \quad \text{..... (4)}$$

   dimana F(i) melambangkan nilai intensitas dari piksel pada sebuah region. N adalah total piksel pada region. F(0) adalah intensitas dari piksel pada bagian tengah seperti terlihat pada gambar berikut:



Gambar 3. *Window* MAG 3 x 3

$$z_{ij} = \begin{cases} 1 & MAG > T \\ 1 & x_{ij} = 0 \\ 1 & x_{ij} = 255 \\ 0 & MAG \le T \end{cases} \quad \text{..... (5)}$$

Sebuah nilai *threshold* T dapat ditentukan berdasarkan hasil pengujian. Jika Z_{ij} = 1, maka piksel X_{ij} akan ditandai sebagai kandidat *noise*, jika tidak, berarti X_{ij} tidak memiliki *noise*.

2. Skema Reduksi *Noise*

Untuk piksel *noise* yang terdeteksi, digunakan sebuah *window* M berukuran 3 x 3 seperti berikut:

$$M = \begin{bmatrix} x_{i-1,j-1} & x_{i,j-1} & x_{i+1,j-1} \\ x_{i-1,j} & x_{i,j} & x_{i+1,j} \\ x_{i-1,j+1} & x_{i,j+1} & x_{i+1,j+1} \end{bmatrix} \quad \text{..... (6)}$$

Dengan mengurutkan elemen pada M berdasarkan pada jaraknya dari piksel pusat (piksel tengah), diperoleh deretan terurut sebagai berikut:

a. Urutkan 5 buah elemen {x_{ij}, $x_{i-1,j-1}$, $x_{i+1,j-1}$, $x_{i-1,j+1}$ dan $x_{i+1,j+1}$) secara menaik, maka diperoleh {a_1, a_2, a_3, a_4, a_5} dimana {$a_1 < a_2 < a_3 < a_4 < a_5$}. Median = a_3.

b. Urutkan 4 buah elemen lainnya termasuk median dari langkah (a) {$x_{i,j-1}$, $x_{i-1,j}$,$x_{i+1,j}$,$x_{i,j+1}$ dan Median} secara menaik, maka diperoleh {b_1, b_2, b_3, b_4, b_5} dimana {$b_1 < b_2 < b_3 < b_4 < b_5$}.

Median = b_3.

Maka citra hasil restorasi y dengan y_{ij} dihitung dengan menggunakan rumusan berikut:

$$y_{ij} = \begin{cases} Median & jika \;\; b_3 \neq 0 \;\; or \; b_3 \neq 255 \\ Average(b_4, b_5) & jika \;\; b_3 = 0 \\ b_2 & jika \;\; b_4 = 0 \\ Average(b_1, b_2) & jika \; b_3 = 255 \\ b_1 & jika \; b_2 = 255 \end{cases} \qquad \text{…….. (7)}$$

3. Peningkatan Kualitas Citra (*Image Enhancement*)

Untuk menghasilkan detail dan sisi dari citra hasil restorasi, maka akan diaplikasikan filter langsung untuk meningkatkan kualitas citra. Untuk melakukan hal tersebut, maka harus diaplikasikan standar deviasi pada citra *noise* untuk menentukan apakah filter langsung perlu diterapkan atau tidak. Standar deviasi dapat didefinisikan sebagai berikut:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(F(i) - \mu)^2} \qquad \text{…….. (8)}$$

dimana F(i) melambangkan nilai intensitas dari citra *noise*. N adalah jumlah piksel pada citra dan μ adalah rata-rata dari citra *noise*. Jika standar deviasi lebih besar dari nilai *threshold* T, maka filter langsung akan diterapkan pada citra.

Sebuah piksel X_k pada citra akan dipartisi menjadi (Y_1, Y_2, Y_3, Y_4) dimana {X_1, X_2, X_4} termasuk pada Y_1, yang terdapat pada bagian kiri atas dari *mask*.

$$d_1 = | x_1 - x_2 | \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (9)$$

$$d_2 = | x_1 - x_4 | \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

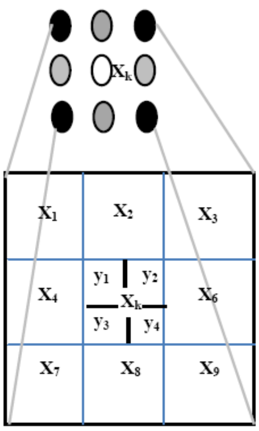
$$d_3 = | x_4 - x_2 | \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (11)$$

Sehingga:

$$y_1 = \begin{cases} avg(x_1, x_2) \ jika \ d_1 \leq \min(d_1, d_2, d_3) \\ avg(x_1, x_4) \ jika \ d_2 \leq \min(d_1, d_2, d_3) \\ avg(x_4, x_2) \ jika \ d_3 \leq \min(d_1, d_2, d_3) \end{cases} \ldots\ldots\ldots\ldots\ldots\ldots (12)$$

Y_2, Y_3 dan Y_4 dengan {X_2, X_3, X_6}, {X_4, X_7, X_8} dan {X_6, X_8, X_9} akan dihitung dengan cara yang sama.

$$X_K = \text{Average}(Y_1, Y_2, Y_3, Y_4) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (13)$$



Gambar 2.21*Window* Filter Langsung

2.3.2 Skema *Spatial Median Filter*

Algoritma untuk menentukan *spatial median* dari sekumpulan titik x_1, …, x_N dapat dijabarkan sebagai berikut:

1. Untuk setiap vektor x, hitung S yang merupakan sekumpulan nilai penjumlahan dari kedalaman spasial dari x ke setiap vektor lainnya.
2. Tentukan nilai kedalaman spasial maksimum dari kumpulan ini, S_{max}.
3. S_{max} adalah *spatial median* dari kumpulan titik.

Kedalaman spasial antara sebuah titik dan kumpulan titik dapat didefinisikan sebagai berikut:

$$S_{depth}(X, x_1, \ldots, x_N) = 1 - \frac{1}{N-1} \left\| \sum_{i=1}^{N} \frac{X - x_i}{\|X - x_i\|} \right\| \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (14)$$

2.4 Penilaian Kualitas Citra

Secara tradisional, kualitas citra telah dievaluasi oleh manusia dengan metode subyektif. Sejumlah pengamat dipilih dengan menguji kemampuan visual mereka, kemudian ditunjukkan serangkaian tes citra pada layar dan diminta menilai kualitas citra tersebut. Evaluasi subyektif ini dapat diandalkan namun biasanya sangat merepotkan, membutuhkan waktu yang lama dan mahal, sehingga terdapat model komputasi yang secara otomatis dapat memprediksi kualitas citra(Youssif, et al., 2010).

Yang paling sederhanan dan banyak digunakan dalam mengukur kualitas citra adalah metric mean squared error (MSE), dihitung dengan rata-rata

perbedaan intensitas kuadrat piksel gambar terdistorsi dan referensi, dengan kuantitas terkait peak signal to noise ratio (PSNR). Metrik tersebut menarik karena sederhana untuk menghitung, memiliki arti fisik yang jelas dan matematis dalam konteks optimasi. Tapi kedua metric tersebut sangat tidak cocok untuk dirasakan kualitas secara visual (Wang, et al., 2004).

2.4.1 Mean Square Error

Pengukuran yang paling sederhana dalalm pengukuran kualitas gambar adalah Mean Square Error (MSE). Nilai besar MSE berarti bahwa gambar berkualitas buruk. MSE didefinisikan sebagai berikut (Youssif, et al., 2010):

$$\mathbf{MSE}=\frac{1}{MN}\sum_{m=1}^{M}\sum_{n=2}^{N}(x(m,n)-\hat{x}(m,n))^2 \quad \text{..... (2-12)}$$

Keterangan :

MSE : nilai *Mean Square Error*

f(x,y) : intensitas citra asli

f'(x,y) : intensitas citra hasil filter

2.4.2 Peak Signal to Noise Ratio (PSNR)

Sebuah gambar berkualitas tinggi memiliki nilai kecil Peak Signal Noise Ratio (PSNR) . PSNR didefinisikan sebagai berikut (Youssif, et al., 2010):

$$\mathbf{PSNR}=\left[10\log\frac{255^2}{MSE}\right]. \quad \text{.....(2-13)}$$

Keterangan :

PSNR: nilai *Peak Signal to Noise Ratio*

MSE : nilai *Mean Squared Error*

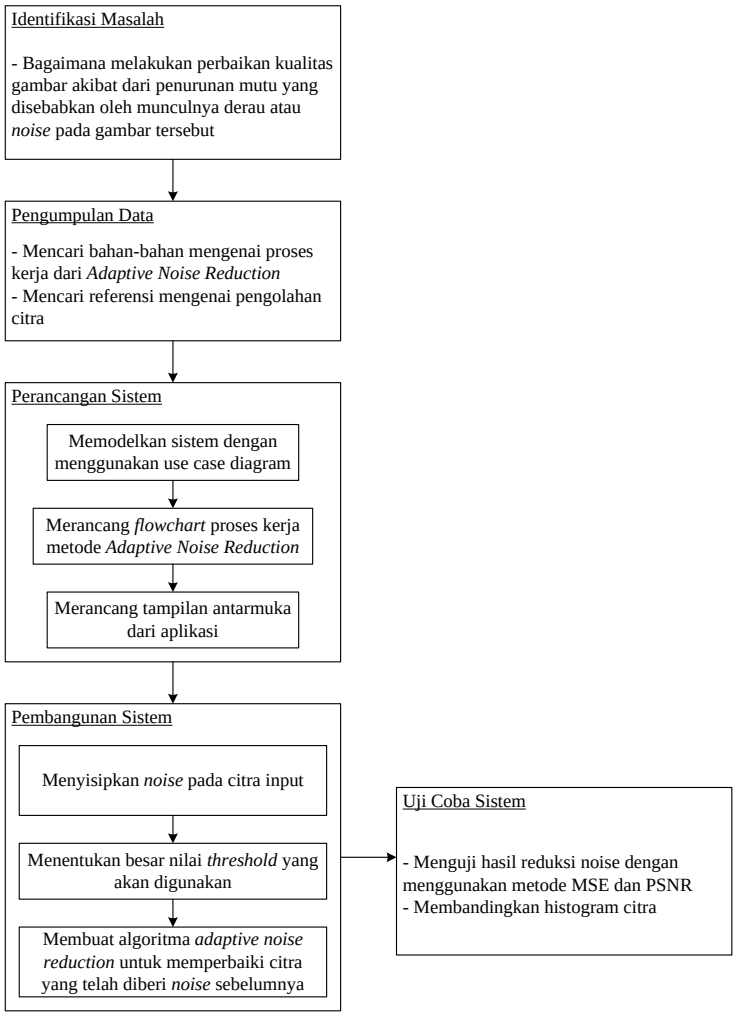
255nilai skala keabuan citra

BAB III

METODE PENELITIAN

3.1 Tahapan Penelitian

Proses analisa diperlukan untuk mengetahui kebutuhan dari perangkat lunak yang akan dibuat beserta aturan yang digunakan dalam perangkat lunak dan pemodelan sistem yang akan dibuat. Tahapan perancangan aplikasi pengolahan citra ini dapat dilihat pada gambar berikut:



Gambar 3.1 *Flowchart* Metode Penelitian

a. Analisis Sistem

Tahapan ini mencakup analisis proses kerja algoritma yang digunakan dengan menggunakan *flowchart diagram*. Setelah itu, akan dilakukan analisa kebutuhan sistem secara fungsional dengan menggunakan *use case diagram* dan non-fungsional dengan menggunakan kerangka PIECES (*Performance, Information, Economic, Control, Efficiency, Service*).

b. Perancangan Sistem

Pada tahap ini dilakukan perancangan tampilan antarmuka pada sistem (*user interface*) dengan aplikasi Microsoft Visual C#.

c. Pembuatan Sistem

Pada tahap ini dilakukan proses pengkodingan (*coding*) sistem dengan metode *Spatial Median Filter* dan *Adaptive Noise Reduction* menggunakan bahasa pemrograman C#.

d. Pengujian Sistem

Sistem yang telah selesai dibuat akan diuji dengan tujuan untuk mengetahui bagaimana cara kerja metode *Spatial Median Filter* dan *Adaptive Noise Reduction* yang telah diimplementasikan pada sistem. Pengujian akan dilakukan dengan menggunakan metode MSE dan PSNR.

3.2 Metode Pengumpulan Data

Teknik pengumpulan data dapat didefinisikan sebagai suatu cara yang digunakan untuk memperoleh data yang dibutuhkan sebagai bahan masukan bagi penulis dalam penyusunan skripsi ini.

Proses dimulai dengan mengumpulkan data-data yang diperlukan dalam penelitian, adapun metode pengumpulan data dalam penelitian dilakukan melalui Penelitian Kepustakaan (*library research*), yaitu penulis mengumpulkan data-data melalui *internet* dan buku-buku yang relevan yang berhubungan dengan topik yang dibahas.

3.3 Analisis Sistem yang Sedang Berjalan

Proses pengiriman citra digital melalui saluran komunikasi dapat mengakibatkan munculnya *impulse noise*. *Noise* ini dapat mengakibatkan citra digital menjadi rusak dan kualitas citra menjadi kurang bagus. Untuk mengatasi masalah ini, maka diperlukan sebuah aplikasi pengolahan citra yang mampu untuk menghapus *noise* yang terdapat pada citra digital. Oleh karena itu, maka perlu dirancang sebuah aplikasi *image enhancement* yang menerapkan metode *Spatial Median Filter* dan *Adaptive Noise Reduction*.

Proses kerja dari algoritma *Spatial Median Filter* adalah sebagai berikut:

a. Pemakai memasukkan citra yang akan dilakukan proses pemfilteran. Citra ini berupa citra berwarna yang berukuran persegi. Apabila citra *input* bukan citra persegi, maka akan dilakukan penambahan piksel putih pada citra input hingga citra berukuran persegi. Misalkan citra *input* berukuran 3 x 3 dengan warna piksel sebagai berikut:

(125, 117, 138)	(105, 78, 45)	(78, 125, 141)
(63, 19, 141)	(65, 87, 38)	(54, 65, 78)
(250, 119, 104)	(241, 220, 254)	(163, 155, 141)

b. Untuk setiap piksel pada citra, hitung nilai kedalaman spasialnya.

Misalkan ditentukan ukuran *mask* adalah 2 x 2 sehingga N = 4, maka proses perhitungan kedalaman spasial adalah sebagai berikut:

Piksel 1:

X = 125

x_1 = 125

x_2 = 105

x_3 = 63

x_4 = 65

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{N-1} \| \sum_{i=1}^{N} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \sum_{i=1}^{4} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \frac{125-125}{\| 125-125 \|} + \frac{125-105}{\| 125-105 \|} + \frac{125-63}{\| 125-63 \|} + \frac{125-65}{\| 125-65 \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| \frac{0}{0} + \frac{20}{20} + \frac{62}{62} + \frac{60}{60} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| 0 + 1 + 1 + 1 \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} * 3 = 1 - 1 = 0$$

X = 105

x_1 = 125

x_2 = 105

x_3 = 63

x_4 = 65

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{N-1} \| \sum_{i=1}^{N} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \sum_{i=1}^{4} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \frac{105-125}{\| 105-125 \|} + \frac{105-105}{\| 105-105 \|} + \frac{105-63}{\| 105-63 \|} + \frac{105-65}{\| 105-65 \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| \frac{-20}{20} + \frac{0}{0} + \frac{42}{42} + \frac{40}{40} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| -1 + 0 + 1 + 1 \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} * 1 = \frac{2}{3}$$

X = 63

x_1 = 125

x_2 = 105

x_3 = 63

x_4 = 65

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{N-1} \| \sum_{i=1}^{N} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \sum_{i=1}^{4} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \frac{63-125}{\| 63-125 \|} + \frac{63-105}{\| 63-105 \|} + \frac{63-63}{\| 63-63 \|} + \frac{63-65}{\| 63-65 \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| \frac{-62}{62} + \frac{-42}{42} + \frac{0}{0} + \frac{-2}{2} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| -1 + -1 + 0 + -1 \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} * 3 = 1 - 1 = 0$$

X = 65

x_1 = 125

x_2 = 105

x_3 = 63

x_4 = 65

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{N-1} \| \sum_{i=1}^{N} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \sum_{i=1}^{4} \frac{X - x_i}{\| X - x_i \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{4-1} \| \frac{65-125}{\| 65-125 \|} + \frac{65-105}{\| 65-105 \|} + \frac{65-63}{\| 65-63 \|} + \frac{65-65}{\| 65-65 \|} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| \frac{-60}{60} + \frac{-40}{40} + \frac{2}{2} + \frac{0}{0} \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} \| -1 + -1 + 1 + 0 \|$$

$$S_{depth}(X, x_1, x_2, x_3, x_4) = 1 - \frac{1}{3} * 1 = 1 - \frac{1}{3} = \frac{2}{3}$$

c. Nilai kedalaman maksimum yang terpilih pertama kali adalah piksel ke-2, x_2 dengan nilai *Red* dari piksel = 105, maka nilai piksel dengan kedalaman spasial maksimum yang terpilih adalah 105.

d. Warna *Red* dari piksel pertama diganti dari 125 menjadi 105.

e. Proses diatas akan diulangi untuk semua piksel lainnya.

f. Setelah warna *Red* dari semua piksel diproses, maka proses akan diulangi untuk warna *Green* dan *Blue* dari semua piksel.

Proses kerja dari algoritma *Adaptive Noise Reduction* ini dapat dirincikan sebagai berikut:

a. Pemakai memasukkan citra yang akan dilakukan proses pemfilteran. Citra ini berupa citra berwarna yang berukuran persegi. Apabila citra *input* bukan citra persegi, maka akan dilakukan penambahan piksel putih pada citra input hingga citra berukuran persegi. Misalkan citra *input* berukuran 3 x 3 dengan warna piksel sebagai berikut:

(125, 117, 138)	(105, 78, 45)	(78, 125, 141)
(63, 19, 141)	(65, 87, 38)	(54, 65, 78)
(250, 119, 104)	(241, 220, 254)	(163, 155, 141)

b. Setelah itu, pemakai memasukkan nilai *threshold*. Misalkan nilai *threshold* ditentukan sebesar 40.

c. Gunakan metode MAG untuk mendeteksi piksel *noise*.

Piksel 1 :

Window MAG dari piksel 1:

(0, 0, 0)	(0, 0, 0)	(0, 0, 0)
(0, 0, 0)	(125, 117, 138)	(105, 78, 45)
(0, 0, 0)	(63, 19, 141)	(65, 87, 38)

F(8)	F(1)	F(2)
F(7)	F(0)	F(3)
F(6)	F(5)	F(4)

Karena ukuran *window* MAG = 3 x 3, maka nilai N = 9.

$$MAG = \frac{1}{N-1}\sum_{i=0}^{n-1} F(0) - F(i)$$

$$MAG = \frac{1}{8}\sum_{i=0}^{8} F(0) - F(i)$$

$$MAG = \frac{1}{8}[(125-125) + (125-0) + (125-0) + (125-105) + (125-65) + (125-63) + (125-0) + (125-0) + (125-0)]$$

$$MAG = \frac{1}{8}[0+125+125+20+60+62+125+125+125]$$

$$MAG = \frac{1}{8} * 767 = 95.875$$

Karena nilai MAG yang diperoleh lebih besar daripada nilai *threshold* yang dimasukkan (95.875 > 40), maka Z_{ij} = 1. Hal ini berarti bahwa piksel pertama akan ditandai sebagai piksel *noise*.

d. Lakukan proses reduksi *noise*:

Karena piksel pertama terdeteksi sebagai *noise*, maka lakukan proses reduksi *noise* terhadap piksel pertama:

(0, 0, 0)	(0, 0, 0)	(0, 0, 0)
(0, 0, 0)	(125, 117, 138)	(105, 78, 45)
(0, 0, 0)	(63, 19, 141)	(65, 87, 38)

$$M = \begin{bmatrix} x_{i-1,j-1} & x_{i,j-1} & x_{i+1,j-1} \\ x_{i-1,j} & x_{i,j} & x_{i+1,j} \\ x_{i-1,j+1} & x_{i,j+1} & x_{i+1,j+1} \end{bmatrix}$$

Urutkan 5 buah elemen {x_{ij}, $x_{i-1,j-1}$, $x_{i+1,j-1}$, $x_{i-1,j+1}$ dan $x_{i+1,j+1}$) secara menaik, {125, 0, 0, 0, 65} diurutkan menjadi {0, 0, 0, 65, 125}. Nilai median yang diperoleh adalah 0.

Urutkan 4 buah elemen lainnya termasuk median dari langkah (a) {$x_{i,j-1}$, $x_{i-1,j}$,$x_{i+1,j}$,$x_{i,j+1}$ dan Median} secara menaik, {0, 0, 105, 63, 0} diurutkan menjadi {0, 0, 0, 63, 105}. Nilai median yang diperoleh adalah 0.

Maka citra hasil restorasi y dengan y_{ij} dihitung dengan menggunakan rumusan berikut:

Karena $b_3 = 0$, maka $y(1, 1) = (b_4 + b_5)/2 = (63 + 105) / 2 = 84$

e. Lakukan peningkatan kualitas citra (*image enhancement*)

Hitung nilai *mean* dan standar deviasi.

Mean = (0 + 0 + 0 + 0 + 125 + 105 + 0 + 63 + 65)/9 = 39.78

Standar deviasi = $\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(F(i)-\mu)^2}$

Standar deviasi =

$$\sigma = \sqrt{\frac{1}{9}[(0-39.78)^2 + (0-39.78)^2 + (0-39.78)^2 + (0-39.78)^2 + (125-39.78)^2 + (105-39.78)^2 + (0-39.78)^2 + (63-39.78)^2 + (65-39.78)^2]}$$

$$\sigma = \sqrt{\frac{1}{9}[1582.4484 + 1582.4484 + 1582.4484 + 1582.4484 + 7262.4484 + 4253.6484 + 1582.4484 + 539.1684 + 636.0484]}$$

$$\sigma = \sqrt{2289.28395555555555555555556}$$

$\sigma = 47.85$

Nilai standar deviasi 47.85 lebih besar dari nilai *threshold*, maka proses filter akan diterapkan.

Hitung nilai untuk $Y_1 = \{x_1, x_2, x_4\}$

$d_1 = |0 – 0| = 0$

$d_2 = |0 – 0| = 0$

$d_3 = |0 – 0| = 0$

sehingga:

$Y_1 = (0 + 0)/2 = 0$

Hitung nilai untuk $Y_2 = \{x_2, x_3, x_6\}$

$d_1 = |0 – 0| = 0$

$d_2 = |0 – 105| = 105$

$d_3 = |105 – 0| = 105$

sehingga:

$Y_2 = (0 + 0)/2 = 0$

Hitung nilai untuk $Y_3 = \{x_4, x_7, x_8\}$

$d_1 = |0 – 0| = 0$

$d_2 = |0 – 63| = 63$

$d_3 = |63 – 0| = 63$

sehingga:

$Y_3 = (0 + 0)/2 = 0$

Hitung nilai untuk $Y_4 = \{x_6, x_8, x_9\}$

$d_1 = |105 – 63| = 42$

$d_2 = |105 – 65| = 40$

$d_3 = |65 – 63| = 2$

sehingga:

$Y_4 = (63 + 65)/2 = 64$

Jadi, nilai piksel sekarang diubah menjadi:

$Xk = (0 + 0 + 0 + 64) / 4 = 16$

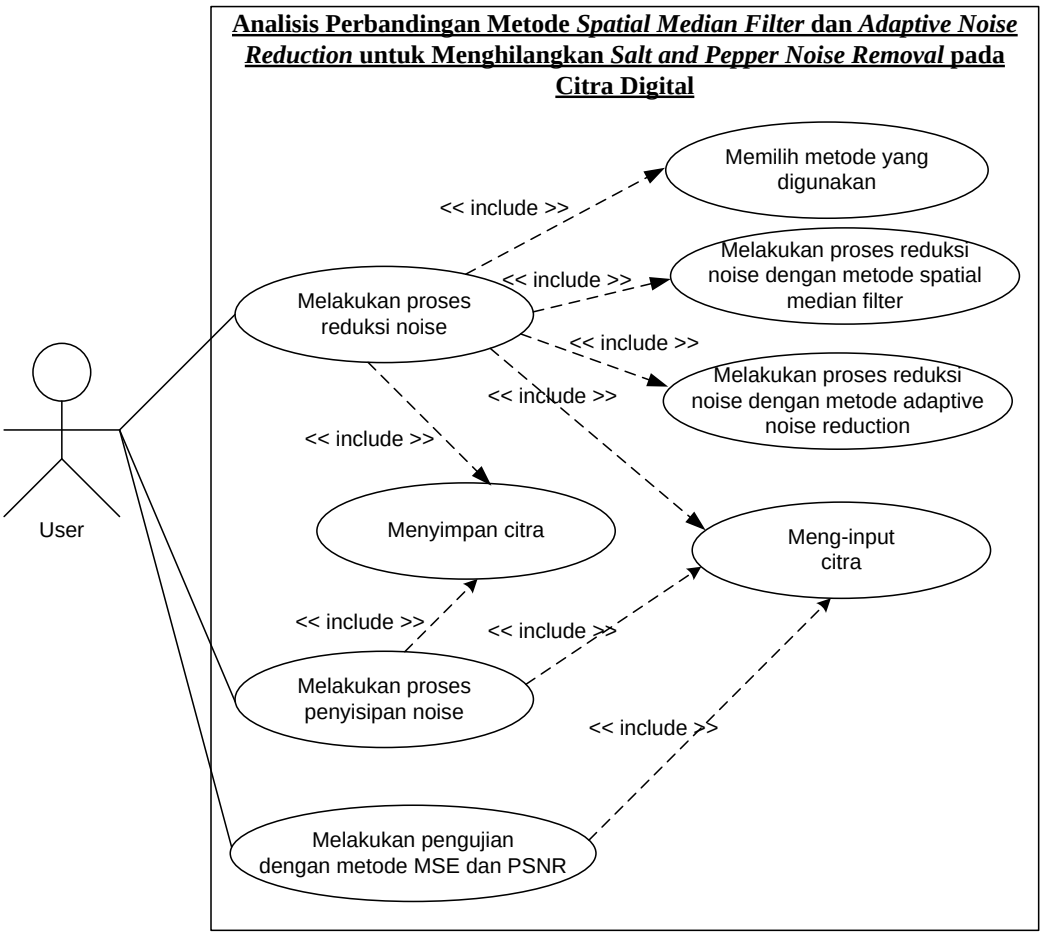
f. Proses diatas akan dilakukan untuk piksel lainnya hingga semua piksel pada citra diproses.

3.4 Rancangan Penelitian

3.4.1 Perancangan UML

1. Use Case Diagram

Aplikasi *Image Enhancment* dengan menggunakan *Adaptive Noise Reduction* ini dapat dimodelkan dengan menggunakan *use case diagram* seperti terlihat pada gambar berikut:



Gambar 3.2 *Use Case* Diagram dari Sistem

Tabel 3.1 Narasi dari *Use Case* Menginput Citra

Nama *use case*	Menginput Citra
Aktor	*User*
Deskripsi	*Use case* ini berfungsi untuk melakukan proses pengisian citra *input*.
Prakondisi	Data citra yang diperlukan dalam proses reduksi *noise* belum dimasukkan.
Sasaran	*Use case* ini diawali saat *user* ingin memasukkan citra *input* yang diperlukan dalam proses reduksi *noise*.

	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik tombol ‘Browse’. 3. *User* memilih *file* citra yang diinginkan. 4. *User* mengklik tombol ‘Open’.	2. Sistem menampilkan kotak dialog open. 5. Sistem membaca isi *file* citra dan menampilkannya di *picturebox*.
Bidang alternatif	Alt. Langkah 4. *User* mengklik tombol ‘Cancel’. Alt. Langkah 5. Sistem menutup kotak dialog dan isi *file* citra tidak dibaca.	
Kesimpulan	*Use case* ini untuk menginput data citra yang diperlukan	
Postkondisi	Data *input* citra telah disimpan ke dalam memori sementara.	

Tabel 3.2 Narasi dari *Use Case* Menyimpan Citra

Nama *use case*	Menyimpan Citra
Aktor	*User*
Deskripsi	*Use case* ini berfungsi untuk menyimpan citra hasil ke sebuah *file* citra
Prakondisi	*File* citra yang belum disimpan
Sasaran	*Use case* ini diawali saat *user* ingin menyimpan *file* citra

	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik *link* 'Browse'	2. Sistem membuka kotak dialog *save* untuk pengisian nama citra hasil dan memilih lokasi penyimpanan citra
Bidang alternatif	-	
Kesimpulan	*Use case* ini digunakan untuk menyimpan *file* citra	
Postkondisi	Citra telah disimpan	

Tabel 3.3 Narasi dari *Use Case* Melakukan Proses Reduksi *Noise* dengan Metode *Adaptive Noise Reduction*

Nama *use case*	Melakukan Proses Reduksi *Noise* dengan Metode *Adaptive Noise Reduction*	
Aktor	*User*	
Deskripsi	*Use case* ini berfungsi untuk melakukan proses reduksi *noise* dengan metode *Adaptive Noise Reduction*.	
Prakondisi	Citra input belum dilakukan proses *reduksi noise*.	
Sasaran	*Use case* ini diawali saat *user* ingin melakukan proses reduksi *noise* dengan metode *Adaptive Noise Reduction*.	
	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik tombol 'Proses' pada form	2. Sistem menampilkan *form* 'Reduksi Noise'.

	Pemilihan Metode. 3. *User* memilih citra input. 4. *User* mengklik tombol ‘Proses’.	5. Sistem mengecek apakah citra input telah dimasukkan. Jika valid, maka proses dilanjutkan. 6. Sistem melakukan proses reduksi *noise* terhadap citra input dengan menggunakan metode *Adaptive Noise Reduction*. 7. Sistem menampilkan citra hasil reduksi *noise*.
Bidang alternatif	Alt. Langkah 5 : Sistem menampilkan pesan kesalahan bahwa data citra belum lengkap ataupun ukuran citra kurang dari 100 x 100 piksel.	
Kesimpulan	*Use case* ini untuk melakukan proses reduksi *noise*	
Postkondisi	Citra telah direduksi *noise*-nya	

Tabel 3.4 Narasi dari *Use Case* Melakukan Proses Reduksi *Noise* dengan Metode *Spatial Median Filter*

Nama *use case*	Melakukan Proses Reduksi *Noise* dengan Metode *Spatial Median Filter*	
Aktor	*User*	
Deskripsi	*Use case* ini berfungsi untuk melakukan proses reduksi *noise* dengan metode *spatial median filter*.	
Prakondisi	Citra input belum dilakukan proses *reduksi noise*.	
Sasaran	*Use case* ini diawali saat *user* ingin melakukan proses reduksi *noise* dengan metode *spatial median filter*.	
	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik tombol 'Proses' pada form Pemilihan Metode. 3. *User* memilih citra input. 4. *User* mengklik tombol 'Proses'.	2. Sistem menampilkan *form* 'Reduksi Noise'. 5. Sistem mengecek apakah citra input telah dimasukkan. Jika valid, maka proses dilanjutkan. 6. Sistem melakukan proses reduksi *noise* terhadap citra input dengan menggunakan

		metode *Spatial Median Filter*. 7. Sistem menampilkan citra hasil reduksi *noise*.
Bidang alternatif	Alt. Langkah 5 : Sistem menampilkan pesan kesalahan bahwa data citra belum lengkap ataupun ukuran citra kurang dari 100 x 100 piksel.	
Kesimpulan	*Use case* ini untuk melakukan proses reduksi *noise*	
Postkondisi	Citra telah direduksi *noise*-nya	

Tabel 3.5 Narasi dari *Use Case* Melakukan Proses Penyisipan *Noise*

Nama *use case*	Melakukan Proses Penyisipan *Noise*	
Aktor	*User*	
Deskripsi	*Use case* ini berfungsi untuk melakukan proses penyisipan *noise* ke dalam citra *input*.	
Prakondisi	Citra input belum ditambahkan *noise*.	
Sasaran	*Use case* ini diawali saat *user* ingin menambahkan *noise* pada citra *input*.	
	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik menu ‘Penambahan Noise’. 3. *User* memilih citra input.	2. Sistem menampilkan *form* ‘Penyisipan Noise’.

	4. *User* memasukkan persentase *noise*. 5. *User* mengklik tombol 'Proses'.	6. Sistem mengecek apakah citra *input* telah dimasukkan. Jika valid, maka proses dilanjutkan. 7. Sistem menambahkan *noise* terhadap citra input dengan jumlah *noise* menurut persentase *noise* yang dimasukkan *user*. 8. Sistem menampilkan citra hasil penyisipan *noise*.
Bidang alternatif	Alt. Langkah 6 : Sistem menampilkan pesan kesalahan bahwa data citra belum lengkap ataupun ukuran citra kurang dari 100 x 100 piksel.	
Kesimpulan	*Use case* ini untuk melakukan proses penyisipan *noise*	
Postkondisi	Citra telah disisipkan *noise*-nya	

Tabel 3.6 Narasi dari *Use Case* Melakukan Pengujian dengan Metode MSE dan PSNR

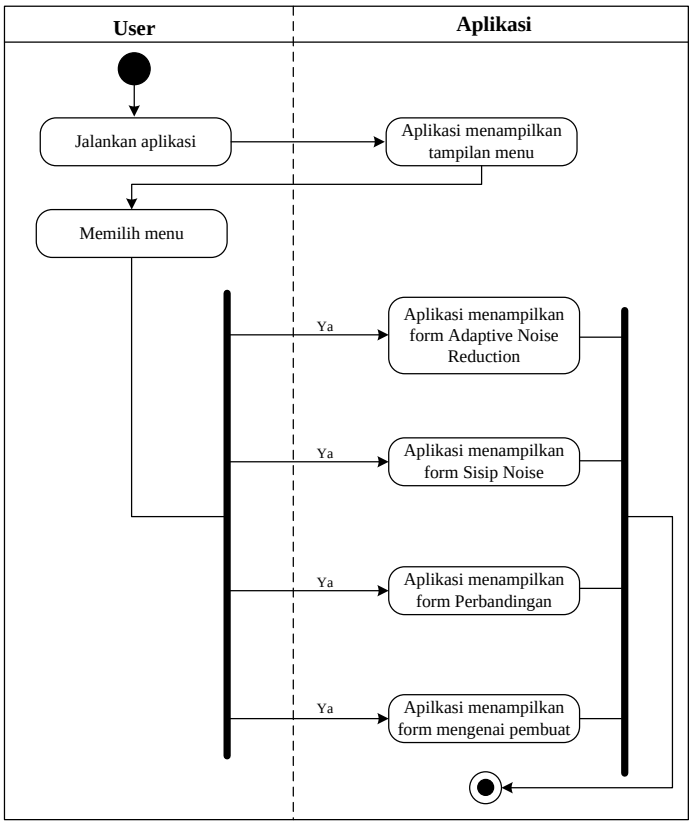
Nama *use case*	Melakukan Pengujian dengan Metode MSE dan PSNR
Aktor	*User*

Deskripsi	*Use case* ini berfungsi untuk melakukan proses pengujian dengan menggunakan metode MSE dan PSNR.	
Prakondisi	Informasi hasil pengujian MSE dan PSNR belum diperoleh.	
Sasaran	*Use case* ini diawali saat *user* ingin melakukan pengujian dengan menggunakan metode MSE dan PSNR.	
	Aksi Aktor	**Respons Sistem**
Bidang khas suatu event	1. *User* mengklik menu ‘Perbandingan Kinerja’. 3. *User* memilih citra input dan citra hasil. 4. *User* mengklik tombol ‘Proses’.	2. Sistem menampilkan *form* ‘Perbandingan’. 5. Sistem mengecek apakah citra *input* telah dimasukkan. Jika valid, maka proses dilanjutkan. 6. Sistem menghitung nilai MSE dan PSNR antara citra input dan citra hasil. 7. Sistem menampilkan nilai MSE dan PSNR antara citra input dan citra hasil.
Bidang alternatif	Alt. Langkah 5 : Sistem menampilkan pesan kesalahan bahwa data citra belum lengkap ataupun ukuran citra	

	kurang dari 100 x 100 piksel.
Kesimpulan	*Use case* ini untuk melakukan proses pengujian dengan menggunakan metode MSE dan PSNR.
Postkondisi	Informasi mengenai nilai MSE dan PSNR dari citra input dan citra hasil telah diperoleh.

2. Activity Diagram

Prosedur kerja dari aplikasi *image enhancement* ini dapat digambarkan seperti terlihat pada gambar *activity diagram* berikut:

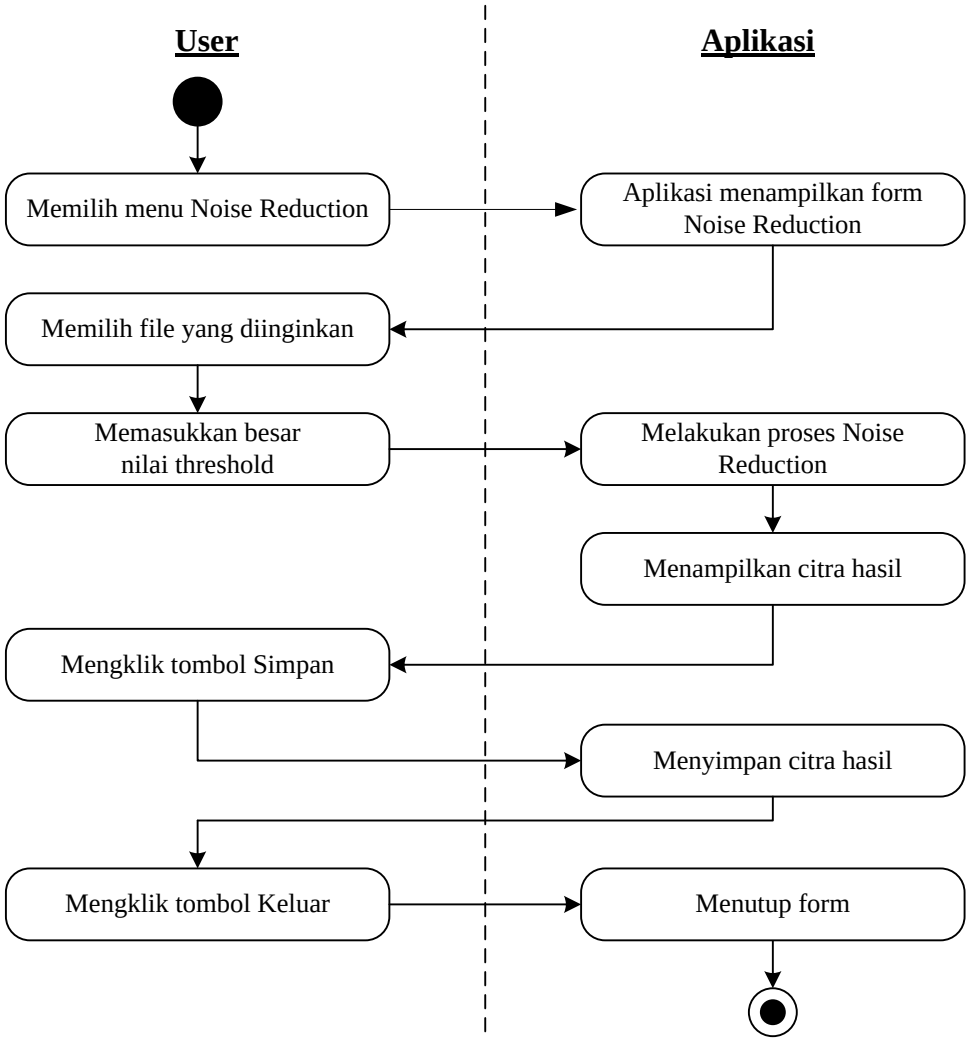


Gambar 3.3 *Activity Diagram* dari Menu

Pada saat *user* menjalankan aplikasi, aplikasi pengacakan citra akan menampilkan menu pilihan untuk *user*. Apabila *user* memilih menu penyisipan

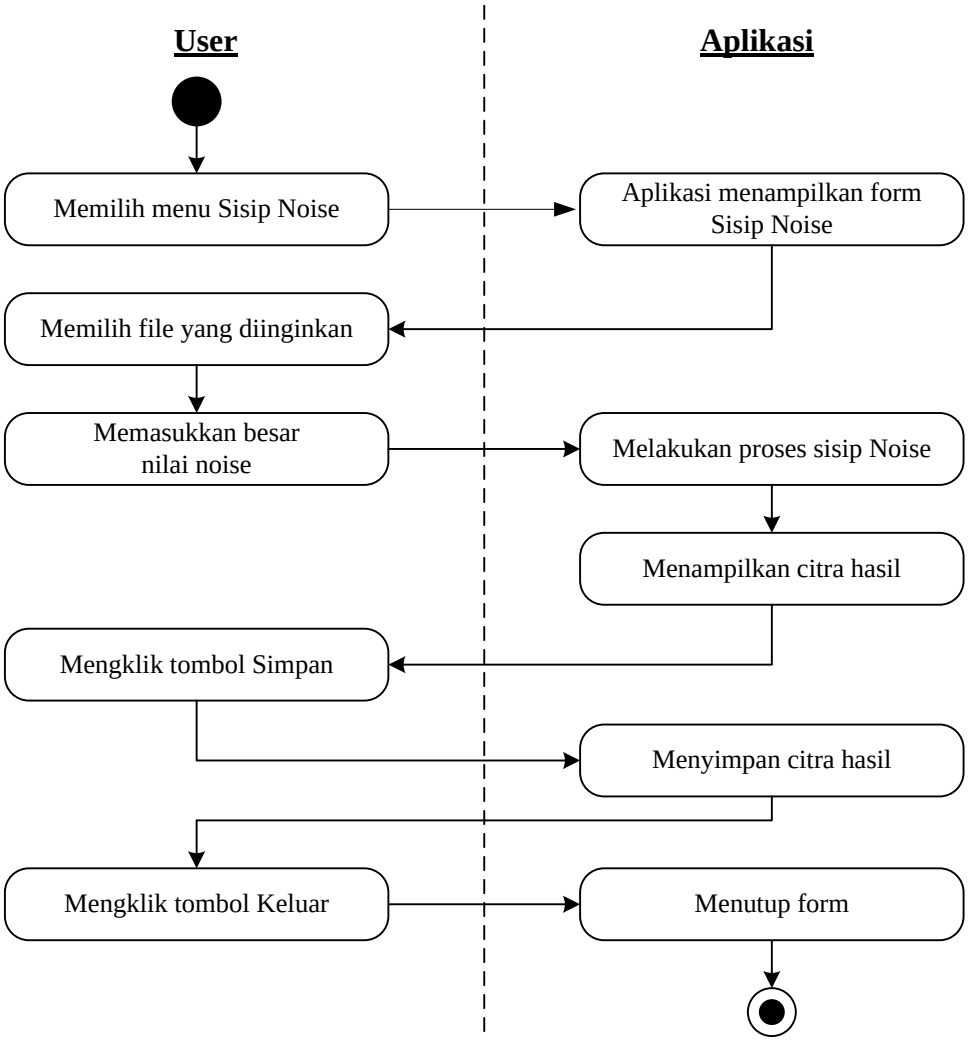
pesan, maka aplikasi akan menampilkan *form* penyisipan pesan. Sementara itu, apabila *user* memilih menu ekstraksi pesan maka aplikasi pengacakan citra akan menampilkan *form* ekstraksi pesan. Terakhir, apabila *user* memilih menu mengenai pembuat, maka aplikasi pengacakan citra akan menampilkan *form* mengenai pembuat.

Rancangan *activity diagram* yang menggambarkan proses *Noise Reduction* dapat dilihat pada gambar 3.4.



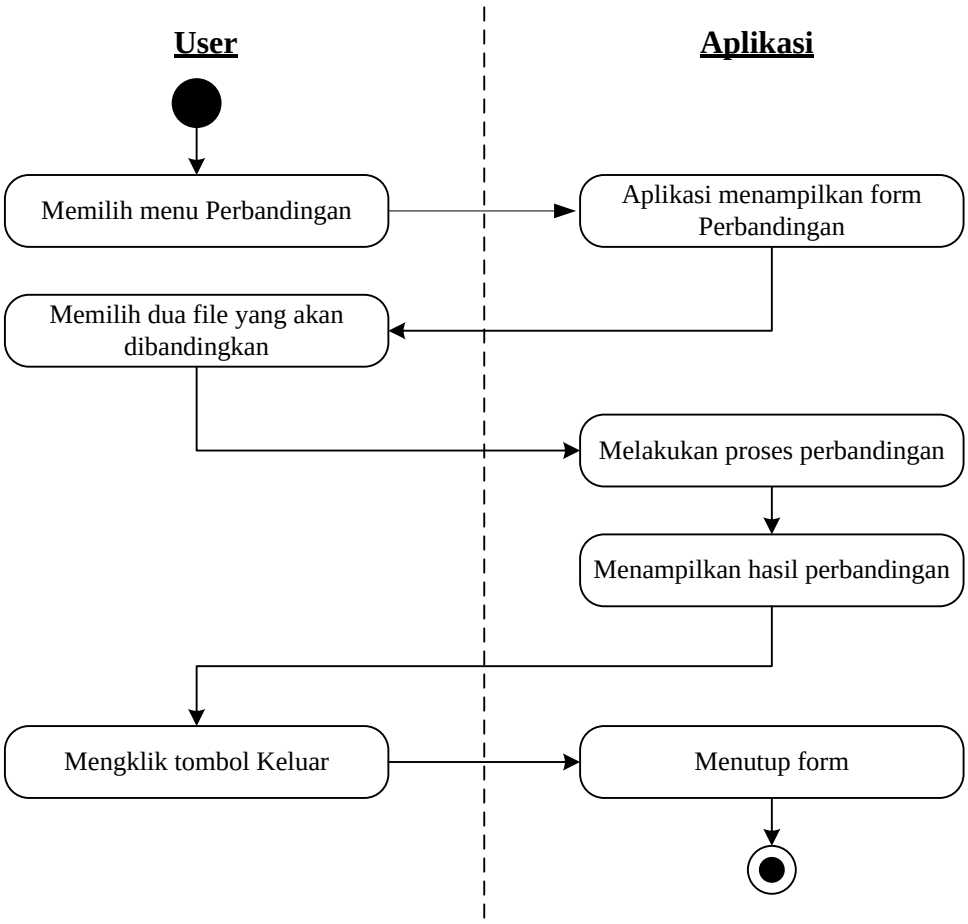
Gambar 3.4 Rancangan *Activity Diagram* untuk Proses *Noise Reduction*

Rancangan *activity diagram* yang menggambarkan proses Sisip *Noise* dapat dilihat pada gambar 3.4.



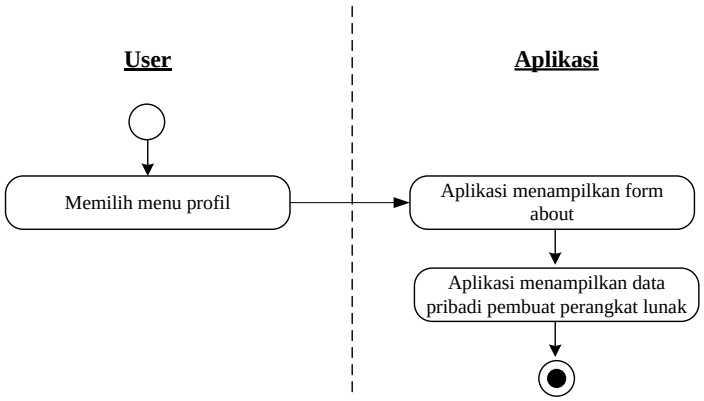
Gambar 3.5 Rancangan *Activity Diagram* untuk Proses Sisip *Noise*

Rancangan *activity diagram* yang menggambarkan proses Perbandingan dapat dilihat pada gambar 3.6.



Gambar 3.6 Rancangan *Activity Diagram* untuk Proses Perbandingan

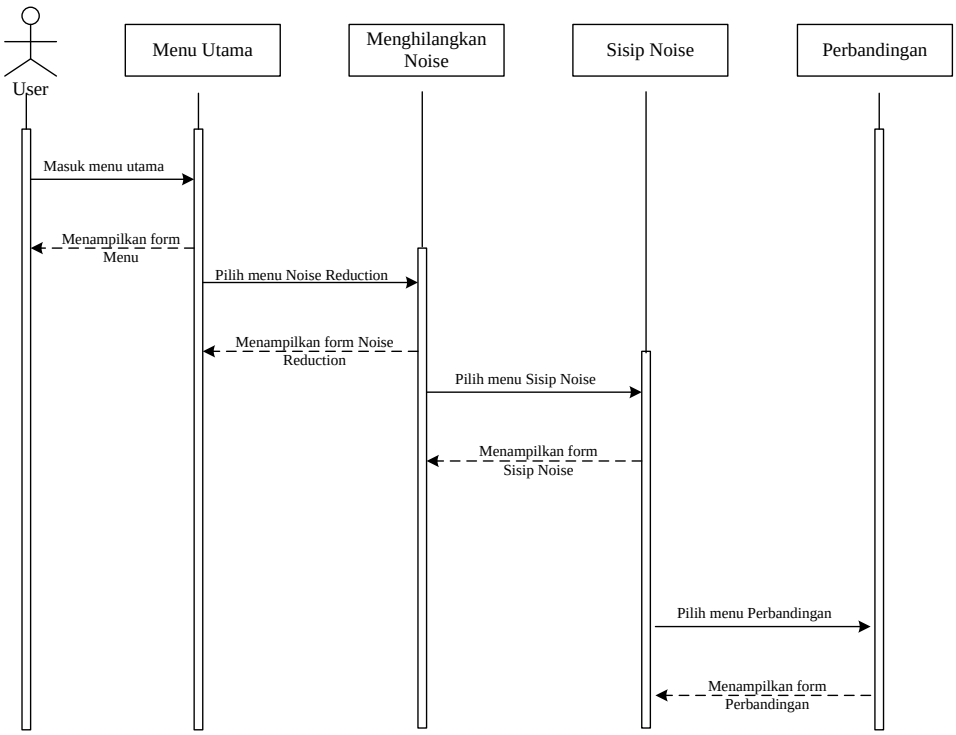
Rancangan *activity diagram* yang menggambarkan proses mengenai pembuat dapat dilihat pada gambar 3.7 Setelah *user* memilih *form* mengenai pembuat maka sistem akan menampilkan informasi mengenai pembuat aplikasi.



Gambar 3.7 Rancangan Activity Diagram untuk Proses Mengenai Pembuat

3. Sequence Diagram

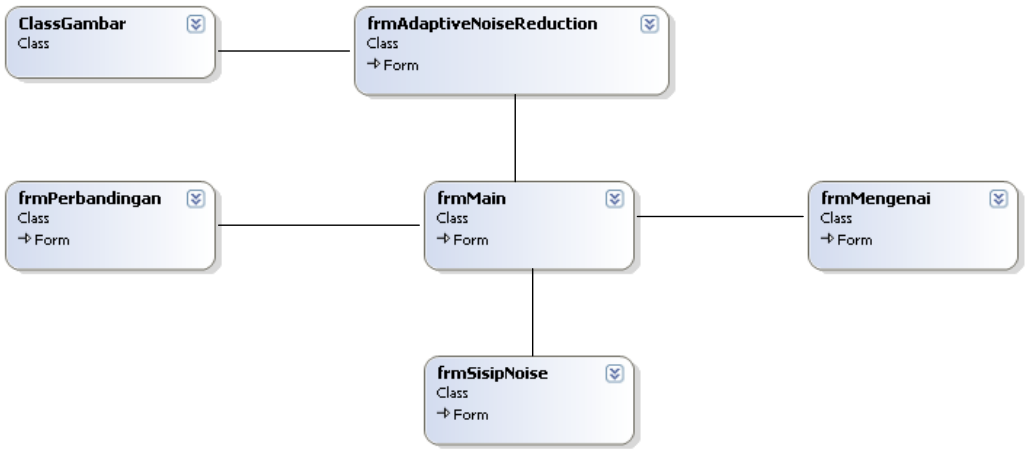
Sequence diagram ini akan menggambarkan rangkaian langkah – langkah yang menggambarkan *respon* dari *event* pada aplikasi ini, gambaran dari *sequence* diagram pada aplikasi ini dapat dilihat pada gambar 3.8.



Gambar 3.8 *Sequence Diagram*

4. Class Diagram

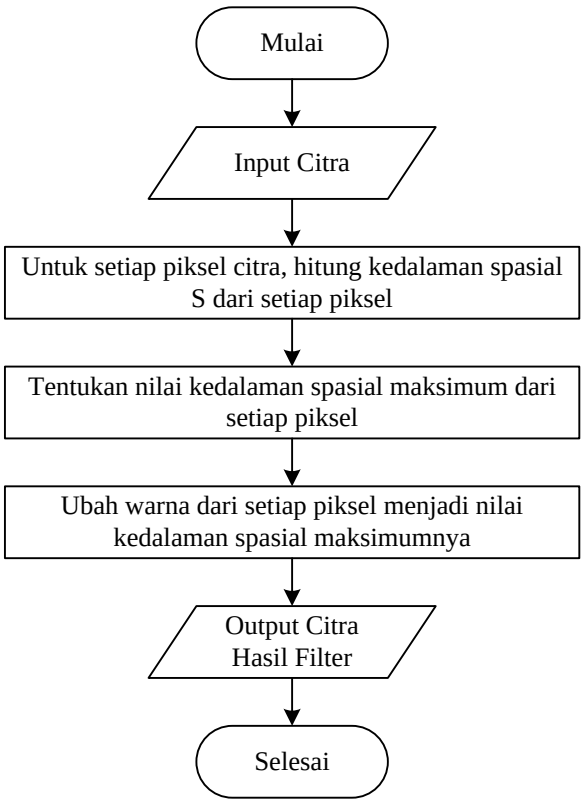
Class diagram untuk aplikasi *image enhancement* dapat dilihat pada gambar 3.9. Berdasarkan gambar dibawah, sistem akan menampilkan menu utama yang merupakan pusat kontrol dari segala aktifitas di dalam aplikasi, di dalam menu utama terdapat *form* yang berisi *label link* untuk menuju ke *form Adaptive Noise Reduction*, *form Sisip Noise* dan *form* Perbandingan serta *form* Mengenai.



Gambar 3.9 *Class Diagram*

3.4.2 Flowchart Metode Adaptive Median Filter

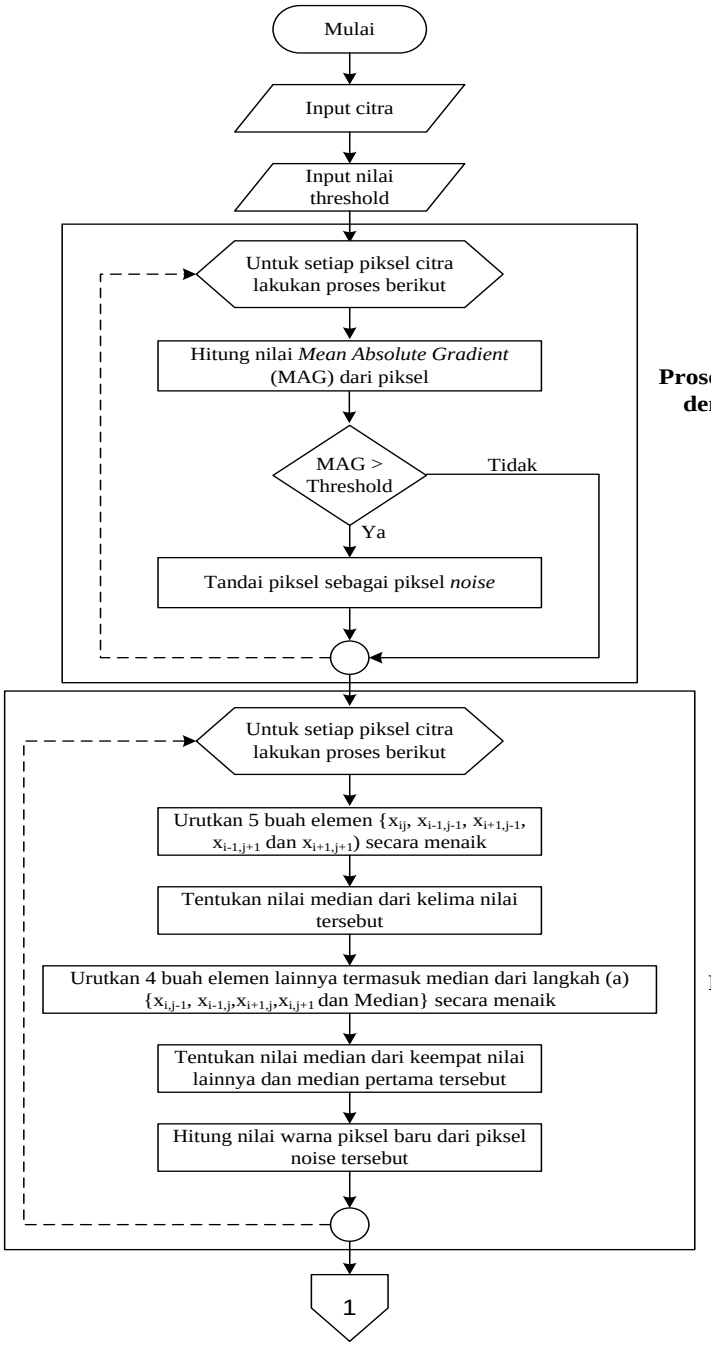
Proses ini akan melakukan proses reduksi *noise* dengan menggunakan metode *Spatial Median Filter*. Prosedur kerja dari proses *Spatial Median Filter* ini dapat dideskripsikan seperti terlihat pada gambar berikut:



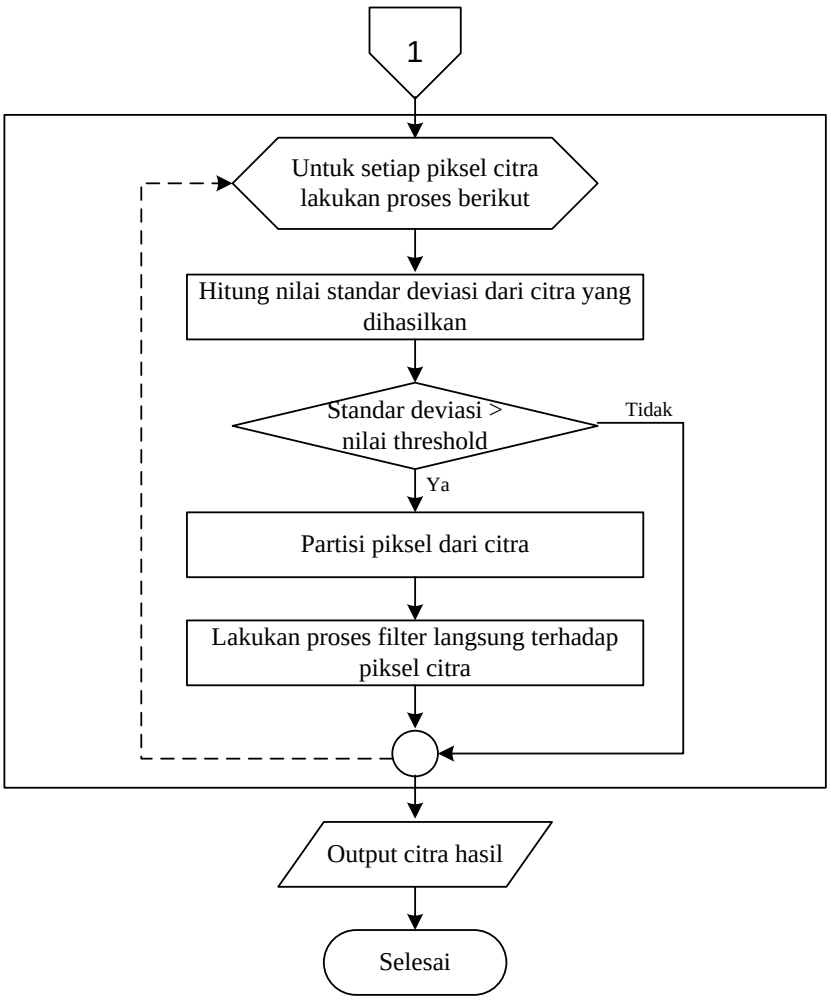
Gambar 3.10 *Flowchart* Proses *Spatial Median Filter*

3.4.3 Flowchart Metode Adaptive Noise Reduction

Langkah-langkah di atas dapat dirumuskan melalui gambar berikut yang ada dibawah ini:



Gambar 3.10 *Flowchart* Proses *Adaptive Noise Reduction*



Gambar 3.11 *Flowchart* Proses *Adaptive Noise Reduction(lanjutan)*

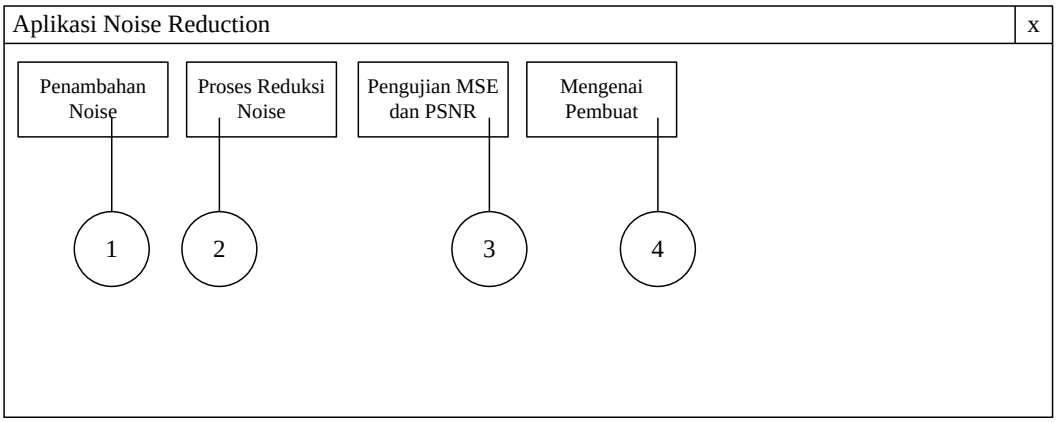
Proses kerja dari metode *Adaptive Noise Reduction* terbagi menjadi tiga tahapan besar yaitu proses pendeteksian *noise* dengan menggunakan metode MAG, proses reduksi *noise* dan proses *image enhancement*. Proses pendeteksian noise berfungsi untuk mengecek apakah sebuah piksel akan ditandai sebagai *noise* atau tidak yaitu dengan menghitung nilai *Mean Absolute Gradient* (MAG). Sedangkan, proses reduksi *noise* berfungsi untuk membuang *noise* yang terdapat pada citra. Terakhir, akan dilakukan proses *image enhancement* untuk

meningkatkan kualitas dari citra yang telah direduksi *noise*-nya. Proses *image enhancement* ini menggunakan nilai threshold yang dimasukkan oleh pemakai.

3.4.4 Perancangan Interface

1. *Form* 'Main'

Form ini merupakan *form* inti dari perangkat lunak yang berfungsi untuk menghubungkan *form-form* yang ada pada perangkat lunak. Rancangan tampilan dari *form* 'Main' ini dapat dilihat pada gambar 3.12 berikut:



Gambar 3.12 Rancangan *Form* 'Main'

Keterangan :

1 : *Button* 'Penambahan Noise', yang berfungsi untuk menampilkan *form* penambahan *salt and pepper noise*.

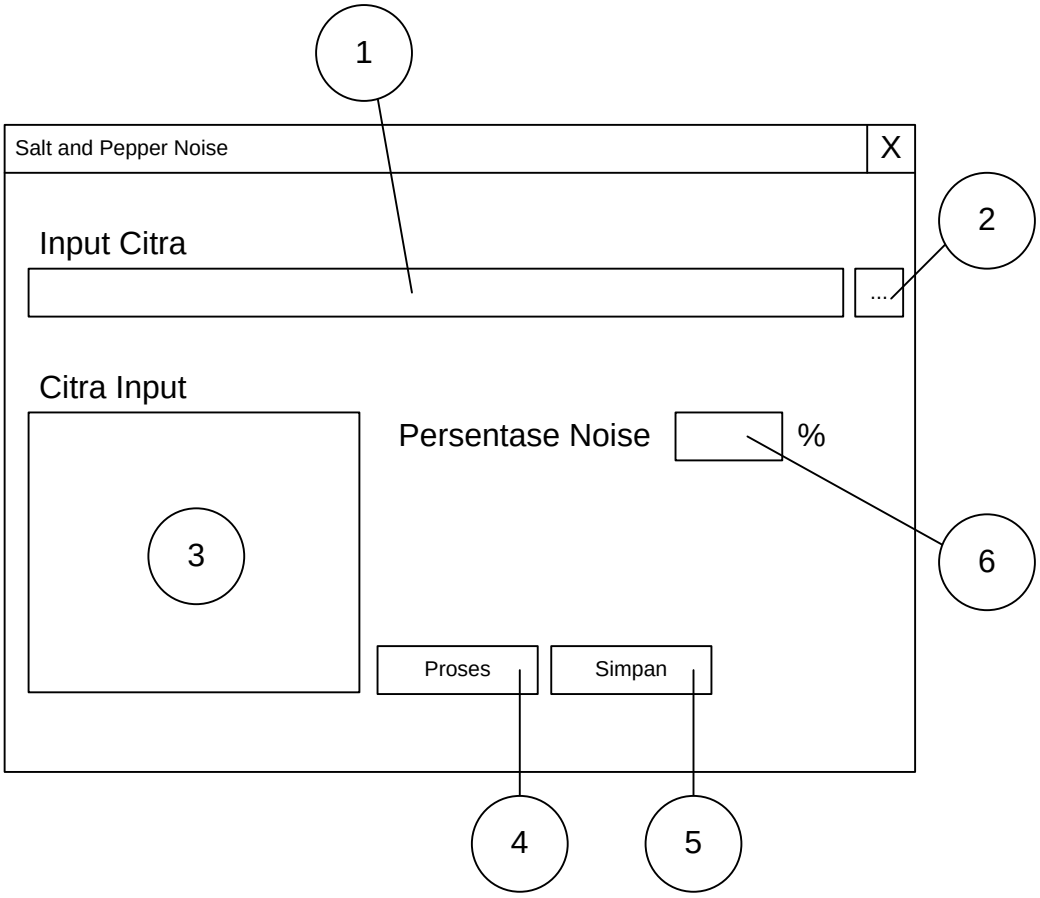
2 : *Button* 'Proses Reduksi *Noise*', yang berfungsi untuk menampilkan *form* reduksi *noise* dengan metode *Spatial Median Filter* atau *Adaptive Noise Reduction*.

3 : *Button* 'Pengujian MSE dan PSNR', yang berfungsi untuk menampilkan *form* Perbandingan.

4 : *Button* ’Mengenai Pembuat’, yang berfungsi untuk menampilkan *form* Mengenai.

2. *Form* ‘Penyisipan Noise’

Form ini berfungsi untuk melakukan penyisipan *noise* ke dalam sebuah citra tetapi tidak ditampilkan pada program karena tujuan dari skripsi ini lebih fokus ke reduksi *noise*. Rancangan tampilan dari *form* ‘Penyisipan Noise’ dapat dilihat pada gambar 3.13 berikut:



Gambar 3.13 Rancangan *Form* ‘Penyisipan Noise’

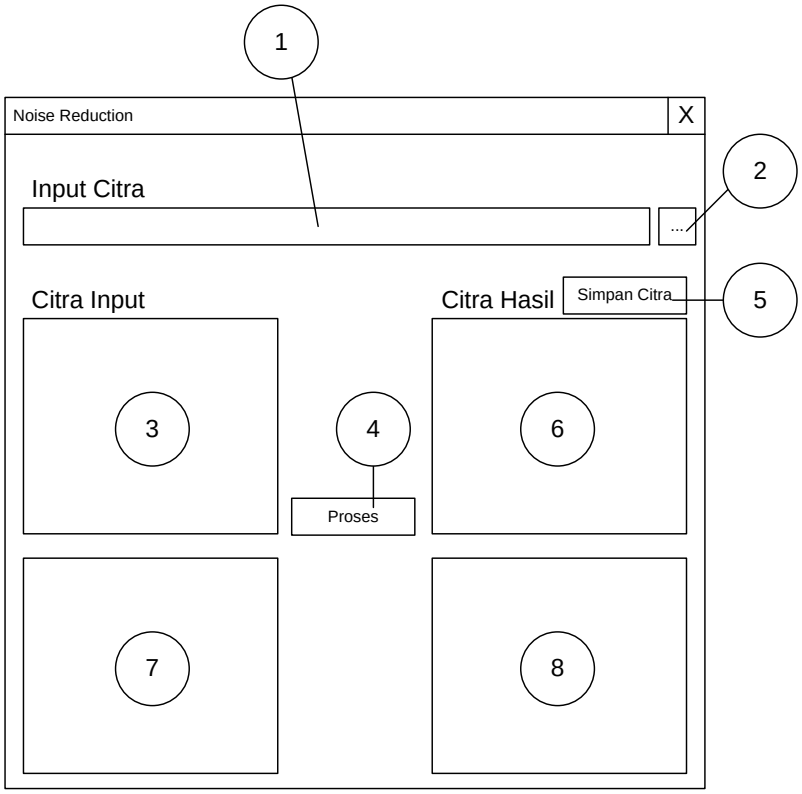
Keterangan:

1. Input citra ke dalam aplikasi melalui tombol ‘*browse*’.

2. Tombol *browse* untuk memilih *file* yang diinginkan.
3. Hasil gambar *input*.
4. Tombol proses untuk melakukan proses penambahan *salt and pepper noise* pada citra.
5. Tombol simpan untuk menyimpan citra hasil penambahan *noise*.
6. *Textbox* sebagai tempat pengisian persentase besar nilai *noise*.

3. *Form* ‘Reduksi Noise’

Form ini berfungsi untuk melakukan proses reduksi *noise* terhadap citra input. Rancangan tampilan dari *form* ‘Reduksi Noise’ dapat dilihat pada gambar 3.14 :



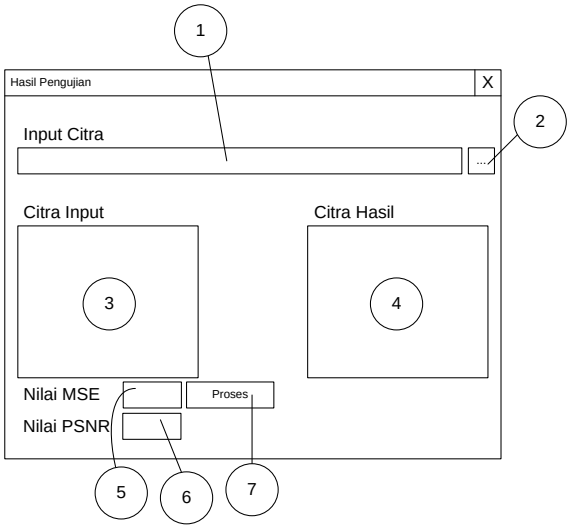
Gambar 3.14 Rancangan *Form* ‘Reduksi Noise’

Keterangan:

1. Input citra ke dalam aplikasi melalui tombol '*browse*'.
2. Tombol *browse* untuk memilih *file* yang diinginkan.
3. Hasil gambar *input*.
4. Tombol 'Proses' untuk memulai proses reduksi *noise* dengan menggunakan algoritma yang dipilih.
5. Tombol 'Simpan Citra' untuk menyimpan citra hasil.
6. Tampilan gambar hasil.
7. Tampilan gambar histogram untuk citra input.
8. Tampilan gambar histogram untuk citra hasil.

4. *Form* 'Perbandingan (MSE dan PSNR)'

Form ini berfungsi untuk melakukan perbandingan antara citra input dan citra hasil dengan menggunakan metode MSE dan PSNR. Rancangan tampilan dari *form* 'Ekstraksi' dapat dilihat pada gambar 3.15 :



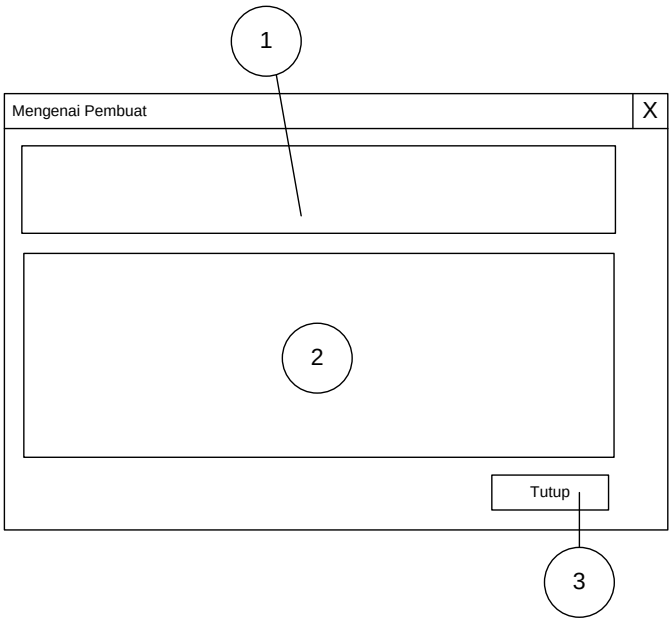
Gambar 3.15 Rancangan *Form* 'Perbandingan (MSE dan PSNR)'

Keterangan:

1. Input citra ke dalam aplikasi melalui tombol '*browse*'.
2. Tombol *browse* untuk memilih *file* yang diinginkan.
3. Hasil gambar *input*.
4. Tampilan gambar hasil.
5. *Textbox* sebagai tempat menampilkan nilai MSE.
6. *Textbox* sebagai tempat menampilkan nilai PSNR.
7. Tombol 'Proses' untuk memulai proses perbandingan.

5. *Form* 'Mengenai Pembuat'

Form ini berfungsi untuk menampilkan informasi mengenai data pribadi pembuat perangkat lunak. Rancangan tampilan dari *form* 'Mengenai Pembuat' dapat dilihat pada gambar 3.16:



Gambar 3.16 Rancangan *Form* 'Mengenai Pembuat'

Keterangan:

1. Daerah tampilan nama perangkat lunak.
2. Daerah tampilan data pribadi dari pembuat perangkat lunak.
3. Tombol ‘Tutup’ untuk menutup *form*.

BAB IV

HASIL DAN PEMBAHASAN

4.1 Kebutuhan Perangkat Keras dan Perangkat Lunak

Perangkat keras dan perangkat lunak yang disarankan untuk menggunakan aplikasi citra ini yaitu:

1. **Perangkat keras / *Hardware***

   Perangkat keras/ *Hardware* komputer yang disarankan terdiri dari:

   1. Minimal *Prosesor Intel Pentium Core i-3 2.1 GHz*.
   2. Minimal *Memory RAM* 2 *Gb*.
   3. Minimal *Monitor* dengan resolusi 800 × 600 *pixel*.
   4. *Keyboard* dan *Mouse*.
   5. Minimal *Harddisk* 160 *Gb*.

2. **Perangkat lunak / *Software***

   Perangkat lunak yang digunakan dalam menjalankan aplikasi citra ini terdiri dari:

   1. Minimal Sistem Operasi Windows XP 32-bit.
   2. Windows Photo Viewer.

4.2 Hasil

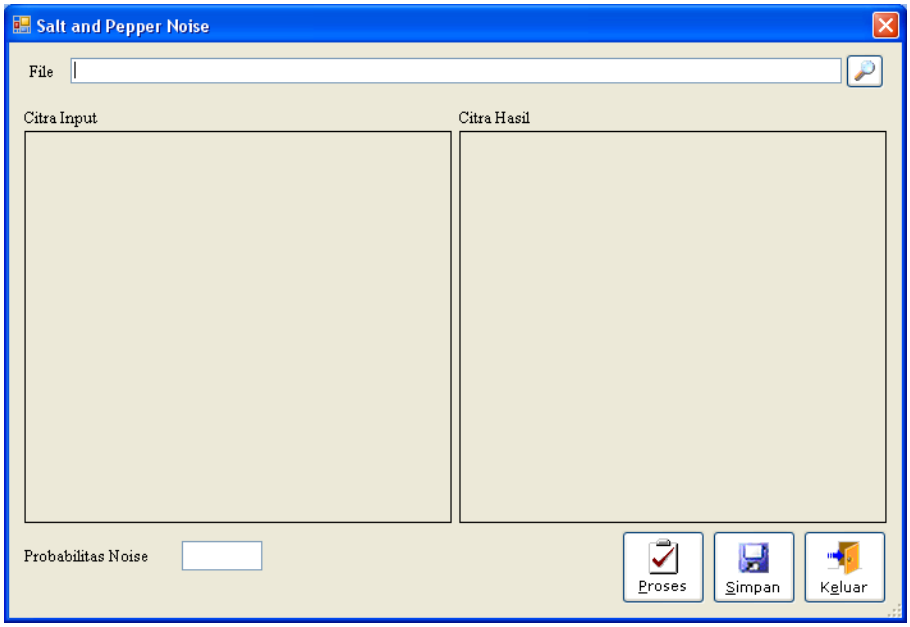
Setelah sistem dianalisis dan didesain secara rinci, maka akan menuju tahap implementasi. Implementasi merupakan tahap meletakkan sistem sehingga

siap untuk dioperasikan. Implementasi bertujuan untuk mengkonfirmasi modul-modul perancangan, sehinga pengguna dapat memberikan masukan kepada pembangun sistem. Hasil dari sistem yang dibangun tersebut dapat dilihat sebagai berikut:

Pada saat pertama kali menjalankan aplikasi *Salt and Pepper Noise Removal* dengan menggunakan *Spatial Median Filter* dan *Adaptive Noise Reduction*, maka *form* yang akan muncul pertama kali adalah *form* 'Main', yang dapat dilihat pada gambar 4.1 berikut ini :

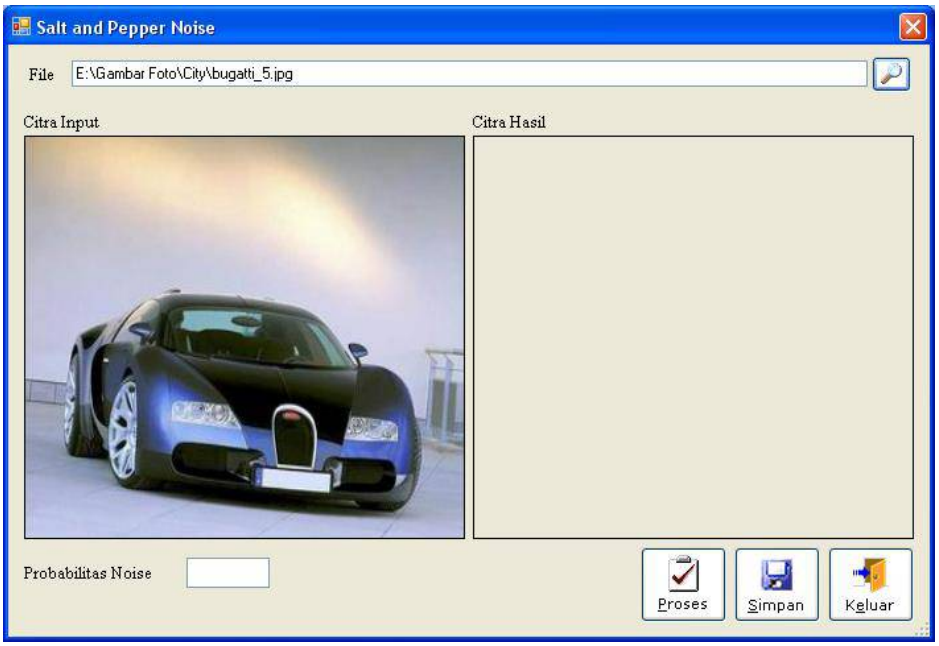
Gambar 4.1 *Form Main*

Untuk melakukan penambahan *noise*, maka dapat mengklik *link* Penambahan *Noise*, sehingga sistem akan menampilkan *form* Penambahan *Noise* seperti terlihat pada gambar berikut:



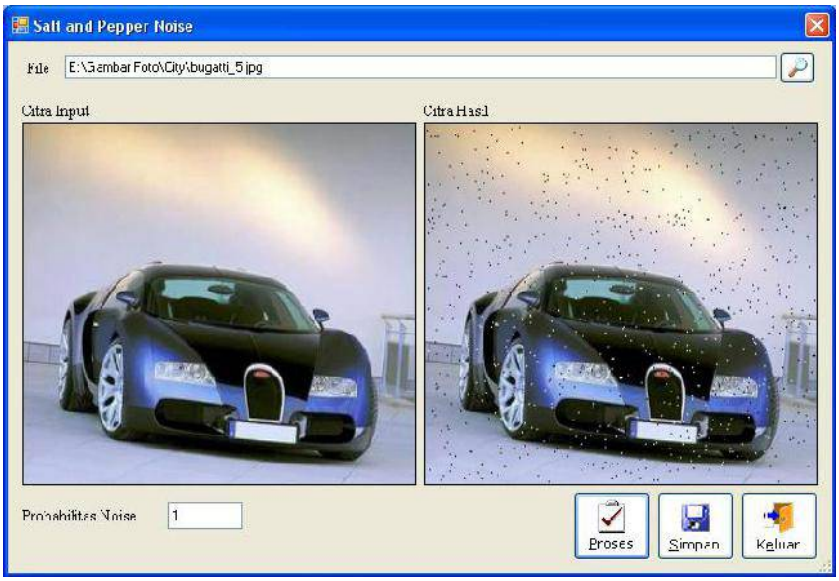
Gambar 4.2 *Form* Penambahan *Noise*

Untuk menambah *noise* pada citra, maka pemakai dapat mengisi persentase *noise* yang diinginkan, yaitu antara 1 sampai 50 %. Tampilan *form* setelah penambahan *noise* dapat dilihat pada gambar berikut:



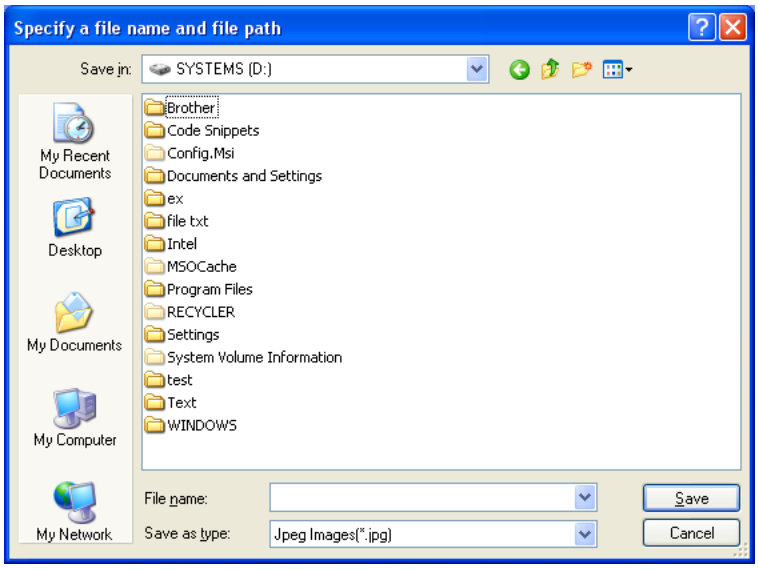
Gambar 4.3 *Form* Penambahan *Noise* Setelah Pengisian Data

Setelah itu, pemakai mengklik tombol Proses untuk menambahkan *noise* pada citra. Tampilan *form* setelah proses penambahan *noise* selesai dilakukan dapat dilihat pada gambar berikut:



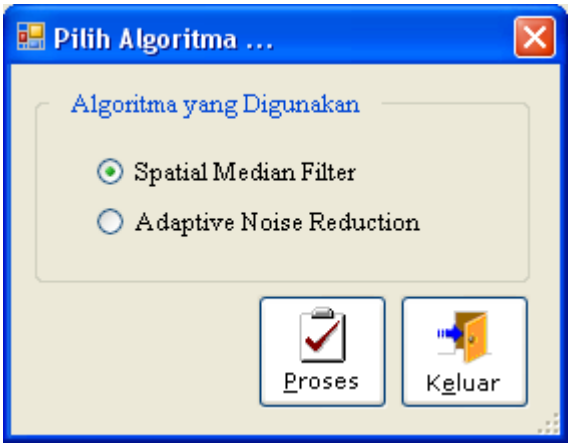
Gambar 4.4 *Form* Penambahan *Noise* Setelah Proses Selesai

Terakhir, untuk menyimpan gambar yang telah disisipkan *noise*, maka pemakai dapat mengklik tombol Simpan sehingga sistem akan menampilkan kotak dialog *Save*. Pemakai dapat memilih lokasi penyimpanan dan memasukkan nama *file* yang diinginkan. Tampilan kotak dialog *Save* dapat dilihat pada gambar berikut:



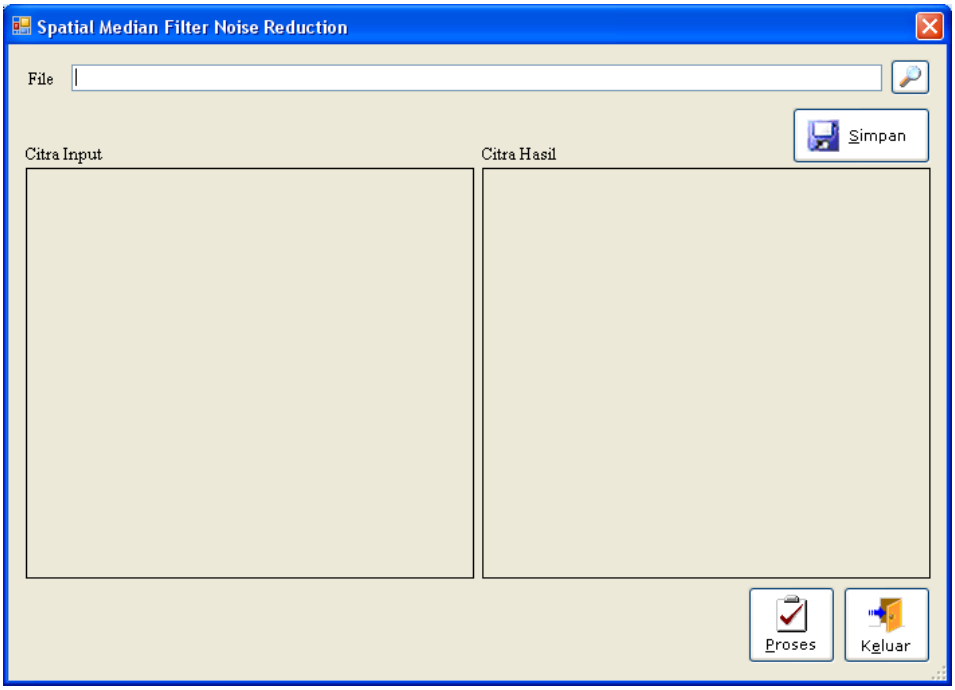
Gambar 4.5 Kotak Dialog *Save*

Untuk melakukan proses reduksi *noise*, maka pemakai dapat mengklik *link* Proses Reduksi *Noise*, sehingga aplikasi akan menampilkan *form* Pilih Algoritma seperti terlihat pada gambar berikut:



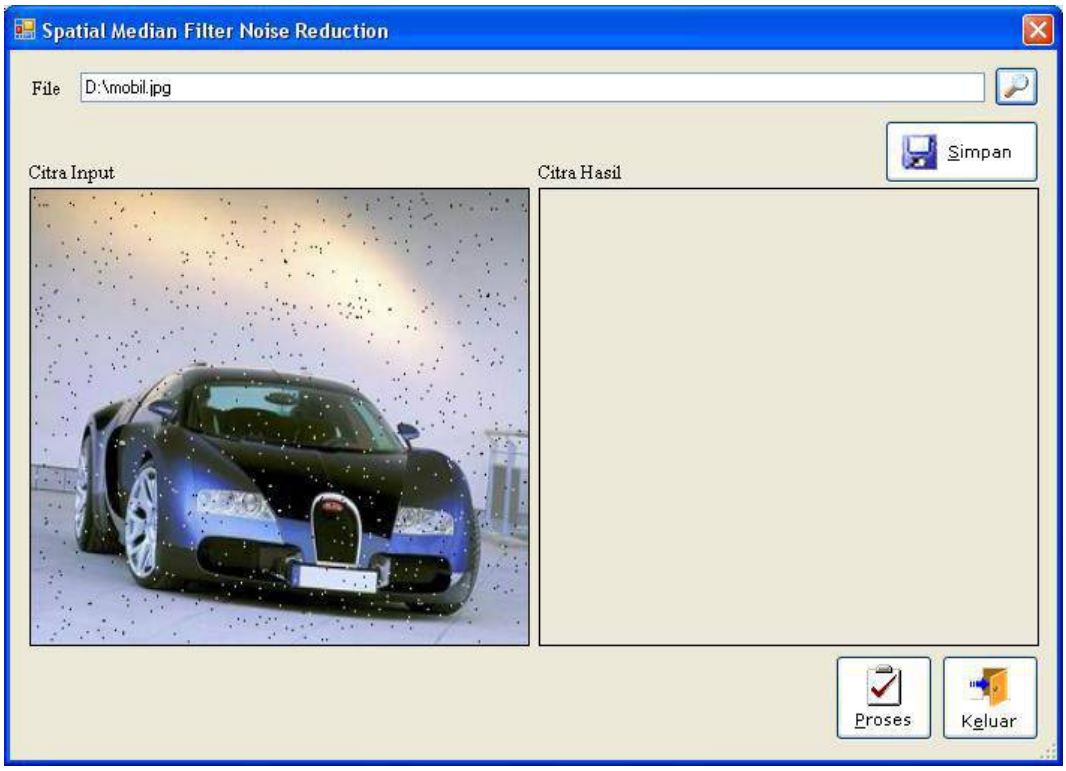
Gambar 4.6 *Form* Pilih Algoritma

Apabila pemakai ingin melakukan proses reduksi *noise* dengan menggunakan algoritma *Spatial Median Filter*, maka pemakai dapat mengklik *link Spatial Median Filter* dan mengklik tombol Proses sehingga aplikasi akan menampilkan *form Spatial Median Filter* seperti terlihat pada gambar berikut:



Gambar 4.7 *Form Spatial Median Filter Noise Reduction*

Pemakai dapat mengklik tombol Browse untuk memilih *file* citra yang diinginkan, sehingga tampilan aplikasi akan terlihat seperti pada gambar berikut:



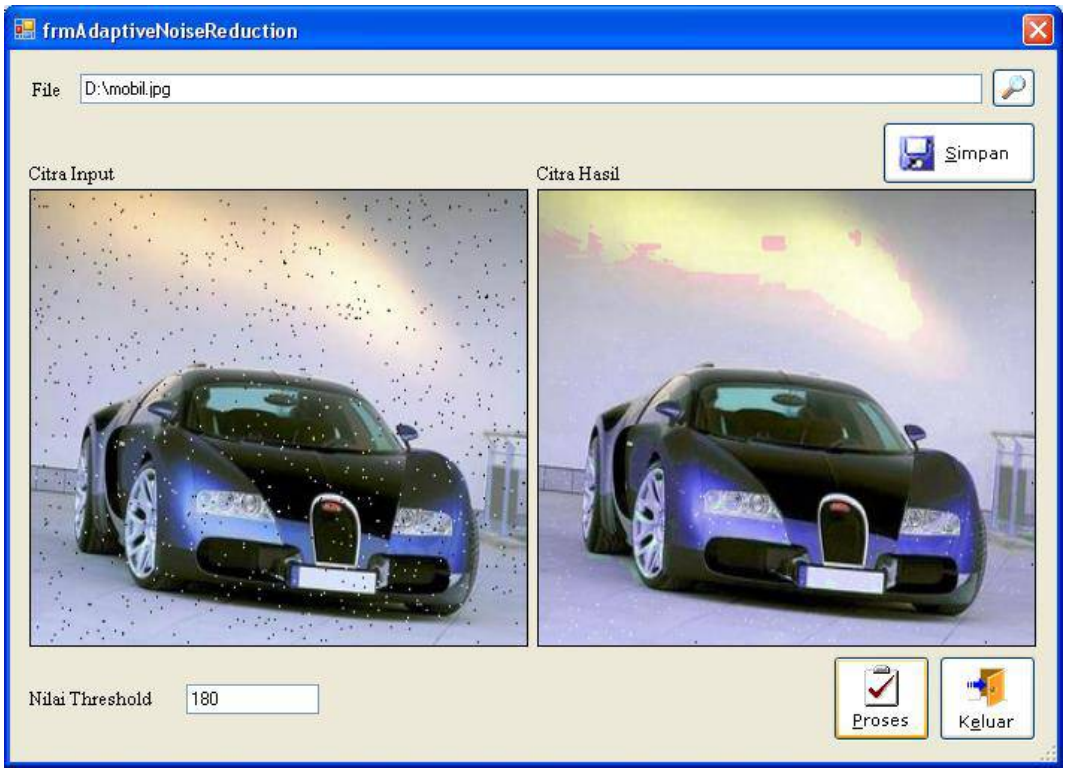
Gambar 4.8 *Form Spatial Median Filter Noise Reduction* Setelah Pengisian Data

Terakhir, pemakai mengklik tombol Proses sehingga aplikasi akan melakukan proses reduksi *noise* terhadap citra *input*.

Sementara itu, apabila pemakai ingin melakukan proses reduksi *noise* dengan menggunakan algoritma *Adaptive Noise Reduction*, maka pemakai dapat mengklik *link Adaptive Noise Reduction* dan mengklik tombol Proses sehingga aplikasi akan menampilkan *form Adaptive Noise Reduction* seperti terlihat pada gambar berikut:

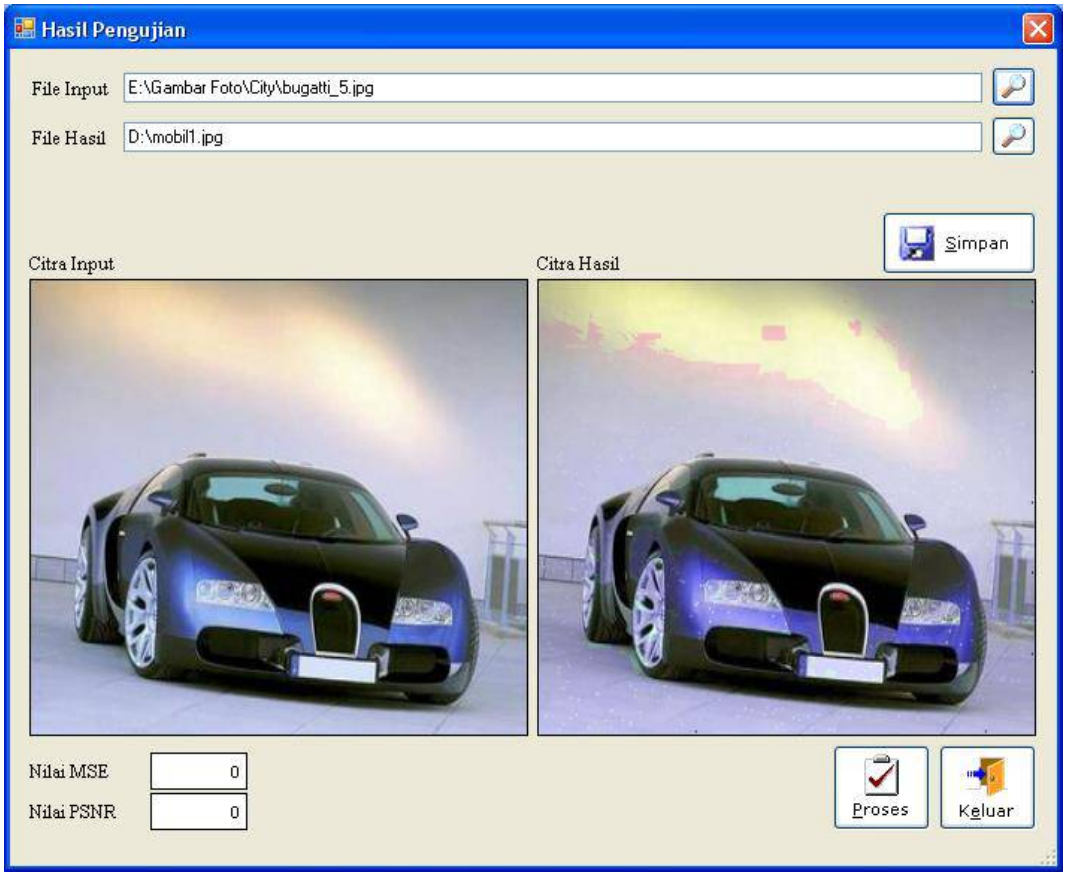
Gambar 4.10 *Form Adaptive Noise Reduction* Setelah Pengisian Data

Setelah itu, klik tombol ’Proses’, maka sistem akan menghasilkan citra hasil dari menggunakan metode *Adaptive Noise Reduction*. Berikut *form* yang dihasilkan ketika sistem berhasil melakukan proses reduksi *noise* terhadap citra:



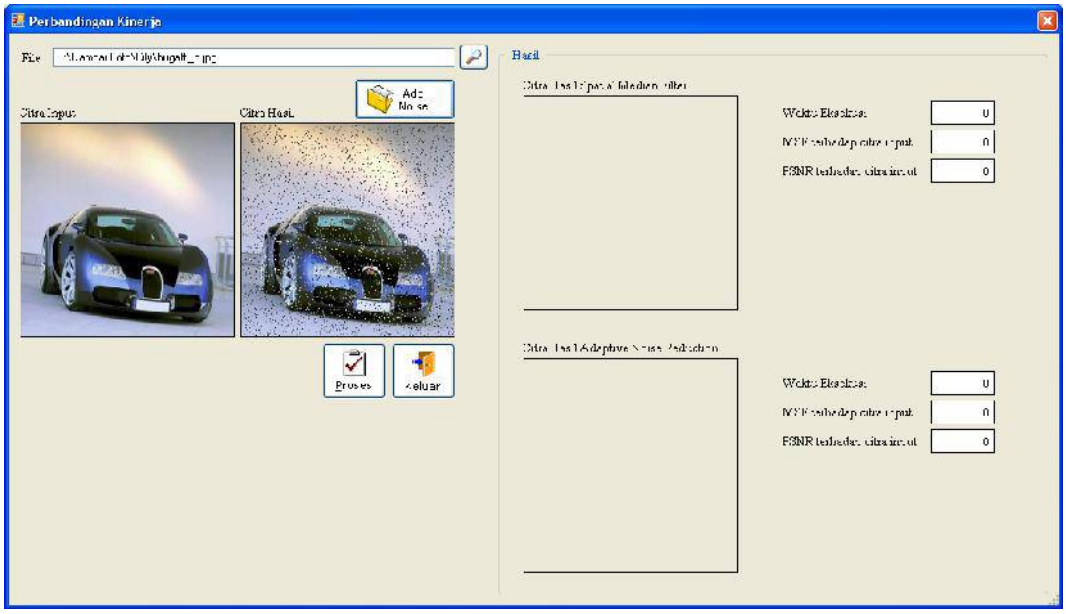
Gambar 4.11 *Form Adaptive Noise Reduction* Setelah Proses Selesai

Setelah melakukan proses reduksi *noise* pada citra, perlu dilakukan pengujian untuk mengecek apakah hasil dari proses reduksi *noise* dapat menghasilkan citra yang lebih baik dari citra input sebelumnya (*original*). Maka sistem yang dibuat perlu dilakukan proses pengujian sehingga penilain kualitas gambar tidak hanya secara visual tetapi memiliki sebuah metrik yang menjadi pengukurannya. Berikut adalah hasil pengujian dengan menggunakan citra sebelumnya.



Gambar 4.12 *Form* Hasil Pengujian Dengan Menggunakan Metode MSE dan PSNR

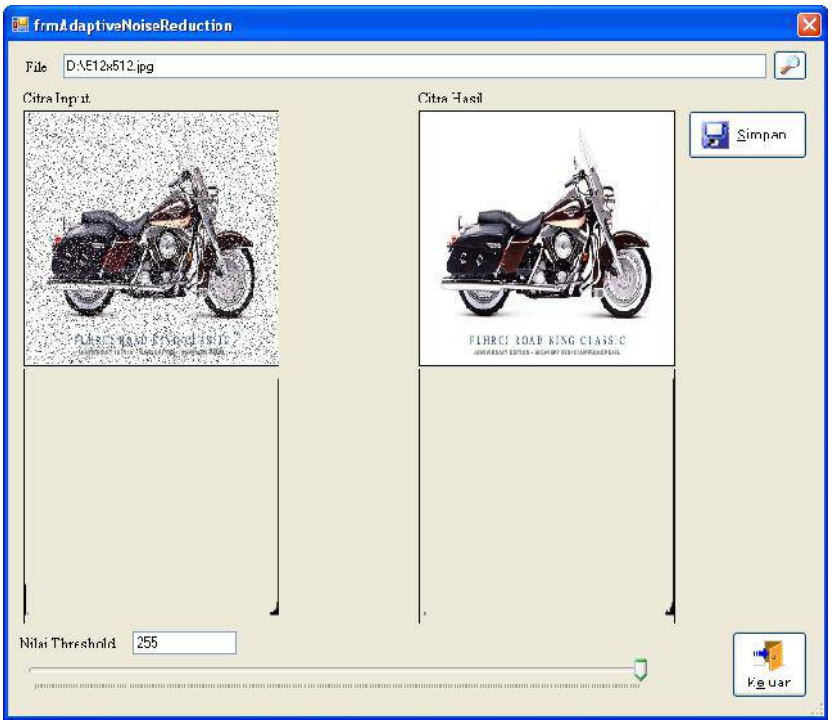
Form perbandingan dilakukan untuk membandingkan kinerja dari dua buah algoritma dalam melakukan proses reduksi *noise* terhadap citra *noise*. Berikut contoh form perbandingan citra *noise.*



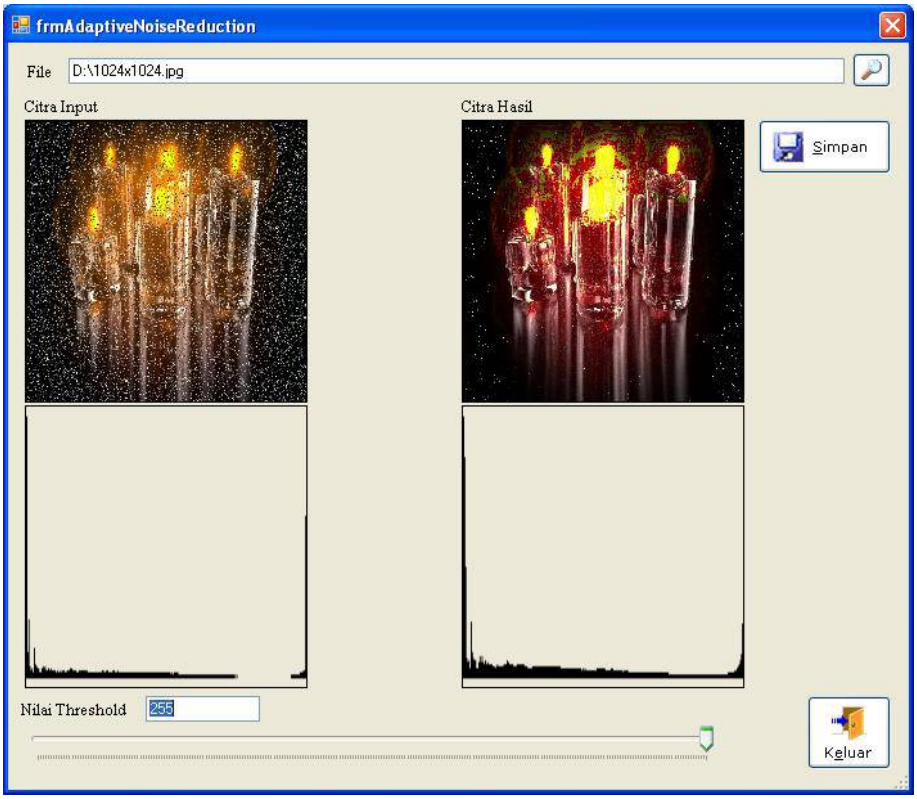
Gambar 4.13 *Form* Perbandingan

4.3 Pengujian

Proses pengujian untuk citra berukuran 512 x 512 piksel dan 1024 x 1024 piksel dapat dilihat pada gambar berikut ini:



Gambar 4.10 *Form* Hasil Pengujian untuk Citra 512 x 512 piksel



Gambar 4.11 *Form* Hasil Pengujian untuk Citra 1024 x 1024 piksel

Berdasarkan proses pengujian yang dilakukan pada perangkat lunak, maka dapat diperoleh beberapa informasi berikut :

1. Bagian perbandingan ini dapat digunakan untuk melakukan perbandingan kinerja antara algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction* dalam melakukan proses reduksi *noise*.
2. Bagian pengujian dapat digunakan untuk melakukan pengujian kualitas citra gambar hasil reduksi *noise* dari algoritma *Spatial Median Filter* dan *Adaptive Noise Reduction*.

Pada bagian aplikasi, berdasarkan proses pengujian yang dilakukan pada perangkat lunak terhadap beberapa gambar dengan probabilitas noise yang berbeda-beda, maka dapat diperoleh beberapa informasi berikut :

1. Dari hasil pengujian yang dilakukan, dapat diketahui bahwa pada proses reduksi *noise* dengan metode *Adaptive Noise Reduction*, nilai filter *input* yang lebih besar dari 150, kualitas citra hasil perbaikan tidak begitu berpengaruh.
2. Citra hasil reduksi *noise* dengan menggunakan metode *Adaptive Noise Reduction* memiliki kualitas yang lebih bagus. Hal ini terlihat dari nilai MSE dan PSNR yang diperoleh.

Untuk melakukan pengujian terhadap metode *adaptive noise reduction*, berikut diberikan beberapa data pengujian untuk 10 sampel citra:

File Citra	Ukuran Citra	Lama Waktu Eksekusi (ms)

	35 x 35	0
	65 x 65	31
	128 x 128	125
	208 x 320	594
	301 x 390	1062

	500 x 326	1407
	600 x 480	2562
	700 x 525	3094

	800 x 600	4109
	1024 x768	6797

BAB 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari pembahasan pada bab-bab sebelumnya, maka akhirnya penelitian pada skripsi ini dapat diambil beberapa kesimpulan, antara lain.

1. Metode *Adaptive Noise Reduction* akan menghasilkan citra hasil reduksi *noise* yang memiliki kualitas yang lebih bagus hal ini dapat dilihat melalui nilai MSE dan PSNR.
2. Citra yang memiliki *impulse noise* dapat diperbaiki dengan menggunakan metode *Adaptive Noise Reduction*.
3. Untuk metode *Adaptive Noise Reduction*, nilai filter threshold yang bagus minimal sebesar 150.
4. Citra yang memiliki *noise* yang diakibatkan adanya gangguan saat pengiriman citra maupun saat perbaikan citra dapat diredam dengan menggunakan metode *Adaptive Noise Reduction* dan *Spatial Median Filter*.
5. Ukuran citra *input* akan mempengaruhi lama waktu eksekusi dari perangkat lunak, dimana waktu eksekusi untuk citra 1024x1024 piksel akan lebih lama daripada citra 512x512 piksel.
6. Aplikasi mampu melakukan perbandingan kinerja dan pengujian kualitas citra gambar hasil reduksi *noise*.

5.2 Saran

Dalam pembuatan sistem ini tentu masih terdapat kekurangan. Oleh karena itu, ada beberapa saran yang dapat dijadikan bahan pertimbangan apabila ada yang berminat untuk mengembangkan tugas akhir ini.

1. Bagi peneliti lain yang ingin melakukan penelitian mengenai peningkatan kualitas citra, ada baiknya penelitian ini dikembangkan misalnya dengan melakukan perbandingan dengan metode-metode peningkatan kualitas citra yang lain, seperti mid point sehingga dapat diketahui hasil peningkatan kualitas citra dengan metode mana yang lebih baik kualitasnya.
2. Nilai *threshold* pada sistem ini masih ditentukan secara manual, ada baiknya sistem ini dapat secara otomatis menentukan nilai threshold yang cocok sehingga kualitas gambar yang dihasilkan lebih memuaskan.
3. Bagi peneliti lain yang ingin melakukan mengembangkan penelitian ini dapat melakukan penelitian terhadap gambar yang bergerak.

DAFTAR PUSTAKA

Destyningtias, B.,Heranurweni, S.,danNurhayati, T.Segmentasi Citra DenganMetodePengambangan.*JurnalElektrika*. Vol.2, No.1: 39 – 49. 2010.

Fachri, barany, agus perdana windarto, and ikhsan parinduri. "penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik." jepin (jurnal edukasi dan penelitian informatika) 5.2 (2019): 202-208.

Fachri, b., windarto, a. P., & parinduri, i. (2019). Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 5(2), 202-208.

Fachri, barany; windarto, agus perdana; parinduri, ikhsan. Penerapan backpropagation dan analisis sensitivitas pada prediksi indikator terpenting perusahaan listrik. Jepin (jurnal edukasi dan penelitian informatika), 2019, 5.2: 202-208.

Hamdi, nurul. "model penyiraman otomatis pada tanaman cabe rawit berbasis programmable logic control." jurnal ilmiah core it: community research information technology 7.2 (2019).

Jayaraman, et al.,*Discontinued Processing*. India : Tata McGraw Hill. 2009.

Munir, R. *Pengolahan Citra Digital*. Bandung : Informatika. 2011.

Noor, S. C. Mengubah Citra BerwarnaMenjadiGrayscale Dan Citra Biner. *Jurnal Teknologi Informasi Dinamik.* Vol.16 No.1.2011.

Nurul Fuad dan Yuliana Melita, Analisa Hasil Perbandingan Metode Low-Pass Filter Dengan Median Filter Untuk Optimalisasi Kualitas Citra Digital, Jurnal Teknika Vol.4 No.2 September 2012, ISSN No. 2085-0859.

Permana, aminuddin indra. "kombinasi algoritma kriptografi one time pad dengan generate random keys dan vigenere cipher dengan kunci em2b." (2019).

Putra, randi rian. "sistem informasi web pariwisata hutan mangrove di kelurahan belawan sicanang kecamatan medan belawan sebagai media promosi." jurnal ilmiah core it: community research information technology 7.2 (2019).

Putra, randi rian, et al. "decision support system in selecting additional employees using multi-factor evaluation process method." (2019).

Putra, randi rian. "implementasi metode backpropagation jaringan saraf tiruan dalam memprediksi pola pengunjung terhadap transaksi." jurti (jurnal teknologi informasi) 3.1 (2019): 16-20.

Putra, D. *Pengolahan Citra Digital*. Yogayakarta : Andi. 2010.

Saputra, muhammad juanda, and nurul hamdi. "rancang bangun aplikasi sejarah kebudayaan aceh berbasis android studi kasus dinas kebudayaan dan pariwisata aceh." journal of informatics and computer science 5.2 (2019): 147-157

Sidik, a. P., efendi, s., & suherman, s. (2019, june). Improving one-time pad algorithm on shamir's three-pass protocol scheme by using rsa and elgamal algorithms. In journal of physics: conference series (vol. 1235, no. 1, p. 012007). Iop publishing.

Sitepu, n. B., zarlis, m., efendi, s., & dhany, h. W. (2019, august). Analysis of decision tree and smooth support vector machine methods on data mining. In journal of physics: conference series (vol. 1255, no. 1, p. 012067). Iop publishing

Tasril, v., wijaya, r. F., & widya, r. (2019). Aplikasi pintar belajar bimbingan dan konseling untuk siswa sma berbasis macromedia flash. Jurnal informasi komputer logika, 1(3).

T. Sutoyo, et al., *Teori Pengolahan Citra Digital*.Yogyakarta : Andi Offset. 2009.

Ville, D. V. D.et al., Noise Reduction b Fuzzy Image Filtering. *IEEE TRANSACTIONS ON FUZZY SYSTEMS*. Vol. 11. No. 4. 2003.

Wang, Z. Et al. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSSING*. Vol. 13. No. 1. 2011.