TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Penerapan *Laplacian of Gaussian* Dalam Mendeteksi Tepi Luka Bakar Pada Manusia

Hasanuddin Gulo

Prodi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: hasan@gmail.com

Abstrak−Luka bakar adalah kerusakan jaringan yang disebabkan oleh kontak dengan suhu tinggi seperti api, air panas, listrik, bahan kimia, dan radiasi. Deteksi tepi citra menjadi bagian yang penting dalam radioterapi karena proses ini merupakan langkah awal pemisahan objek dalam citra yang memiliki perbedaan karakteristik dan ciri khas tersendiri. Batas antara satu objek dengan objek lain dalam citra yang berbeda karakteristik telah jelas, maka selanjutnya citra medis dapat dilakukan analisa citra lebih lanjut dan juga diagnosa kondisi penyakit pasien yang terlihat dengan menginterpretasikan citra medis tanpa harus melakukan pembedahan. Deteksi tepi dilakukan dengan menggunakan *Laplacian of Gaussian*, operator ini akan menangkap tepian dari semua arah dan menghasilkan tepian yang lebih tajam.

Kata Kunci: Laplacian of Gaussian, Luka Bakar

Abstract−Burns are tissue damage caused by contact with high temperatures such as fire, hot water, electricity, chemicals, and radiation. Image edge detection is an important part of radiotherapy because this process is the first step in separating objects in images that have different characteristics and characteristics. The boundary between one object and another object in an image with different characteristics is clear, then the medical image can be further analyzed and also the patient's disease condition diagnosis is seen by interpreting the medical image without having to perform surgery. Edge detection is performed using Laplacian of Gaussian, this operator will capture edges from all directions and produce sharper edges.

Keywords: Laplacian of Gaussian, Burns

1. PENDAHULUAN

Luka bakar adalah kerusakan jaringan yang disebabkan oleh kontak dengan suhu tinggi seperti api, air panas, listrik, bahan kimia, dan radiasi. Cedera lain yang termasuk luka bakar adalah sambaran petir, sengatan listrik, sinar X, dan bahan korosif. Kerusakan yang terjadi tergantung pada tinggi suhu dan lama kontak. Luka bakar memicu inflamasitidak terkontrol dan menekan system imun yang cenderung menyebabkan infeksi, sepsis dan kegagalan multi organ dengan tingkat kematian tinggi. Penelitian ini bertujuan untuk mengurangi rasa sakit pada luka bakar.

Citra (gambar) adalah suatu representasi, kemiripan, atau imitasi dari suatu objek . Citra mengandung informasi tentang objek yang dipresentasikan. Sehingga citra mampu memberikan informasi yang lebih banyak dibanding data teks. Untuk mempresentasikan objek lebih akurat dilakukan pengolahan citra. Pengolahan citra merupakan proses memanipulasi dan menganalisis citra menggunakan bantuan komputer yang bertujuan untuk memperbaiki, mengekstrak informasi dan menambah kualitas citra. Secara umum operasi pengolahan citra dapat diklarifikasikan sebagai perbaikan kualitas citra, restorasi citra, pemampatan citra, segmentasi citra, dan rekonstruksi citra. Salah satu pengolahan citra yang lebih spesifik adalah deteksi tepi.

Tepi (*edge*) adalah perubahan nilai insensitas abu yang mendadak (besar) dalam jarak yang singkat. Tapi biasanya terdapat pada batas antara dua daerah yang berbeda pada citra karena tepi mencirikan batas-batas objek di tepi dalam citra. Pendeteksian merupakan langkah pertama untuk melingkupi informasi didalam citra. Informasi yang diperoleh dapat berupa bentuk maupun ukuran objek. Tujuan pendeteksian tepi adalah untuk meningkatkan penampakan garis batas suatu daerah atau objek didalam citra.

Pada tepi yang curam, turunan keduanya mempunyai persilangan nol (*zero-crossing*), yaitu titik dimana terdapat pergantian tanda nilai keturunan kedua. Operator turunan kedua (*Laplace*) sangat sensitif terhadap derau (*noise*) sehingga pendeteksian tepi kurang akurat. Untuk mengatasi hal tersebut, deteksi tepi menggunakan turunan kedua (*Laplace*) dikombinasikan dengan fungsi *Gauss*. Fungsi *Gauss* bertujuan untuk melemahkan derau (*noise)* yang ada pada citra. Teknik deteksi tersebut dikenal dengan Metode *Laplace Gauss*. Metode LoG *(Laplacian of Gaussian)* adalah operator deteksi tepi orde kedua yang boleh dikata sukses untuk mengurangi tingkat sensitive terhadap derau. Hal ini disebabkan penggunaan fungsi *Gaussian* yang memuluskan citra dan berdampak pada pengurangan derau pada citra. Akibatnya, operator mereduksi jumlah tepi yang salah terdeteksi.

Penelitian, Nurhasanah, yang berjudul "Pendeteksian Tepi Citra CT Scan dengan menggunakan *Laplacian of Gaussianc(LOG)*, mengatakan bahwa metode LoG *(Laplacian of Gaussian)* adalah salah satu operator deteksi tepi yang dikembangkan dari turunan kedua. *Laplacian of Gaussian* terbentuk dari proses *Gaussian* yang diikuti operasi *laplace*. Fungsi *Gaussian* akan mengurangi derau sedangkan *Laplacian mask* meminimalisasi kemungkinan kesalahan deteksi tepi. Operator *Laplace* mendeteksi lokasi tepi lebih akurat khususnya pada tepi yang curam. Hal ini dikarenakan *zero-crossing* sendiri yang mendefenisikan lokasi tepi. Pada tepi yang curam, turunan keduanya mempunyai *zerocrossing*, yaitu titik dimana terdapat pergantian tanda nilai turunan kedua sedangkan pada tepi yang landai tidak terdapat *zero-crossing*. Tepi dari suatu objek pada image dimodelkan dengan menentukan spesifikasi dari unsure posisi, orientasi dan nilai intensitas yang konstan. Operator ini bekerja dengan mencari nilai nol pada turunan kedua dari citra, karena ketika turunan pertama terdapat pada nilai maksimum maka turunan kedua akan menghasilkan nilai nol [1].

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Penelitian Kustanto, yang berjudul "Computing Grasycale Of Face Detection menggunakan Metode *Sobel* dan *Laplacian Of Gaussian*, mengatakan bahwa metode LoG (*Laplacian of Gaussian)* merupakan metode yang menggunakan operator *Laplacian* Operator *Laplacian* adalah operator yang berbasis gradien yang menggunakan dua buah kernel yang berukuran 3x3 pixel. Operator ini mengambil arah diagonal untuk penentan arah dalam penghitungan nilai gradient [2].

Penelitian Annikmah Ritonga, mengatakan bahwa metode LoG (*Laplacian of Gaussian)* merupakan sebuah metode pendeteksian tepi yang menggunakan turunan keduanya untuk melakukan proses edge detection dan menghasilkan sebuah tampilan image yang berbeda dengan menampilkan efek relif. Operator ini sangat berbeda dari operator lainnya karena operator *Laplacian of Gaussian* berbentuk *omny directional* (tidak horizontal dan tidak vertikal). Operator ini akan menangkap tepian dari semua arah dan menghasilkan tepian yang lebih tajam. Operator ini mengambil arah diagonal untuk penentan arah dalam penghitungan nilai gradient [3].

Maka mendesain dan mengembangkan program deteksi tepi (*edge detection*) berdasarkan operator turunan kedua (*Laplacian*) untuk bagaimana agar obyek dari tepi penyakit *aterosklerosis* dapat disederhanakan bentuknya dari bentuk sebelumnya dengan perpaduan garis tepi serta dapat membantu mendeteksi tepi citra dari penyakit *aterosklerosis.* Oleh karena itu, penulis mencoba menerapkan metode *Laplacian of Gaussian* pada pendeteksian tepi citra penyakit *aterosklerosis* sebagai alternatif dalam penyelesaian masalah deteksi tepi citra penyakit *aterosklerosis.*

2. METODOLOGI PENELITIAN

2.1 Deteksi Tepi

Deteksi tepi (Edge Detection) pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari obyek-obyek citra, tujuannya adalah untuk menandai bagian yang menjadi detail citra dan untuk memperbaiki detail dari citra yang kabur, yang terjadi karena error atau adanya efek dari proses akuisisi citra. Suatu titik (x,y) dikatakan sebagai tepi (edge) dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangganya. Deteksi tepi berfungsi untuk memperoleh tepi objek. Deteksi tepi memanfaatkan perubahan nilai intensitas yang drastis pada batas dua area. Definisi tepi di sini adalah "himpunan piksel yang terhubung yang terletak pada batas dua area". Deteksi tepi dapat dibagi menjadi dua golongan. Golongan pertama disebut deteksi tepi orde pertama, yang bekerja dengan menggunakan turunan atau diferensial orde pertama. Termasuk kelompok ini adalah operator Roberts, Prewitt, dan Sobel. Golongan kedua dinamakan deteksi tepi orde kedua, yang menggunakan turunan orde kedua. Contoh yang termasuk kelompok ini adalah Laplacian of Gaussian (LoG). Berbagai teknik deteksi tepi bekerja dengan cara yang berbeda. Masing-masing memiliki kekuatan. Itulah sebabnya, eksperimen pada suatu aplikasi dengan menggunakan berbagai teknik deteksi tepi perlu dilakukan untuk mendapatkan hasil yang terbaik [4].

2.2 Citra

Citra adalah suatu representasi (gambaran), kemiripan, atau imitasi dari suatu objek. Citra sebagai keluaran suatu sistem perekaman data dapat bersifat optik berupa foto, bersifat analog berupa sinyal-sinyal video seperti gambar pada monitor televisi, atau bersifat digital yang dapat langsung disimpan pada suatu media penyimpan. Pengolahan citra adalah pemrosesan citra, khususnya menggunakan komputer, menjadi citra yang kualitasnya lebih baik dan sesuai dengan keinginan pemakai. Pengolahan citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra ke citra yang lain. Jadi masukannya adalah citra dan keluarannya juga citra, namun citra keluaran atau hasil mempunyai kualitas lebih baik dari pada citra masukan [3].

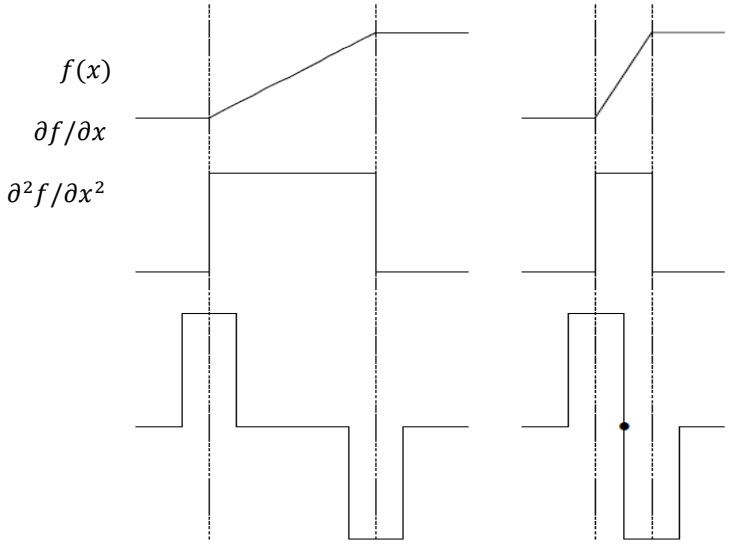
2.3 Metode *Laplacian of Gaussian*

Laplacian of Gaussian adalah salah satu operator deteksi tepi yang dikembangkan dari turunan kedua. *Laplacian of Gaussian* terbentuk dari proses *Gaussian* yang diikuti operasi *laplace*. Fungsi *Gaussian* akan mengurangi derau sedangkan *Laplacian mask* meminimalisasi kemungkinan kesalahan deteksi tepi. Operator Laplace mendeteksi lokasi tepi lebih akurat khususnya pada tepi yang curam. Hal ini dikarenakan *zero-crossing* sendiri yang mendefenisikan lokasi tepi. Pada tepi yang curam, turunan keduanya mempunyai zerocrossing, yaitu titik dimana terdapat pergantian tanda nilai turunan kedua sedangkan pada tepi yang landai tidak terdapat *zero-crossing*. Tepi dari suatu objek pada image dimodelkan dengan menentukan spesifikasi dari unsure posisi, orientasi dan nilai intensitas yang konstan. Operator ini bekerja dengan mencari nilai nol pada turunan kedua dari citra, karena ketika turunan pertama terdapat pada nilai maksimum maka turunan kedua akan menghasilkan nilai nol [1].

Laplacian of Gaussian (LoG) adalah operator deteksi tepi orde kedua yang boleh dikata sukses untuk mengurangi tingkat sensitive terhadap derau. Hal ini disebabkan penggunaan fungsi Gaussian yang memuluskan citra dan berdampak pada pengurangan derau pada citra (Crane, 1997). Operator turunan kedua disebut juga operator Laplace. Operator Laplace mendeteksi lokasi tepi lebih akurat khususnya pada tepi yang curam. Pada tepi yang curam, turunan keduanya mempunyai persilangan nol (*zero-crossing*), yaitu titik di mana terdapat pergantian tanda nilai turunan kedua (Gambar 2.6), sedangkan pada tepi yang landai tidak terdapat persilangan nol. Persilangan nol merupakan lokasi tepi yang akurat. Turunan kedua fungsi dengan dua pengubah adalah:

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2)$$



(a)Tepi Landai (b) Tepi Curam

Gambar 1. Deteksi tepi dengan operator turunan kedua
Sumber : (Darma Putra, 2010)

Dengan menggunakan definisi hampiran selisih-mundur (*backward difference approximation*):

$$G_3(x) = \frac{\partial f(x,y)}{\partial x} = \frac{f(x,y) - f(x-\Delta x,y)}{\Delta x} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

$$G_3(y) = \frac{\partial f(x,y)}{\partial x} = \frac{f(x,y) - f(x,y-\Delta y)}{\Delta y} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4)$$

Maka :

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

$$G_1\big(G_3(x)\big) = G_1\big(G_3(y)\big)$$
$$= \frac{1}{\Delta x} G_1\big(f(x,y)\big) - G_1\big(f(x-\Delta x,y)\big) + \frac{1}{\Delta y} G_1\big(f(x,y)\big) - G_1\big(f(x,y-\Delta y)\big)$$
$$= \frac{1}{\Delta x}\left\{\frac{f(x+\Delta x,y) - f(x,y) - f(x,y) + f(x-\Delta x,y)}{\Delta x}\right\}$$
$$+ \frac{1}{\Delta y}\left\{\frac{f(x,y+\Delta y) - f(x,y) - f(x,y) + f(x,y-\Delta y)}{\Delta y}\right\}$$
$$= \frac{f(x+\Delta x,y) - 2f(x,y) f(x-\Delta x,y)}{(\Delta x^2)}$$
$$+ \frac{f(x,y+\Delta y) - 2f(x,y) f(x,y-\Delta y)}{(\Delta y^2)}$$

Dengan mengasumsikan $\Delta x = \Delta x = 1$, maka diperoleh :

$$\Delta^2 f(x,y) = f(x+1,y) - 2f(x,y) + f(x-1,y) + f(x,y+1) - 2f(x,y) + f(x,y-1)$$
$$= f(x,y-1) + f(x-1,y) - 4f(x,y) + f(x+1,y) + f(x,y+1)$$

Atau dapat dinyatakan sebagai *mask* :

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Selain *mask* di atas, masih ada dua hampiran operator Laplace yang lain, yaitu

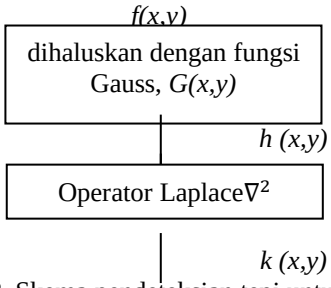
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \text{ dan } \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Kadang-kadang diinginkan memberi bobot yang lebih pada *pixel* tengah di antara *pixel* tetangganya. Operator Laplace yang digunakan untuk tujuan ini adalah

$$\begin{bmatrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{bmatrix}$$

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Operator Laplace termasuk ke dalam penapis lolos-tinggi sebab jumlah seluruh koefisiennya nol dan koefisiennya mengandung nilai negatif maupun positif. Kadangkala pendeteksian tepi dengan operator Laplace menghasilkan tepi-tepi palsu yang disebabkan oleh gangguan. Untuk mengurangi kemunculan tepi palsu, citra disaring dulu dengan fungsi Gaussian (Gambar 2).



Gambar 2. Skema pendeteksian tepi untuk citra yang mengalami gangguan.
Sumber : (Darma Putra, 2010)

Berdasarkan skema pada Gambar 2:

$k\ (x,y) = \nabla^2 h(k,y)$ .....(6)

Dan $h\ (x,y) = f(x,y) * G(x,y)$ .....(7)

Maka dapat dibuktikan bahwa

$\nabla^2[f(x,y) * G(x,y)] = f(x,y) * \nabla^2 G(x,y)$ .....(8)

Jadi $k(x,y) = f(x,y) * \nabla^2 G(x,y)$ .....(9)

yang dalam hal ini $\nabla^2 G(x,y) = \left(\frac{x^2+y^2-2\sigma^2}{\sigma^4}\right) e^{-\frac{(x^2+y^2)}{2\sigma^2}}$ .....(10)

Transformasi Laplace adalah proses mengubah fungsi F(t) dari fungsi waktu ke fungsi kompleks f(s) dari operasi kompleks S. Transformasi Laplace dari suatu fungsi F(t) didefinisikan sebagai:

$L\{F(t)\} = f(s) = \int_0^\infty e^{-st}.\ F(t)\ .\ dt$ .....(11)

Tabel 1. Transformasi *Laplace* untuk beberapa fungsi dasar

1	$\frac{1}{s}$	s > 0
t^n ; (m = 1, 2, 3, …	$\frac{n!}{S^{n+1}}$	s > 0
t^p ; (p > -1)	$\frac{\gamma_{(p+1)}}{S^{p+1}}$	s > 0
e^{at}	$\frac{1}{s-a}$	s > a
$Cos\omega t$	$\frac{s}{s^2+\omega^2}$	s > 0
$Sin\omega t$	$\frac{s}{s^2+\omega^2}$	s > 0
Cosh at	$\frac{s}{s^2-a^2}$	s > |a|
Sinh at	$\frac{a}{s^2+a^2}$	s > |a|

Buktikan L {1} = $\frac{1}{s}$!

Bukti :

$$L\{F(t)\} = \int_0^\infty e^{-st}.F(t).dt$$

$$L[1] = \int_0^\infty e^{-st}(1)dt = \lim_{\mu\to\infty}\int_0^\mu e^{-st}\,dt$$

$$= \lim_{\mu\to\infty}\int_0^\mu e^{-st}\,d(e^{-st}) = \lim_{\mu\to\infty}\left(-\frac{1}{s}e^{-st}\right)\Big|_0^\mu$$

$$= -\frac{1}{s}\lim_{\mu\to\infty}(e^{-\mu s} - e^0) = -\frac{1}{s}(0-1) = \frac{1}{s}$$

Buktikan : L {t} = $\frac{1}{s}$!

Bukti : $L[F(t)] = \int_0^\infty e^{-st}\,F(t)dt$

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

$$L[t] = \int_0^\infty e^{-st}(1)dt = \lim_{\mu \to \infty} \int_0^\mu t.e^{-st}\,dt$$
$$= -\frac{1}{s}\lim_{\mu \to \infty} \int_0^\mu t.d(e^{-st}) = -\frac{1}{s}\lim_{\mu \to \infty}\left[t.e^{-st}\Big|_0^\mu - \int_0^\mu e^{-st}.d(t)\right]$$
$$= -\frac{1}{s}\lim_{\mu \to \infty}\left[t/.e^{-st}\Big|_0^\mu\right] + \frac{1}{s}\lim_{\mu \to \infty}\int_0^\mu e^{-st}.(1)dt$$
$$= -\frac{1}{s}\lim_{\mu \to \infty}\left[1/s.e^{-st}\Big|_0^\mu\right] + \frac{1}{s}\lim_{\mu \to \infty}\int_0^\mu e^{-st}.(1)dt$$
$$= -\frac{1}{s}(0) + \frac{1}{s}\int_0^\infty e^{-st}(1)dt = 0 + \frac{1}{s}.L[1] = \frac{1}{s}.\frac{1}{s} = \frac{1}{s^2}$$

Buktikan : L $\{t^2\} = \frac{2}{s^3}$!
Bukti :

$$L\{F(t)\} = \int_0^\infty e^{-st}.F(t).dt$$
$$L\{t^2\} = \int_0^\infty e^{-st}.t^2.dt$$
$$= \lim_{\rho \to \infty}\int_0^\rho t^2.e^{-st}dt$$
$$= \lim_{\rho \to \infty} -\frac{1}{s}\int_0^\rho t^2.e^{-st}d(-st)$$
$$= -\frac{1}{s}\lim_{\rho \to \infty}\int_0^\rho t^2\,d\,(e^{-st})$$
$$= -\frac{1}{s}\lim_{\rho \to \infty}\left[t^2\,e^{-st}\Big|_o^\rho - \int_0^\rho e^{-st}\,d(t^2)\right]$$
$$= -\frac{1}{s}\left[\lim_{\rho \to \infty}\left\{\frac{t^2}{e^{st}}\Big|_0^\rho\right\} - \lim_{\rho \to \infty}\int_0^\rho e^{-st}(2t)dt\right]$$
$$= -\frac{1}{s}\left[\lim_{\rho \to \infty}\frac{\rho^2}{e^{s\rho}} - \lim_{\rho \to \infty}\frac{0^2}{e^{s0}} - 2\int_0^\infty e^{-st}.t.dt\right]$$
$$= -\frac{1}{s}[0 - 0 - 2\,L\{t\}]$$
$$= -\frac{1}{s}\left[-2.\frac{1}{s^2}\right]$$
$$L\{t^2\} = \frac{2}{s^3}$$

Buktikan : L $\{t^n\} = \frac{n!}{s^{n+1}}$!
Bukti :

L $\{1\}$ = L $\{t^n\} = \frac{1}{3} = \frac{0!}{s^{0+1}}$
L $\{1\} = \frac{1}{s^2} = \frac{1!}{s^{1+1}}$
L $\{t^2\} = \frac{2}{s^3} = \frac{2.1}{s^{2+1}} = \frac{2!}{s^{2+1}}$
L $\{t^2\} = \frac{n!}{s^{n+1}}$

Bukti L $[t^n] = \frac{\gamma(n+1)}{s^{n+1}}$
Bukti :
Fungsi Gamma : $\gamma_{(n)} = \int_0^\infty e^{-x}\,x^{n-1}dx$
$\gamma(n+1) = \int_0^\infty e^{-x}\,x^{n+1-1}dx$
$\gamma(n+1) = \int_0^\infty e^{-x}\,x^n\,dx$ misalkan : x = st
dx = s.dt
$\gamma(n+1) = \int_0^\infty e^{-st}\,(st)^n\,s.dt$
$\gamma(n+1) = \int_0^\infty e^{-st}\,s^n\,t^n\,s.dt$
$\gamma(n+1) = s^{n+1}\int_0^\infty e^{-st}\,t^n\,dt$
$\gamma(n+1) = s^{n+1}.L[t^n]$
$L[t^n] = \frac{\gamma(n+1)}{s^{n+1}}$
maka $L[t^n] = \frac{n!}{s^{n+1}} = \frac{\gamma(n+1)}{s^{n+1}}$ → [terbukti]

Buktikan : L $\{e^{at}\} = \frac{1}{s-a}$
Bukti :

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

$$L\{F(t)\} = \int_0^{\infty} e^{-st}.F(t).dt$$

$$L[e^{at}] = \int_0^{\infty} e^{-st}(e^{at})\,dt = \lim_{\mu \to \infty} \int_0^{\mu} e^{-st+at}\,dt$$

$$= \lim_{\mu \to \infty} \int_0^{\mu} e^{-(s-a)t}\,dt = \frac{1}{-(s-a)} \lim_{\mu \to \infty} \int_0^{\mu} e^{-(s-a)t} \mathrm{d}[-\mathrm{s}(\mathrm{t}-\mathrm{a})]$$

$$= \frac{1}{-(s-a)} \lim_{\mu \to \infty} \left[e^{-(s-a)t}|_0^{\mu}\right] = \frac{1}{-(s-a)} \lim_{\mu \to \infty} \frac{1}{e^{-(s-a)t}}\Big|_0^{\mu}$$

$$= \frac{1}{-(s-a)} \left[\lim_{\mu \to \infty} \frac{1}{e^{(s-a)\mu}} - \lim_{\mu \to \infty} \frac{1}{e^{(s-a)0}}\right]$$

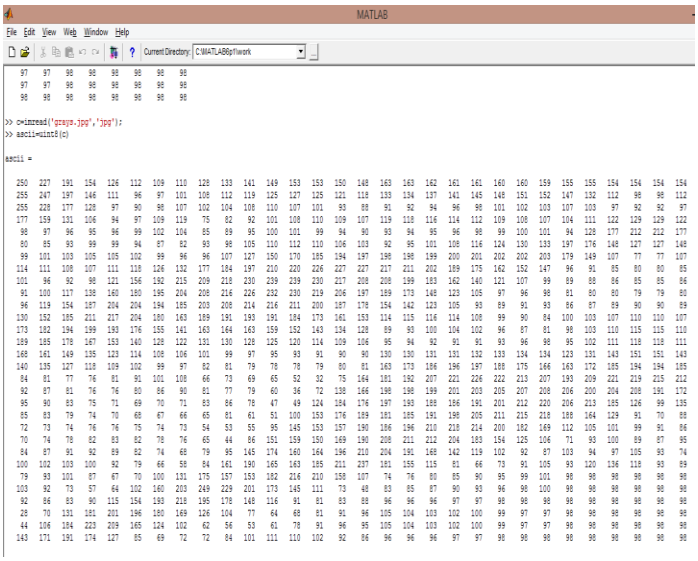
$$= \frac{1}{-(s-a)} \left[0 - \lim_{\mu \to \infty} \frac{1}{e^{0}}\right] = \frac{1}{-(s-a)}.[0-1]$$

$$L\{e^{at}\} = \frac{1}{(s-a)} \rightarrow [\text{terbukti}]$$

3. HASIL DAN PEMBAHASAN

Citra *input* merupakan citra yang memiliki intensitas warna berkisar antara 0 sebagai nilai minimum sampai 255 yang merupakan nilai maksimum. Citra input yang memiliki ukuran 30x30*pixel* kemudian dikonversi ke dalam bentuk matriks 30x30 = 900, untuk masing-masing citra.

Contoh nilai dari citra*grayscale* dengan *image size* 30x30 *pixel* yang dikonversi kedalam bentuk matriks sebagai berikut:



Gambar 3. Matriks citra

3.1 Penerapan Metode *Laplacian of Gaussian*

Filter Gaussian tergolong sebagai *filter* lolos rendah yang didasarkan pada fungsi *gaussian*. Model dua dimensinya berupa:

$$G(x\ y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Dimana *x* dan *y* adalah posisi kordinat pada sumbu *x* dan *y*. Persamaan inilah yang dipakaisebagai dasar untuk menentukan nilai-nilai setiapelemen dalam *filter gaussian* yang akan dibuat. Misalkan dibuat kernel *filter* ukuran 5x5 dan mengisi elemen/bobot *g*(*x*,*y*) pada matriks kernel *gaussian*.

Keterangan:
e (*euler*) = 2,718281828 (konstanta)
σ (*Standart deviasi*) = 2
Maka :

$$G(0,0) = e^{-\frac{0^2+0^2}{2.2^2}}$$

$$= e^{-0}$$

$$= 1$$

$G1,0)=G(0,1)=G(-1,0)=G(0,-1)=e^{(-1/4)} = 0,7788$

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

$G(1,1)=G(-1,1)=G(1,-1)=G(-1,-1)=e^{(-2/4)}=0{,}6065$
$G(1,2)=G(2,1)=G(-1,-2)=G(-2,-1)=e^{(-5/4)}=\ 0{,}2865$
$G(2,0)=G(0,2)=G(-2,0)=G(0,-2)=e^{(-4/4)}=0{,}3678$
$G(2,2)=G(2,-2)=G(-2,2)=G(-2,-2)=e^{(-8/4)}=0{,}1353$

Maka diperoleh nilai elemen atau bobot matrik kernel *gaussian* sebagai berikut:

Tabel 2. Elemen atau Bobot Matrik Kernel

x/y	-2	-1	0	1	2
-2	0,1353	0,2865	0,3678	0,2865	0,1353
-1	0,2865	0,6065	0,7788	0,6065	0,2865
0	0,3678	0,7788	1	0,7788	0,3678
1	0,2865	0,6065	0,7788	0,6065	0,2865
2	0,1353	0,2865	0,3678	0,2865	0,1353

Selanjutnya normalisasi nilai pembobotan setiap bobot dengan nilai terkecil dari nilai bobot. Nilai terkecil 0,1353dan kemudian hasilnya dibulatkan ke atas). Dengan demikian, diperoleh hasil seperti berikut:

Tabel 3. Normalisasi Nilai Pembobotan

Nilai bobot	Pembagian bobot
1	1/0,1353 = 7
0,7788	0,7788/0,1353 = 6
0,6065	0,6065/0,1353 = 4
0,3678	0,3678/0,1353 = 3
0,2865	0,2865/0,1353 = 2
0,1353	0,1353/0,1353 = 1

Maka diperoleh matriks sebagai berikut:

Tabel 4. Hasil Elemen atau bobot matrik

x/y	-2	-1	0	1	2
-2	1	2	3	2	1
-1	2	4	6	4	2
0	3	6	7	6	3
1	2	4	6	4	2
2	1	2	3	2	1

Jumlah semua elemen nilai pembobot pada *filter* agar selang nilai intensitas tetap seperti semula. Berdasarkan matriks pada tabel 4 jumlah semua elemen nilai pembobot pada *filter* = 79. Berikut adalah *filter gaussian* hasil rancangannya.

$$g(x,y)\frac{1}{79}$$

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

Untuk melakukan operasi perbaikan *noise* citra dengan metode *filter gaussian*. Operasi ini dilakukan dengan cara konvolusi, konvolusi seringkali dilibatkan dalam operasi ketetanggan *pixel*. Konvolusi pada citra sering disebut konvolusi 2 dimensi. Konvolusi 2 dimensi didefinisikan sebagai proses untuk memperoleh suatu *pixel* berdasarkan nilai *pixel* itu sendiri dan tetangganya, dengan melibatkan suatu matriks yaitu kernel yang mempresentasikan pembobotan. Penjelasan rumus yang digunakan dalam konvolusi filter Gaussian adalah sebagai berikut:

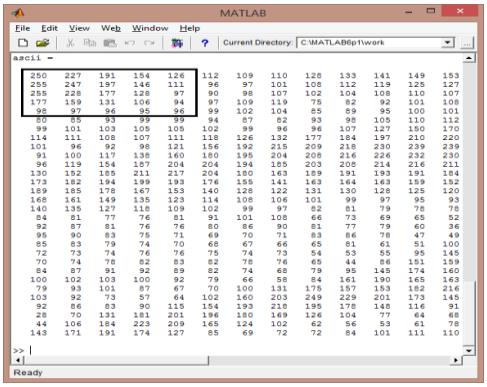
TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

$$h(x,y) = f(x,y) * g(x,y) = \sum_{k=1}^{M} \sum_{l=1}^{N} f(k,l.g(x-k,y-l)$$

Keterangan:
h(x,y): gambar output
f(x,y) : adalah gambar input
g(x,y) : adalah filter gaussian

Jadi secara umum rumus diatas adalah jumlah dari perkalian antara pixel citra dengan filter gaussian dan hasilnya dibagi dengan jumlahdari matriks filter agar selang nilai intensitas tetap seperti semula. Untuk penjelasan proses konvolusi penulis membuat sebuah perumpamaan matriks citra*grayscale*yang terdapatpadagambar 3.3, dengan resolusi matriks 30x30*pixel* yang akan dikonvolusikan dengan filter gaussian dengan ukuran matriks 5x5. Langkah selanjutnya adalah melakukan operasi konvolusi dengan cara menempatkan atau menumpangkansuatufilter atau kernelpada setiap *pixel*yang ditimpali, kemudian nilai rerata diambil dari hasil-hasil tersebut. Pada proses pelaksanaan konvolusi kernel digeser sepanjang baris dankolomdalamcitra sehinggadiperolehnilaibarupadacitrakeluaran. Langkah-langkah konvolusi digambarkan sebagai berikut:

Langkah 1: Tempatkan *kernel* pada sudut kiri atas, kemudian hitung nilai *pixel* pada posisi (0,0) dari *kernel*



Gambar 4. Kernel Pixel

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x250) + (2x227) + (3x191) +(2x154) +(1x126) + (2x255) +(4x247) +(6x197) +(4x146) +(2x111) + (3x255) +(6x228) +(7x177) +(6x128) +(3x97) + (2x177) +(4x159) +(6x131) +(4x106) +(2x94) + (1x98) + (2x97) +(3x96) +(2x95) +(1x96) = (250 + 454 + 573 + 308 + 126 + 510 + 988 + 1182 + 584 + 222 + 765 + 1368 + 1239 + 768 + 291 + 354 + 636 + 786 + 424 + 188 + 98 + 194 + 288 + 190 +96) = 12882 : 79 = 163.063

Nilai *pixel* yang diubah adalah 177 menjadi 163

Langkah 2: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (1.0) dari *kernel*.

227	191	154	126	112
247	197	146	111	96
228	177	128	97	90
159	131	106	94	97
97	96	95	96	99

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x227) + (2x191) + (3x154) + (2x126) + (1x112) + (2x247) + (4x197) + (6x146) + (4x111) + (2x96) + (3x228) + (6x177) + (7x128) + (6x97) + (3x90) + (2x159) + (4x131) + (6x106) + (4x94) + (2x97) + (1x97) + (2x96) + (3x95) + (2x96) + (1x99) = (454 + 382 + 462 + 252 + 112 + 494 + 588 + 876 + 444 + 192 + 684 + 702 + 896 + 582 + 270 + 318 + 524 + 636 + 376 + 194 + 97 + 192 + 285 + 192 + 99) = 10303 : 79 = 130.417

Nilai *pixel* yang diubah adalah 128 menjadi 130.

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

Langkah 3: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (2,0) dari *kernel*.

191	154	126	122	109
197	146	111	96	97
177	128	97	90	98
131	106	94	97	109
96	95	96	99	102

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x191) + (2x154) + (3x126) +(2x112) + (1x109) + (2x197) + (4x146) + (6x111) +(4x96) +(2x97) + (3x177) + (6x128) + (7x97) + (6x90) + (3x98) + (2x131) + (4x106) + (6x94) + (4x97) + (2x109) + (1x96) + (2x95) + (3x96) + (2x99) + (1x102) = (191 + 308 + 378 + 224 + 109 + 394 + 584 + 666 + 384 + 194 + 531 + 768 + 679 + 540 + 294 + 262 + 424 + 564 + 388 + 218 + 96 + 190 + 288 +198 + 102) = 8974 : 79 = 113.594
Nilai *pixel* yang diubah adalah 97 menjadi 113.

Langkah 4: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (3,0) dari *kernel*.

154	126	112	109	110
146	111	96	97	101
128	97	90	98	107
106	94	97	109	119
95	96	99	102	104

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x154) + (2x126) + (3x112) + (2x109) + (1x110) + (2x146) +(4x111) + (6x96) + (4x97) +(2x101) + (3x128) + (6x97) + (7x90) + (6x98) + (3x107) + (2x106) + (4x94) + (6x97) + (4x109) + (2x119) + (1x95) + (2x96) + (3x99) + (2x102) + (1x104) = (154 + 252 + 336 + 218 + 110 + 292 + 444 + 576 + 388 + 202 + 384 + 582 + 630 + 588 + 321 + 212 + 376 + 582 + 436 + 238 + 95 + 192 + 297 + 204 + 104) = 8213 : 79 = 103.962
Nilai *pixel* yang diubah adalah 90 menjadi 104

Langkah 5: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (4,0) dari *kernel*.

126	112	109	110	128
111	96	97	101	108
97	90	98	107	102
94	97	109	119	75
96	99	102	104	85

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x126) + (2x112) + (3x109) + (2x110) + (1x128) + (2x111) +(4x96) + (6x97) + (4x101) +(2x108) + (3x97) + (6x90) + (7x98) + (6x107) + (3x102) + (2x94) + (4x97) + (6x109) + (4x119) + (2x75) + (1x96) + (2x99) + (3x102) + (2x104) + (1x85) = (126 + 224 + 327 + 220 + 128 + 222 + 384 + 582 + 404 + 216 + 291 + 540 + 686 + 642 + 306 + 188 + 388 + 654 + 476 + 150 + 96 + 98 + 306 + 208 + 85) = 8047 : 79 = 101.860
Nilai *pixel* yang diubah adalah 98 menjadi 102

Langkah 6: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (5,0) dari *kernel*.

112	109	110	128	133
96	97	101	108	112
90	98	107	102	104
97	109	119	75	82
99	102	104	85	89

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x112) + (2x109) + (3x110) + (2x128) + (1x133) + (2x96) +(4x97) + (6x101) + (4x108) +(2x112) + (3x90) + (6x98) + (7x107) + (6x102) + (3x104) + (2x97) + (4x109) + (6x119) + (4x75) + (2x82) + (1x99) + (2x102) + (3x104) + (2x85) + (1x89) = (112 + 218 + 330 + 256 + 133 + 192 + 388 + 606 + 432 + 224 + 270 + 588 + 749 + 612 + 312 + 194 + 436 + 714 + 300 + 164 + 99 + 204 + 312 + 170 + 89) = 8104 : 79 = 102.582
Nilai *pixel* yang diubah adalah 107 menjadi 102

Langkah 7: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (6,0) dari *kernel*.

1	2	3	2	1

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

109	110	128	133	141
97	101	108	112	119
98	107	102	104	108
109	119	75	82	92
102	104	85	89	95

→

2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x109) + (2x110) + (3x128) + (2x133) + (1x141) + (2x97) +(4x101) + (6x108) + (4x112) +(2x119) + (3x98) + (6x107) + (7x102) + (6x104) + (3x108) + (2x109) + (4x119) + (6x75) + (4x82) + (2x92) + (1x102) + (2x104) + (3x85) + (2x89) + (1x95) = (109 + 220 + 384 + 266 + 141 + 194 + 404 + 648 + 448 + 238 + 294 + 642 + 714 + 624 + 324 + 218 + 476 + 450 + 328 + 184 + 102 + 208 + 255 + 178 + 95) = 8144 : 79 = 103.088
Nilai *pixel* yang diubah adalah 102 menjadi 103

Langkah 8: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (7,0) dari *kernel*.

110	128	133	141	149
101	108	112	119	125
107	102	104	108	110
119	75	82	92	101
104	85	89	95	100

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x110) + (2x128) + (3x133) + (2x141) + (1x149) + (2x101) +(4x108) + (6x112) + (4x119) +(2x125) + (3x107) + (6x102) + (7x104) + (6x108) + (3x110) + (2x119) + (4x75) + (6x82) + (4x92) + (2x101) + (1x104) + (2x85) + (3x89) + (2x95) + (1x100) = (110 + 256 + 399 + 282 + 149 + 202 + 432 + 672 + 476 + 250 + 321 + 612 + 728 + 648 + 330 + 238 + 300 + 492 + 368 + 202 + 104 + 170 + 267 + 190 + 100) = 8298 : 79 = 105.037
Nilai *pixel* yang diubah adalah 104 menjadi 105

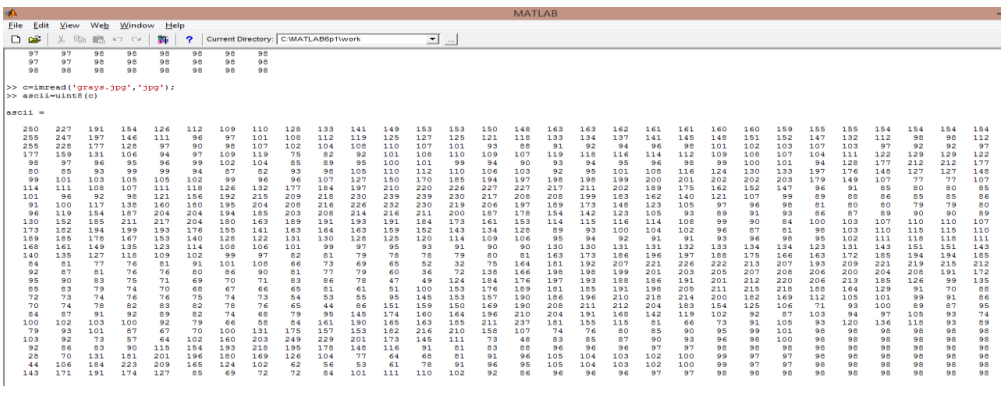
Langkah 9: Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (8,0) dari *kernel*.

128	133	141	149	153
108	112	119	125	127
102	104	108	110	107
75	82	92	101	108
85	89	101	100	101

→

1	2	3	2	1
2	4	6	4	2
3	6	7	6	3
2	4	6	4	2
1	2	3	2	1

G(x,y) = (1x128) + (2x133) + (3x141) + (2x149) + (1x153) + (2x108) +(4x112) + (6x119) + (4x125) +(2x107) + (3x102) + (6x104) + (7x108) + (6x110) + (3x107) + (2x75) + (4x82) + (6x92) + (4x101) + (2x108) + (1x85) + (2x89) + (3x95) + (2x100) + (1x101) = (128 + 266 + 428 + 298 + 153 + 216 + 448 + 714 + 500 + 254 + 306 + 624 + 756 + 660 + 321 + 150 + 328 + 552 + 404 + 216 + 85 + 178 + 285 + 200 + 101) = 8571 : 79 = 108.493
Nilai *pixel* yang diubah adalah 108 menjadi 108
Langkah 9 sampai dengan **Langkah 600** sama dengan **Langkah 8**, dimana konvolusi diabaikan sehingga *pixel-pixel* sudut bawah, nilainya tetap sama seperti citra asal.



Gambar 5. Matrik citra Luka bakar

TIN: Terapan Informatika Nusantara
Vol 1, No 7, Desember 2020, Hal 339-349
ISSN 2722-7987 (Media Online)
Website https://ejurnal.seminar-id.com/index.php/tin

4. KESIMPULAN

Berdasarkan analisa dan pembahasan yang dilakukan, maka dapat disimpulkan, metode *Laplacian of Gaussian* dapat mendeteksi tepi luka bakar pada manusia. Proses pengenalan untuk deteksi tepi perlu dilakukan sebelum mendeteksi objek yang penting, untuk memudahkan dalam proses deteksi tepi luka bakar pada manusia.

REFERENCES

J. Fisika, "Pendeteksian Tepi Citra CT Scan dengan Menggunakan Laplacian of Gaussian (LOG)," *Positron*, vol. II, no. 1, pp. 17–22, 2012.

J. Antivirus, J. Informatika, F. Teknologi, U. Islam, and B. Blitar, "Computing Grayscale of Face Detection Menggunakan Metode Sobel Dan," vol. 11, no. 1, pp. 26–34, 2017.

A. Ritonga, "Implementasi Pengolahan Citra Dalam Proses Deteksi Tepi Dengan Metode Laplacian of," vol. 1, no. 2, pp. 20–22, 2016.

S. Komputer, K. Kunci, D. Tepi, and O. Eigenface, "5.3 Hal 17 - 21 JIM_PERBANDINGAN KINERJA METODE DETEKSI TEPI PADA PENGENALAN OBJEK MENGGUNAKAN OpenCV," *J. Inform. Mulawarman*, vol. 11, no. 2, pp. 17–21, 2016.

Darma Putra. (2010). *Pengolahan Citra Digital*. Yogyakarta: C.V ANDI OFFSET.