**KLASIFIKASI ABNORMALITAS SPERMHEAD MANUSIA DENGAN MANUSIA DENGAN MENGGUNAKAN METODE MORFOLOGI DAN ALGORITMA CONVOLUTION NEURAL NETWORK (CNN)**

**TUGAS AKHIR**

Oleh :

**BELSINTA TAMA LUMBATORUAN**

NIM. 161111639

**RIRIN ROULI SIMANJUNTAK**

NIM. 161111515

**RIDA RAZKITA**

NIM. 161113311



**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH TINGGI MANAJEMEN INFORMATIKA DAN KOMPUTER**

**MIKROSKIL**

**MEDAN**

**2019**

# ABSTRAK

Kata Kunci : sperma

**ABSTRACT**

# KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa, atas Berkat, Rahmat dan Karunia-Nya, penulis dapat menyelesaikan Tugas Akhir yang berjudul **“Klasifikasi Abnormalitas Spermhead Manusia dengan menggunakan Metode Morfologi dan Algoritma Convolution Neural Network (CNN)”.**

Tugas Akhir Skripsi ini disusun untuk memenuhi persyaratan kelulusan Sarjana Strata Satu (S1) pada Program Studi Teknik Informatika STMIK Mikroskil Medan.

Adapun keberhasilan dan kelancaran dalam penyusunan dan penulisan skripsi ini tidak lepas dari bantuan dan dukungan dari berbagai pihak. Oleh karena itu, pada kesempatan ini penulis ingin mengucapkan terima kasih kepada :

1. Bapak Dr. Pahala Sirait, S.T., M.Kom., selaku Dosen Pembimbing I (satu) yang telah meluangkan waktu untuk memberikan bimbingan, saran serta motivasi di saat menempuh studi maupun penyusunan skripsi.
2. Bapak Ali Akbar Lubis, S.Kom., M.Kom. selaku Dosen Pembimbing II (dua) yang telah meluangkan waktu untuk memberikan bimbingan, saran serta motivasi di saat menempuh studi maupun penyusunan skripsi.
3. Bapak Dr. Pahala Sirait, S.T., M.Kom., selaku Ketua STMIK Mikroskil Medan.
4. Bapak Djoni, S.Kom., M.T.I., selaku Wakil Ketua I STMIK Mikroskil Medan.
5. Bapak Gunawan, S.Kom., M.T.I. selaku Ketua Program Studi Teknik Informatika STMIK Mikroskil Medan.
6. Bapak Sunaryo Winardi, S.Kom., M.T. selaku Sekretaris Program Studi Teknik Informatika STMIK Mikroskil Medan.
7. Bapak dan Ibu Dosen yang telah mendidik dan membimbing penulis dalam mengerjakan Tugas Akhir ini.
8. Orang Tua yang selalu memberi dukungan, doa, motivasi dan materi selama penulis menyelesaikan Tugas Akhir ini.
9. Keluarga dan sanak saudara penulis yang telah banyak memberikan dorongan dan dukungan serta bantuan dalam menyelesaikan tugas akhir ini.
10. Seluruh teman-teman seperjuangan yang membantu dalam bentuk doa dan dukungan, serta pihak-pihak yang memberi masukan dalam Tugas Akhir ini.
11. Semua pihak yang memberikan dukungan dan semangat kepada penulis sehingga dapa menyelesaikan Tugas Akhir ini.

Akhir kata penulis menyadari bahwa dalam penulisan skripsi ini masih jauh dari kesempurnaan. Karena itu penulis memohon saran dan kritik yang sifatnya membangun demi kesempurnaannya. Semoga hasil dari Tugas Akhir ini bermanfaat bagi pihak yang berkepentingan.

Medan, xx 2020

Penulis

# DAFTAR ISI

[ABSTRAK i](#_Toc31632840)

[KATA PENGANTAR iii](#_Toc31632841)

[DAFTAR ISI v](#_Toc31632842)

[DAFTAR GAMBAR vii](#_Toc31632843)

[DAFTAR TABEL viii](#_Toc31632844)

[DAFTAR LAMPIRAN ix](#_Toc31632845)

[BAB I PENDAHULUAN 1](#_Toc31632846)

[1.1 Latar Belakang 1](#_Toc31632847)

[1.2 Rumusan Masalah 2](#_Toc31632848)

[1.3 Tujuan 3](#_Toc31632849)

[1.4 Manfaat 3](#_Toc31632850)

[1.5 Batasan Masalah 3](#_Toc31632851)

[1.6 Metodologi Penelitian 4](#_Toc31632852)

[BAB II TINJAUAN PUSTAKA 5](#_Toc31632853)

[2.1 Analisis Data 5](#_Toc31632855)

[2.1.1 Analisis Histogram 5](#_Toc31632856)

[BAB III ANALISIS DAN PERANCANGAN 6](#_Toc31632857)

[3.1 Analisis 6](#_Toc31632859)

[3.1.1 Analisis Masalah 6](#_Toc31632860)

[BAB IV HASIL DAN PENGUJIAN 7](#_Toc31632861)

[BAB V KESIMPULAN DAN SARAN 7](#_Toc31632863)

[5.1 Kesimpulan 7](#_Toc31632865)

[5.2 Saran 8](#_Toc31632866)

[DAFTAR PUSTAKA 9](#_Toc31632870)

# DAFTAR GAMBAR

# DAFTAR TABEL

# DAFTAR LAMPIRAN

Lampiran 1. Daftar Riwayat Hidup

# BAB I PENDAHULUAN

## Latar Belakang

Sel spermatozoa (*sperm cell)* adalah sel gamet jantan yang diproduksi oleh manusia laki-laki. Sel sperma berbentuk menyerupai kecebong, tersusun atas tiga bagian utama , yaitu kepala (*head*), leher (*coneccting piece*), dan ekor (*tail*) (Andrianto, 2017). Kelainan sperma terdapat pada bentuk kepala menurut kategori WHO antara lain : bentuk kepala yang runcing disebut *Tapered*, bentuk kepala seperti buah *pear* disebut juga *Pyriform,* bentuk kepala yang tidak beraturandisebut juga *amorf.* Banyaknya faktor penyebab turunnya produktivitas seorang pria seperti gaya hidup masa kini, jenis makanan, pola makan, dan kebiasaan sehari-hari yang tidak sehat membuat kualitas sperma menjadi buruk. Standar laboratorium untuk melihat kesuburan sperma masih dilakukan berdasarkan morfologi sperma(WHO, 2010). Saat ini klasifikasi bentuk sperma masih dilakukan secara manual dan bersifat subjektif (Johny BF, 2011). Masalah yang ditemukan pada penelitian sebelumnya ialah pengklasifikasian abnormalitas sel *spermhead* masih menggunakan dataset tunggal sehingga perlu dilakukan proses pemotongan gambar.

Dalam membedakan masing-masing objek dan background digunakan metode morpologi. Pada penelitian sebelumnya dalam tahap *pre-prosesing* citra dikonversi menjadi gambar *grayscale* yang kemudian dilanjutkan dengan melakukan proses *mean filter* untuk menghilangkan noise pada gambar. Setelah itu citra dikonversi kembali menjadi citra biner dan kemudian disegmentasi menggunakan *global threshold* (Metode Otsu)untuk memisahkan warna sperma tunggal dan latar belakang*.*  Metode ini memiliki akurasi yang baik dalam mengektrak objek sperma dari citra gambar namum masih banyak noise yang ikut tertangkap oleh sistem (Hatta & Susrama, 2017). Sedangkan untuk mengklasifikasi setiap kepala sperma menjadi masing-masing kelas perlu memperhitungkan ketidakpastian bentuk kelas dari masing-masing kepala sperma. Sehingga sangat sulit jika hanya menggunakan metode morpologi, dan diperlukan algoritma pengklasifikasian untuk mendapatkan akurasi yang lebih baik (Chang, Heutte, Petitjean, Hartel, & Hitschfeld, 2017).

Penelitian sebelumnya proses *classification* menggunakan algoritma SVM dimana kernel sebagai parameter utamanya dan nantinya objek akan dipisah menjadi beberapa kelas yang memberikan tingkat *accuracy* sebesar 82.33% (Ilhan, Serbes, & Aydin, 2018). Algoritma SVM hanya menganggap setiap gambar yang diinput sebagai vektor (Ding, Liu, & Mengyuan, 2016). Penelitian ini memiliki hasil yang lumayan baik, namun tingkat keberhasilan masih dapat ditingkatkan lagi dengan cara menambah atau mengganti metode yang sudah diterapkan. Pengujian lain dilakukan dengan menggunakan algoritma CNN (*Convolution Neural Network*) untuk pengklasifikasian abnormalitas spermhead dengan membandingkan dua dataset yang berbeda. CNN secara signifikan mengurangi jumlah parameter yang perlu dilatih dibandingkan algoritma klasifikasi yang lainnya (Ghamisi, Yushi, & Xiao Xiang, 2016). CNN dapat menangani citra secara langsung dan dapat mengeksploitasi setiap piksel dengan operasi konvolusi yang membuatnya lebih baik dalam mengklasifikasikan gambar yang ada (Ding, Liu, & Mengyuan, 2016).

Oleh karena itu, penelitian ini akan dilakukan dengan beberapa tahap proses yaitu tahap *pre-processing* dan tahap klasifikasi. Pada tahap *pre-processing* akan dilakukan penginputan gambar RGB yang kemudian akan dilakukan perbaikan kualitas citra dengan median filter dan di konversi menjadi citra *Grayscale* dan citra biner. Selanjutnya citra biner akan diolah dengan *connected* *component labeling* . Dimana *connected component labeling* ini adalah metode yang digunakan untuk menentukan region atau objek dalam citra digital. (yudhistiro, 2017). Kemudian pada tahap klasifikasi akan diterapkan algoritma CNN (Convolutional Neural Network), dimana algoritma ini melatih dan menguji setiap gambar masukan melalui serangkaian proses, yaitu Feature Learning dan Classification. Feature Learning pada metode ini akan digantikan dengan operasi morfologi, boundary edge base dan penyesuaian nilai threshold warna guna untuk mengekstrak fitur bentuk. Karakteristik morfologi yang digunakan ada 3 jenis, yaitu area, parameter, dan morfologi erosi untuk menghilangkan noda atau titik-titik kecil pada citra (Iswan & Saputra S, 2018). Sehingga didapatkan hasil pengklasifikasian yang tepat berdasarkan pencocokan antara citra yang telah diolah dengan dataset yang ada.

Berdasarkan uraian diatas, judul Tugas Akhir ini adalah ***“Klasifikasi Abnormalitas spermhead Manusia menggunakan Metode Morfologi dan Algoritma Convolution Neural Network (CNN)”.***

## Rumusan Masalah

Berdasarkan pada uraian latar belakang di atas, maka yang menjadi permasalahan sehingga perlu dilakukan penelitian ini adalah :

1. Mengklasifikasi bentuk kepala sperma normal, abnormal dengan 3 kelas yaitu *Tapered, Pyriform,* dan *Amorf*  dengan proses pemotongan gambar
2. Mendapatkan *threshold* warna yang sesuai untuk pengklasifikasian

## Tujuan

Tujuan dari tugas akhir ini untuk membangun aplikasi yang dapat mengklasifikasi bentuk abnormalitas sperma dengan menggunakanmetodemorfologi dan*Algoritma Convolution Neural Network (CNN).*

## Manfaat

Dalam penyusunan tugas akhir ini memiliki manfaat sebagai berikut :

1. Memberikan kemudahan untuk ilmu kedokteran dalam mengklasifikasi bentuk abnormalitas sperma.
2. Menjadi refrensi pembelajaran dan penelitian klasifikasi abnormalitas sperma dengan metode morfologi dan *Algoritma Convolution Neural Network (CNN).*
3. Memudahkan kita dalam mempelajari sperma yang sangat beraneka ragam dan juga untuk mengetahui hubungan kekerabatan antara makhluk hidup satu dengan yang lain.

## Batasan Masalah

Adapun batasan masalah dari tugas akhir ini adalah sebagai berikut :

1. Dataset citra sperma diperoleh dari *Human Sperm Head Morphology Dataset(HuSHeM)* dengan judul “*Deep Learning For The Classification of Human Sperm”* (Riordon, McCallum, & Sinton, 2019)*.*
2. Penelitian yang dilakukan pada abnormalitas sperma hanya berdasarkan bentuk kepala sperma.
3. Penelitian ini hanya mengklasifikasi 3 kelas bentuk abnormalitas sperma yaitu *Tapered, pyform, amorf* dan 1 bentuk normal.
4. Gambar dataset yang digunakan berformat .BMP.

## Metodologi Penelitian

Adapun langkah – langkah yang dilakukan dalam pengerjaan tugas akhir ini adalah :

1. Pengumpulan Data

Pada tahap ini pengumpulan data dilakukan untuk kebutuhan dalam pemgembangan aplikasi yang akan dibangun, berikut proses yang terjadi :

1. Mengumpulkan refrensi untuk tugas akhir dari jurnal, buku dan lainnya yang dibutukan untuk membangun aplikasi.
2. Mengumpulkan *dataset* citra *spermhead*  sebagai data pelatihan untuk proses klasifikasi abnormalitas *spermhead.* Dimana *dataset* citra *spermhead*  yang terkumpul sebanyak 54 citra *spermhead*  normal, 53 citra *spermhead*  abnormal *tapered,* 57 citra *spermhead*  abnormal *pyriform,* dan 52 citra *spermhead*  abnormal *amorf*.
3. Metodologi pengembangan sistem yang digunakan pada penelitian tugas akhir ini adalah dengan menggunakan metodologi *waterfal* (Pressman, 2002):
4. Analisis Kebutuhan Sistem

Pada tahap ini dilakukan identifikasi kebutuhan pengguna dan sistem secara fungsional dan non-fungsional. Identifikasi kebutuhan fungsional menggunakan usecase diagram dan identifikasi kebutuhan non-fungsional menggunakan PIECES (*Performance, Information, Efficiency, Control, Economy, Service*).

1. Perancangan aplikasi

Perancangan tampilan *user interface* dari sistem akan didesain dengan menggunakan *software Balsamiq*.

1. Penulisan program

Penulisan kode program dilakukan dengan menggunakan C#.net.

1. Pengujian Aplikasi

Proses pengujian ini dilakukan dengan membandingkan 25 buah citra dari HuSHem (*Human Sperm Head Morphology dataset*). Skenario pengujian citra *sperm head* normal dan abnormal menggunakan *threshold* warna dan pengujian menggunakan algoritma CNN untuk mengklasifikasi bentuk-bentuk *spermhead* sehinggadapat diketahui dan dipisahkan berdasarkan kelas nya. Setelah abnormalitas *spermhead* diklasifikasi, akan dilakukan pengujian *accuracy*, *precision* dan *recall* dari semua data yang telah diuji.

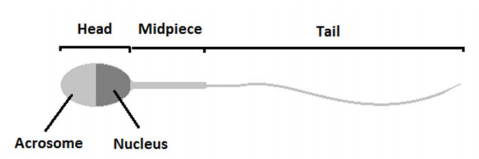
1. Penarikan Kesimpulan

Penarikan kesimpulan diambil berdasarkan hasil pengujian yang dilakukan pada tahap sebelumnya.

# BAB II TINJAUAN PUSTAKA

## Spermatozoa

Spermatozoa merupakan sel yang dihasilkan oleh fungsi reproduksi pria. Sel tersebut memiliki bentuk yaitu kepala, badan, dan ekor. Pada manusia jumlah sperma yang diejakulasikan secara normal sekitar 2-6 ml dengan pH antara 7,2-7,6. Komposisi cairan semen untuk setiap kali ejakulasi normal pada manusia dinyatakan sebagai berikut (Hardijanto, et al., 2010):



Gambar 2.30 Bagian-bagian dari Sperma

(World Health Organization laboratory manual for the examination and processing of human semen, 2010)

Bentuk sperma sempurna adalah merupakan sel yang memanjang, yang terdiri dari kepala dengan bentuk tumpul yang didalamnya terdapat *nucleus* atau inti, dan ekor yang mengandung *apparatus* untuk pergerakan sel. Pada kepala terdapat akrosom yang memiliki struktur dinding yang rangkap dan terletak diantara membran plasma bagian anterior *nucleus*, bagian akrosom harus ada dan menutupi 1/3 bagian kepala sperma, ukuran panjang kepala sperma harus 3-5 mikron, lebar kepala sperma harus 1/2 atau 1/3 dari panjangnya kepala.leher menhubungkan kepala dan ekornya (*flagela*) yang dibagi lagi menjadi bagian tengah, pokok dan akhir yang bagia-bagian tersebut mempunyai struktur yang berbeda (World Health Organization laboratory manual for the examination and processing of human semen, 2010).

### *Sperm Head* Normal

Bentuk utama dari kepala sperma normal adalah oval, kepala harus halus, dan berkontur teratur. Harus ada wilayah akrosom yang terdefinisi dengan baik yang terdiri dari 40-70% dari luas kepala. Wilayah akrosom seharusnya tidak mengandung vakuola besar, dan tidak lebih dari dua vakuola kecil, yang seharusnya tidak menempati lebih dari 20% dari kepala sperma. Wilayah pasca-akrosom seharusnya tidak mengandung vakuola (World Health Organization laboratory manual for the examination and processing of human semen, 2010)

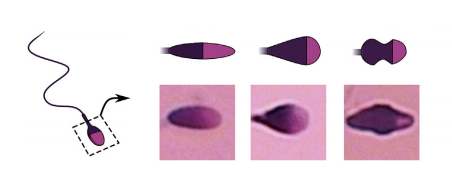


Gambar 2.31 Bentuk Kepala Sperma Normal

(Deep Learning For The Classification of Human Sperm , 2019)

### *Sperm Head Abnormal*

Abnormalitas sperma dapat dibedakan menjadi dua yaitu abnormalitas primer dan abnormalitas sekunder. Abnormalitas primer yang berhubungan dengan kepala dan akromoso. Abnormalitas sekunder terjadi adanya sitoplasmic droplet pada mid piece pada ekor.



Gambar 2.32 Bentuk Kepala Sperma Abnormal

(Deep Learning For The Classification of Human Sperm , 2019)

* 1. Tapered

Kepala pipih (bentuk lepto). Spermatozoa yang mempunyai kepala dengan perbandingan ukuran lebar lebih pendek dari pada panjangnya. Kepala sperma kelompok ini berbentuk cerutu dengan kedua sisi sejajar yang kemudian dapat bertemu dengan titik. Panjang sperma > 7 mikron dan lebar sperma > 3 mikron (Susilawati, 2011).

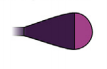


Gambar 2.33 Bentuk kepala sperma *Tapered*

(Deep Learning For The Classification of Human Sperm , 2019)

* 1. Pyriform

Kepala berbentuk pir. Kepala lebih menyolok berbentuk sebagai “tetesan air mata”, bagian runcingnya berhubungan dengan bagian tengah sperma, disini ukurannya tidak diperhatikan .



Gambar 2.34 Bentuk kepala sperma *Pyriform*

(Deep Learning For The Classification of Human Sperm , 2019)

* 1. Amorf

Morfologi kepala tidak jelas bentuknya batas akrosom tidak jelas dan umumnya tanpa akrosom .

D:\TA\pic\amorf1.png

Gambar 2.35 Bentuk kepala sperma *Amorf*

(Deep Learning For The Classification of Human Sperm , 2019)

## Pengertian Citra

Citra (*image*) adalah salah satu komponen multimedia yang memegang peranan yang sangat penting sebagai bentuk informasi visual. Citra sebagai kumpulan dari titik (*pixel*) yang mempunyai identitas tertentu untuk membentuk satu k esatuan perpaduan yang mempunyai arti, baik secara “artistik” maupun “intrinsik”. Citra yang dapat menampilkan keindahan gambar (aristik) dan kejelasan gambar untuk penganalisaan dan maksud-maksud lainnya (intrinsik) adalah karakteristik citra yang baik (Munir, 2004).

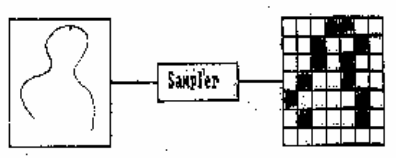
*Pixel*  menunjukkan satuan dari sebuah citra. *Pixel* merupakan singkatan dari *picture element*, yaitu unsur gambar atau representasi sebuah titik terkecil dalam sebuah gambar grafis yang dihitung per-inci. Setiap *pixel* mewakili tidak hanya satu titik dalam sebuah citra melainkan sebuah bagian berupa kotak yang merupakan bagian terkecil yang biasa disebut sel (Putra, 2010).

Ada dua macam jenis citra yaitu analog dan citra digital. Citra *analog* adalah citra yang bersifat kontinu seperti gambar pada monitor televisi, foto sinar x, hasil CT *Scan* dan lain-lain. Sedangkan pada citra *digital* adalah citra yang dapat diolah oleh computer (Sutoyo, et al., 2009)

### Citra Analog

Citra *analog*  adalah citra yang bersifat *continue,* seperti citra pada monitor, telelvisim foto sinar X, foto yang tercetak dikertas foto, lukisan, hasil *CT scan,* citra - citra yang terekam pada pita kaset, dan lain sebagainya. Citra *analog* tidak dapat direpresentasikan dalam computer, sehingga tidak bias diproses di komputer secara langsung (Munir, 2004).

Oleh sebab itu , agar bias diproses di komputer, proses konversi analog ke digital harus dilakukan terlebih dahulu. . Citra analog dihasilkan dari alat-alat analog, video kamera analog, kamera foto analog, Web Cam, CT *scan*, sensor *ultrasound* pada sistem USG, dan lain-lain (Sutoyo, et al., 2009).



Gambar 2.1 Operasi erosi

(Sutoyo, et al., 2009)

### Citra Digital

Secara matematis, sebuah citra dapat didefinisikan sebagai fungsi dua dimensi *f(x,y),* dimana *x* dan *y* adalah koordinat spasial *(plane)* dan *f(x,y)* adalah nilai intensitas warna pada koordinat *x* dan *y*. Nilai *f, x* dan *y* semuanya adalah nilai berhingga. Citra digital umumnya dua dimensi (2D) yang dinyatakan dalam bentuk matriks dengan jumlah elemen berhingga. Setiap elemen matriks citra memiliki posisi koordinat *x* dan *y* tertentu dan juga memiliki nilai (Madenda, 2015).

Secara umum citra digital merupakan representasi *pixel-pixel* dalam ruang 2D yang dinyatakan dalam *matrix* berukuran N baris dan M kolom. Setiap elemen *matrix* citra disebut *pixel* *(picture element, image element atau pel)*. Nilai setiap *pixel* *f* pada posisi koordinat *x* dan *y* merepresentasikan intensitas warna dan dapat dikodekan dalam 24 bit untuk citra berwarna (dengan tiga komponen warna RGB: R = *red*, G = *green*, dan B = *blue*), 8 bit untuk citra gray-level atau 1 bit untuk citra biner (Madenda, 2015). Pada umumnya citra digital berbentuk empat persegi panjang dan dimensi ukurannya dinyatakan sebagai tinggi x lebar (Munir, 2004). Citra digital yang tingginya N, lebarnya M dan memiliki L derajat keabuan dapat dianggap sebagai fungsi dapat dilihat pada rumus (1) (Munir, 2004) :

|  |  |
| --- | --- |
|  | (1) |

Citra digital yang berukuran N x M lazim dinyatakan dengan *matrix* yang berukuran N baris dan M kolom. Rumus (2) memperlihatkan representasi citra digital ke dalam *matrix* dua dimensi.

|  |  |
| --- | --- |
|  | (2) |

### Elemen-elemen Citra Digital

Di dalam citra digital terdapat elemen-elemen sebagai berikut (Putra, 2010) :

1. Kecerahan (*brightness*)

Kecerahan merupakan intensitas cahaya yang dipancarkan pada sebuah titik (*pixel*) melalui citra yang ditangkap oleh indera penglihatan. Sedangkan intensitas rata-rata dari area yang melingkupinya meliputi kecerahan pada sebuah titik di dalam sebuah citra.

1. Kontras (*contrast*)

Kontras merupakan penyebaran warna terang (*lightness*) dan warna gelap (*darkness*) pada suatu citra. Citra dengan kontras rendah didirikan oleg sebagian besar komposisi citranya adalah terang atau sebagian besar gelap. Pada citra dalam kontra yang baik, komposisi gelap dan terang tersebar secara merata.

1. Kontur (*countour*)

Kontur merupakan proses terjadinya perubahan intensitas cahaya pada *pixel*-*pixel* yang bertetangga. Karena adanya perubahan intensitas maka kita mampu mendeteksi tepi-tepi (*edge*) objek di dalam citra.Warna (*colour*)

1. Warna (colour)

Warna merupakan presepsi panjang gelombang cahaya yang dipantulkan oleh objek dan ditangkap oleh system penglihatan.

1. Bentuk (*shape*)

Bentuk merupakan property intrinsic utama dari objek tiga dimensi yang digunakan untuk *system* penglihatan manusia.

1. Tekstur (*texture*)

Tekstur merupakan keteraturan pola-pola tertentu yang berbentuk dari susunan piksel-piksel didalam citra digital yang dicirikan sebagai distribusi spasial dari derajat keabuan didalm sekumpulan piksel-piksel yang berdekatan sehingga dapat digunakan untk membedakan sifat-sifat spesifik dari kekasaran maupun kehalusan permukaan objek didalam suatu citra.

### Format File Citra

Format *file* menentukan bagaimana informasi data dipresesnntasikan dalam suatu file. Informasi tersebut meliputi ada tidaknya kompresi, program aplikasi (*feature*) yang didukung (*support*), penggunaan enkripsi, dan lain-lain. Tiap format *file* memiliki kelebihan dan kelemahan masing-masing. Dalam sistem operasi *Windows*, format *file* dapat dibedakan dari namanya yaitu diakhiri titik dan diikuti tiga atau empat huruf terakhir sebagai penanda format. Format *file* citra ada yang terkompresi dengan sifat *lossless* dan ada yang bersifat *lossy*. ***Lossless*** adalah jenis kompresi data citra atau data lainnya yang tidak satu pun mengalami perubahan atau kehilangan data/informasi saat dilakukan rekonstruksi, atau dengan kata lain bahwa data hasil rekonstruksi adalah tepat sama dengan data aslinya. ***Lossy*** adalah jenis kompresi data citra atau data lainnya yang dapat mengalami perubahan atau kehilangan data/informasi saat dilakukan rekonstruksi, atau dengan kata lain bahwa data hasil rekonstruksi adalah tidak selalu tepat sama dengan data aslinya. Untuk *file* citra (*image*), format yang umum digunakan adalah (Madenda, 2015) :

1. BMP (*Windows bitmap*)

BMP (*Windows bitmap*) merupakan format *file* grafik dari sistem operasi windows, umumnya *file* BMP yang terkompresi dengan sifat *lossless* dan menggunakan algoritma RLE (*Run-Length Encoding*) dimana dapat menyimpan citra biner, *grayscale,* citra berwarna *true colour,*  dan citra berwarna *indexed* yang digunakan dalam program *Windows*. Pada umumnya, format *file* ini yang berukuran besar tidak dikomporesi.

1. TIF (*Tagged Image Format*)

TIF (*Tagged Image Format*) merupakan format penyimpanan citra yang terkompresi maupun tidak terkompresi seperti citra bitmap hingga berwarna.

1. GIF (*Graphics Interchange Format*)

GIF (*Graphics Interchange Format*) merupakan format *file* bitmap (grafik) yang didasarkan pada jenis data raster 2D dengan berbagai resolusi. Format GIF memiliki kelebihan yaitu dapat menyimpan citra sekuensial (beberapa citra) dalam satu *file* sehingga dapat memberikan efek animasi saat ditampilkan pada layar komputer.

1. PNG (*Portable Network Graphics*)

PNG (*Portable Network Graphics*) merupakan format *file* citra bitmap yang menggunakan algortima kompresi *lossless* untuk menyimpan data citra. PNG merupakan format *file* citra *open source* (dapat digunakan tanpa lisensi hak cipta) dan diciptakan untuk memperbaiki kekurangan format GIF. Format PNG dapat menyimpan citra berwarna dalam ruang warna RGB dengan 24 bit per-*pixel* atau ruang warna RGBA (A adalah *alpha channel*) dengan 32 bit per *pixel*, dan citra *grayscale* baik dengan atau tanpa *alpa channel*.

1. JPEG (*Joint Photographic Experts Group*)

JPEG (*Joint Photographic Experts Group*) merupakan format *file* citra yang digunakan untuk menyimpan dan mengirimkan gambar foto di *World Wide Web*. Namun, nama ini sering tidak digunakan dan tetap hanya disebut JPEG. Format JPEG menggunakan algoritma kompresi yang bersifat *lossy*, dimana kualitas informasi citra sangat bergantung pada banyaknya data atau informasi yang hilang atau berubah. Rasio kompresi JPEG relatif besar, dengan kualitas citra rekonstruksi yang tetap baik.

1. JPEG 2000

JPEG 2000 merupakan format kompresi citra dan sistem pengkodean standar yang dibuat oleh *Joint Photographic Experts Group* pada tahun 2000. Dengan maksud menggantikan metode *Discrete Cosine Transform* (DCT) yang digunakan pada standar JPEG dengan metode terbaru *Discrete Wavelet Transform* (DWT) *multiscale* atau multi-resolusi. Format ini memiliki ekstensi nama *file* standar .jp2 untuk ISO/IEC 15444-1 dan .jpx untuk ISO/IEC 15444-2. Format kompresi JPEG 2000 memiliki rasio dan kualitas kompresi yang lebih baik dibandingkan dengan format kompresi JPEG. Keuntungan utama JPEG 2000 adalah *codestream* (*bitstream*) hasil kompresinya memiliki fleksibilitas resolusi yang signifikan. *Codestream* ini dapat dipenggal-penggal, dimana setiap penggalannya menghasilkan resolusi atau ukuran citra yang berbeda-beda. Namun, format ini memiliki kelemahan dibandingkan JPEG yaitu algoritma kompresi dan dekompresi (*encoder & decoder*) yang lebih kompleks dan waktu komputasinya pun lebih lama.

### Jenis-jenis Citra

Untuk menyimpan citra dan foto digunakan format citra layar *kuadratis* (berbentuk kotak) yang terdiri atas titik-titik citra kecil yang disebut dengan *pixel. Pixel*  berbentuk bujur sangkar dengan ukuran relatif kecil. Banyaknya *Pixel* tiap satuan luas tergantung pada resolusi yang digunakan (Putra, 2010).

Suatu pixel memiliki nilai rentang tertentu, dari nilai minimum hingga nilai maksimum. Jangkauan 0-255 yang digunakan berbeda-beda tergantung dari jenis warna dari citra tersebut (Madenda, 2015). Beberapa jenis citra yang umum digunakan dalam pemrosesan citra yaitu (Kadir, et al., 2013) :

1. Citra Warna (RGB)

Citra warna (RGB) adalah jenis citra yang memilki 3 buah *channel* warna didalamnya yaitu *channel red, channel green, channel blue.* Setiap *channel* memiliki 8 bit (nilai dari 0-255) yang menghasilkan kemungkinan warna 16.777.216 warna (Kadir, et al., 2013).

|  |  |  |
| --- | --- | --- |
|  |  |  |

Gambar 2.2 citra warna

(Hidayatullah, 2017)

1. Citra Grayscale (Skala Keabuan)

Citra *grayscale*  adalah citra yang memiliki piksel nilai *luminance* 8 bit (28 -1) dimana bervariasi dari nilai 0 sebagai warna hitam sampai 255 sebagai warna putih, dan nilai diantaranya mempresentasikan warna keabuan yang bervariasi dan warna gelap (hitam) menuju warna cerah (putih) (Madenda, 2015).

|  |  |  |
| --- | --- | --- |
|  |  |  |

*Gambar 2.3 citra grayscale*

(Hidayatullah, 2017)

1. Citra Biner

Citra biner atau citra hitam putih (*black and white image*) adalah citra yang memiliki *pixel* nilai *luminance* 1 bit dimana nilai untuk setiap *pixel-*nya hanya ada 2 kemungkinan nilai yaitu 0 atau 1. Nilai 0 sebagai warna hitam sedangkan nilai 1 sebagai warna putih. Jenis citra ini sering digunakan untuk segmentasi citra (Hidayatullah, 2017). Citra biner diperoleh berdasarkan nilai ambang (*Threshold*) dimana jika nilai *pixel*  lebih kecil daripada nilai *threshold*  maka nilai *pixel* diubah menjadi 0 (hitam) dan sebaliknya jika nilai *pixel* lebih besar atau sama dengan nilai *threshold* maka nilai *pixel*  diubah menjadi 1 (Putih) (Madenda, 2015).



Gambar 2. 4 Citra biner buah

(Hidayatullah, 2017)

## Pengolahan Citra

Banyaknya informasi yang tersirat di dalam sebuah citra tidak akan terhindar dari penurunan kualitas (degredasi) diantaranya terdapat derau (*noise*), warna yang terlalu kontras maupun kurang tajam, efek warna yang tidak merata mengakibatkan suatu citra sulit untuk dikenali dikarenakan adanya *blurring* (kabur), dan lain sebagainya. Oleh sebab itu, perlu dilakukan pengolahan citra sebagai proses untuk memanipulasi citra yang terdegrasi menjadi kualitas yang lebih baik. Pemrosesan citra dilakukan, dengan maksud mendapatkan kualitas citra yang diinginkan, terdapat berbagai operasi pengolahan citra yang dapat diklasifikasikan dalam berbagai jenis. Operasi-operasi yang dilakukan di dalam pengolahan citra banyak ragamnya. Namun, secara umum operasi pengolahan citra dapat diklasifikasikan dalam beberapa jenis sebagai berikut (Munir, 2004):

1. Perbaikan Kualitas Citra (*image enhancement*)

Jenis operasi ini bertujuan untuk memperbaiki kualitas citra dengan cara memanipulasi parameter parameter citra. Dengan operasi ini, ciri-ciri khusus yang terdapat di dalam citra lebih ditonjolkan. Contoh-contoh operasi perbaikan citra:

1. Perbaikan kontras gelap/terang.
2. Perbaikan tepian objek (*edge enhancement*).
3. Penajaman (*sharpening*).
4. Pemberian warna semu (*pseudocoloring*).
5. *Noise filtering*.
6. Pemugaran Citra (*image restoration*)

Pemugaran citra (*image restoration*) adalah proses merekontruksi atau perbaikan penurunan kualitas citra yang disebabkan oleh derau (*noise*) sehingga kualitas citra menjadi keadaan asli (Kadir, et al., 2013). *Image restoration* berfungsi untuk meminimumkan atau menghilangkan kerusakan (degredrasi) pada citra seperti menghilangkan kesamaran (*deblurring*) dan derau (*noise*) (Munir, 2004). Contoh-contoh operasi pemugaran citra:

1. Penghilangan kesamaran (*deblurring*).
2. Penghilau derau (*noise*).
3. Pemampatan Citra (*image transformation*)

Jenis operasi ini dilakukan agar citra dapat dipresentasikan dalam bentuk yang lebih kompak sehingga memerlukan memori yang lebih sedikit. Hal penting yang harus diperhatikan dalam pemampatan adalah citra yang telah dimampatkan harus tetap mempunyai kualitas gambar yang bagus. Contoh metode pemampatan citra adalah metode JPEG, *Huffman,* RLE, kuantisasi, fractal, dan lain sebagainya.

1. Segmentasi Citra (*image segmentation*)

Tujuan dari operasi ini untuk memecah suatu citra ke dalam beberapa segmen dengan suatu kriteria tertentu. Jenis operasi ini berkaitan erat dengan pengenalan pola. Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya.

1. Analisis Citra (*image analysis*)

Analisis citra berfungsi untuk menghitung nilai kuantitatif dari citra untuk menghasilkan deskripsinya dengan cara mengekstraksi ciri-ciri tertentu yang digunakan untuk mengidentifikasi objek. Proses segmentasi kadangkala diperlukan untuk melokalisasi objek yang diinginkan dari sekelilingnya (Munir, 2004). Contoh-contoh operasi analisis citra:

1. Pendeteksian tepi objek (*edge detection*).
2. Ektraksi batas (*boundary*).
3. Representasi daerah (*region*).
4. Rekonstruksi citra

Jenis operasi ini bertujuan untuk membentuk ulang objek dari beberapa citra hasil proyeksi. Operasi rekonstruksi citra banyak digunakan dalam bidang medis. Misalnya beberapa foto rontgen dengan sinar X digunakan untuk membentuk ulang gambar organ tubuh.

Operasi geometri adalah proses perubahan hubungan spasial antara setiap *pixel* pada sebuah citra. Operasi geometri memetakan kembali *pixel* citra *input* dari posisi awal ke posisi baru pada citra *output*. Proses yang tergolong ke dalam operasi geometri diantaranya adalah translasi, penskalaan, rotasi, refleksi (Putra, 2010). Pada operasi geometrik, koordinat *pixel* berubah akibat transformasi, sedangkan intensitasnya tetap. Pengubahan geometri dari citra menjadi citra baru (Munir, 2004) dapat ditulis sebagai :

(4)

Yang dalam hal ini, dan adalah fungsi transformasi geometrik. Dengan kata lain,

(5) (6)

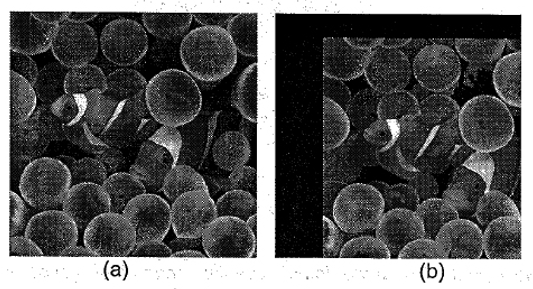
1. Translasi

Rumus translasi citra :

(7) (8)

Yang dalam hal ini, *m* adalah besar pergeseran dalam arah x, sedangkan *n* adalah besar pergeseran dalam arah y. Jika citra semula adalah A dan citra hasil translasi adalah B, maka translasi dapat diimplementasikan dengan menyalin citra dari A ke B :

(9)



Gambar 2. 5 (a) citra semula (b) citra hasil translasi

(Putra, 2010)

1. Rotasi

Rumus rotasi citra :

(10)

(11)

Yang dalam hal ini, = sudut rotasi berlawanan arah jarum jam, lihat gambar 2.9. Jika citra semula adalah A dan citra hasil rotasi adalah B, maka rotasi citra A ke B :

(12)



Gambar 2. 6 (a) citra semula (b) citra hasil rotasi

(Syafrizal, 2015)

1. Refleksi

Refleksi atau pencerminan adalah proses pengolahan citra secara geometri dengan memindahkan nilai-nilai *pixel* pada posisi awal menuju ke posisi baru di pada citra *output* sesuai dengan posisi pencerminan. Posisi pencerminan ada tiga jenis yaitu pencerminan terhadap sumbu *x*, pencerminan terhadap sumbu *y*, dan pencerminan terrhadap sumbu *x* dan *y*.

Pencerminan terhadap sumbu *x* di posisi dapat digambarkan dengan rumus berikut ini.

(13)

(14)

Sedangkan untuk pencerminan terhadap sumbu *y* di posisi adalah :

(15)

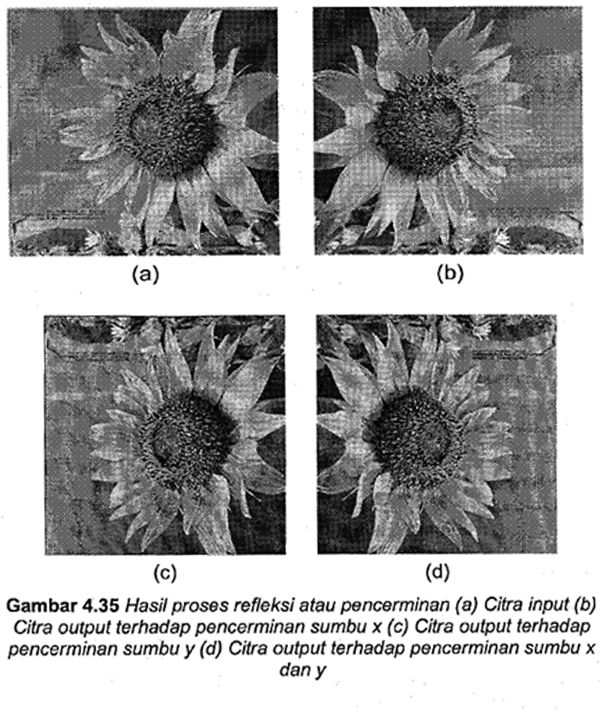
(16)

Dan untuk pencerminan yang dilakukan terhadap kedua sumbu baik *x* dan *y* di posisi adalah:

(17)

(18)

Gambar dibawah ini menunjukan hasil refleksi baik terhadap sumbu *x*, sumbu *y* maupun gabungan antara sumbu *x* dan *y* :

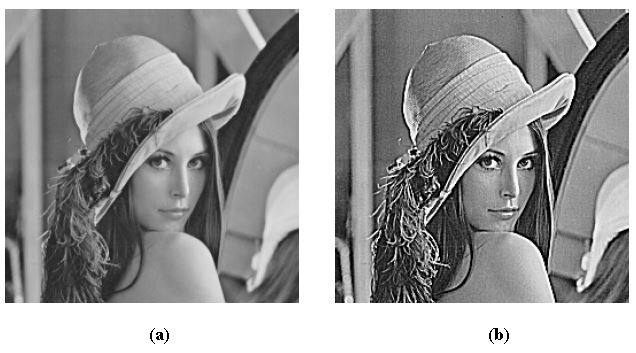


Gambar 2.7 (a) Citra awal (b) Citra hasil pencerminan terhadap sumbu x

(Putra, 2010)

### Perbaikan Kualitas Citra (*image enchamcement*)

Perbaikan Kualitas Citra (*image enhancement*) adalah proses perbaikan sudut pandang citra menjadi lebih baik (Kadir, et al., 2013). *Image Enhancement*  berfungsi untuk memperbaiki kualitas citra dengan cara memanipulasi parameter-parameter citra seperti perbaikan kontras pada citra yang gelap menjadi terang, perbaikan tepi objek (*edge* *enhancement*), penajaman (*sharpening*), pemberian warna semu (*pseudocoloring*), dan pelapisan derau (*noise filtering*) sehingga ciri-cir khusus yang terkandung didalam citra lebih terlihat jelas (Munir, 2004).

**

Gambar 2.8 Perbaikan Citra

Gambar (a) Citra Wanita asli, (b) Citra Wanita setelah ditajamkan

(Statistic For Image Sharpening, 2006)

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 komputer. Perbaikan terhadap suatu citra dapat dilakukan dengan operasi titik (*point operation*), operasi spasial (*spasial operation*), operasi geometri (*geometric operation*) dan operasi aritmatik (*arithmetic operation*) (Putra, 2010). Pada proses perbaikan kualitas citra dilakukan penghilangan *noise* yang dapat mengganggu proses, proses untuk menghilangkan *noise* menggunakan metode *Median Filtering*.

*Median* adalah nilai tengah dari kumpulan data. Untuk mencari *median* dari kumpulan data maka (Image Smoothing Menggunakan Mean Filtering, Median Filtering, Modus Filtering dan Gaussian Filtering, 2010):

(19)

Keterangan : Nilai baru *median*

Jumlah data

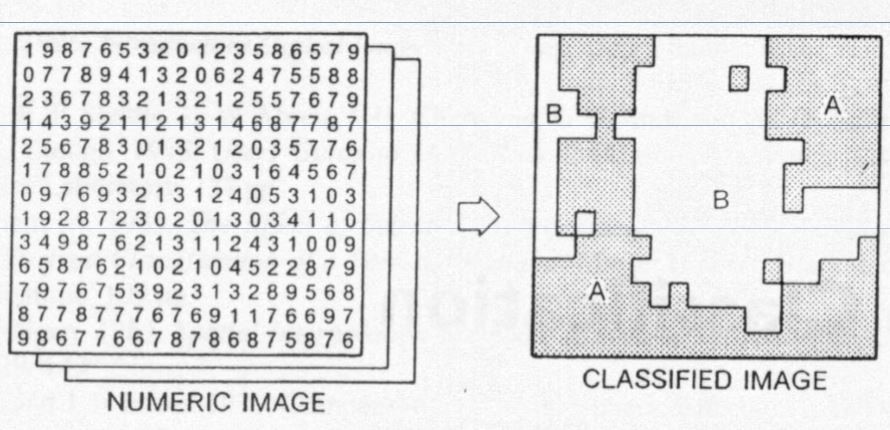
Pada *median filtering* digunakan *matrix* berdimensi N x N. Dari *matrix* tersebut, kemudian data yang ada diurutkan dan dimasukkan dalam sebuah kumpulan data yang telah urut tersebut. Setelah didapatkan nilai tengahnya maka di pertengahan nilai dari *mask* yang diduduki nilai asli diganti dengan nilai yang sudah dirata-rata tersebut. lalu dilanjutkan ke daerah berikutnya (Image Smoothing Menggunakan Mean Filtering, Median Filtering, Modus Filtering dan Gaussian Filtering, 2010).

### Klasifikasi Citra

Klasifikasi citra adalah salah satu tugas utama di bidang *computer* *vision* dan pengolahan citra. Berbagai aplikasi *computer* *vision* melibatkan klasifikasi citra mulai dari mesin pencarian gambar, pendaftaran citra, pengenalan objek dan lokalisasi tempat dalam sistem navigasi (Pengenalan jenis kendaraan menggunakan statistical algorithm dan support vector machine, 2015).

Pada dasarnya, tugas klasifikasi citra terdiri dari membentuk representasi yang tepat dari gambar dan kemudian membandingkan representasi ini dalam rangka untuk menemukan korespondensi. Klasifikasi citra adalah masalah yang menantang dalam menemukan kesamaan diantara gambar yang mewakili objek yang sama secara reliabel berdasarkan deskripsi benda atau dengan kata lain menggambarkan sebuah gambar berdasarkan adegan semantik yang diwakilinya (Pengenalan jenis kendaraan menggunakan statistical algorithm dan support vector machine, 2015).

Sejumlah teknik klasifikasi citra yang didasarkan pada deskriptor fitur diantaranya yaitu SIFT, SURF, FAST dan ORB. Klasifikasi citra menggunakan deskriptor fitur terdiri dari tiga tahap yaitu deteksi fitur, ekstraksi fitur dan pencocokan fitur (Pengenalan jenis kendaraan menggunakan statistical algorithm dan support vector machine, 2015).



Gambar 2. 9 Klasifikasi Citra

(Mani, 2014)

## Thresholding

*Thresholding* adalah proses pengambangan yang akan menghasilkan citra biner, yaitu citra yang memiliki dua nilai tingkat keabuan yaitu hitam dan putih, sehingga dapat diketahui daerah mana yang termasuk objek dan *background* dari citra secara jelas. Citra hasil *thresholding* biasanya digunakan lebih lanjut untuk proses pengenalan objek serta ekstraksi fitur. Metode *thresholding* secara umum dibagi menjadi dua, yaitu (Putra, 2010) :

1. *Global Thresholding*

*Thresholding* dilakukan dengan mempartisi histogram dengan menggunakan sebuah *threshold* (batas ambang) *global* T, yang berlaku untuk seluruh bagian citra. Pada pengambangan *global* akan banyak informasi yang hilang karena hanya menggunakan satu nilai T untuk keseluruhan nilai *pixel*. Persamaan yang digunakan untuk mengkonversi nilai *pixel* citra *grayscale* menjadi biner adalah sebagai berikut :

|  |  |
| --- | --- |
|  | (20) |

Keterangan :

*f(x,y)* : Citra *grayscale*

*g(x,y)* : Citra biner

T : Nilai *threshold*

Nilai T memegang peranan yang sangat penting dalam proses pengambangan. Kualitas hasil citra biner sangat bergantung pada nilai T yang digunakan.

1. *Adaptive Thresholding*

Pengambangan adaptif (*adaptive thresholding*) merupakan pengambangan yang menggunakan nilai ambang lokal, yang dihitung secara adaptif berdasarkan statistik *pixel-pixel* tetangga. Hal ini didasarkan kenyataan bahwa bagian-bagian kecil dalam citra mempunyai iluminasi sama, sehingga lebih tepat kalau nilai ambang dihitung berdasarkan bagian-bagian kecil dalam citra dan bukan berdasarkan seluruh *pixel* dalam citra (Kadir, et al., 2013).

Ada tiga pendekatan yang biasa digunakan untuk mewujudkan pengambangan adaptif. Pendekatan pertama menggunakan statistika rerata terhadap intensitas lokal. Terkadang nilai konstan ikut dilibatkan. Rumusnya sebagai berikut.

(21)

Pendekatan pengambangan adaptif yang kedua menggunakan statistik median. Secara matematika pendekatan ini dapat ditulis menjadi.

(22)

Pendekatan pengambangan adaptif yang ketiga menggunakan statistik maksimum dan minimum, yang dinyatakan degan rumus.

(23)

Dengan keterangan masing-masing rumus :

W : Jendela pada citra

Nw : Jumlah *pixel* dalam jendela

C : Konstanta

## Konvolusi

Konvolusi terdapat pada operasi pengolahan citra yang mengalikan sebuah citra dengan sebuah *mask*atau kernel (Hidayatullah, 2017). Konvolusi sering kali dilibatkan dalam operasi ketetanggan *pixel*. Konvolusi pada citra sering disebut konvolusi dua-dimensi (konvolusi 2D). konvolusi 2D didefinisikan sebagai proses untuk memperoleh suatu *pixel* didasarkan pada nilai *pixel* itu sendiri dan tetangganya, dengan melibatkan suatu *matrix* yang disebut kernel yang mempresentasikan pembobotan (Kadir, et al., 2013).

Konvolusi memiliki 2 buah fungsi *f(x)*dan *g(x)*di definisikan sebagai berikut :

(24)

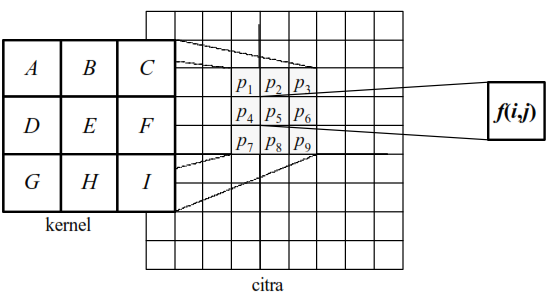
yang dalam hal ini tanda \* menyatakan operasi konvolusi, dan peubah (variabel) α adalah peubah bantu (*dummy* variabel).

            Untuk fungsi diskrit, konvolusi di definisikan sebagai berikut :

(25)

Pada operasi konvolusi diatas, *g(x)*disebut kernel konvolusi atau kernel penapis (*filteri*). Kernel g*(x)*merupakan suatu jendela yang dioperasikan secara bergeser pada sinyal masukan *f(x)*, yang dalam hal ini, jumlah perkalian kedua fungsi pada setiap titik merupakan hasil konvolusi yang dinyatakan dengan keluaran *h(x)*.

Contoh konvolusi ditunjukkan pada Gambar 2.13 berikut :



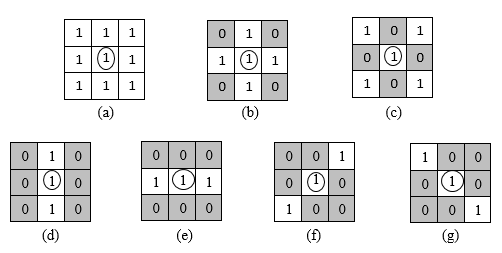
Gambar 2. 10 Contoh Konvolusi

(Hidayatullah, 2017)

## Morfologi

Morfologi (*morphology*) dapat diartikan sebagai bentuk dan struktur suatu objek atau dalam deskripsi lainnya disebutkan bahwa morphologi adalah susunan dan hubungan antar bagian pada suatu objek (Putra, 2010). Pada pengolahan citra, operasi morphologi merupakan metode dalam mendeteksi suatu objek gambar, dimana metode ini merepresentasikan citra dua objek dua dimensi sebagai suatu himpunan matematika, misal citra biner digambarkan sebagai suatu himpunanan titik-titik *pixels* dalam bidang biner Z, terbentuk dari titik-titik himpunan A dan titik-titik himpunan B yang nantinya membentuk objek (Matched Filter dan Operasi Morfologi Untuk Estimasi Derajat Kebengkokan Tulang, 2018) **.**

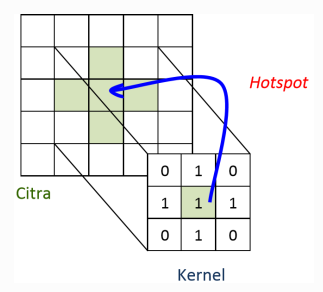
Morphologi cocok digunakan untuk ektraksi pada gambar dengan pemilihan ekstraksi yang didasarkan terhadap bentuk, ukuran, dan orientasi pada gambar (Zhou, et al., 2010). Operasi morphologi bertujuan untuk meningkatkan bentuk (struktur) sehingga lebih mudah untuk di kenali (Matched Filter dan Operasi Morfologi Untuk Estimasi Derajat Kebengkokan Tulang, 2018). Ada 2 operasi morphologi yaitu dilasi dan erosi. Kedua operasi dasar tersebut menjadi basis untuk membuat berbagai operasi seperti *opening, closing, hit and miss transform, thinning,* dan *thickening* (Putra, 2010). Operasi morphologi menggunakan dua input himpunan yaitu suatu citra (pada umumnya citra biner) dan suatu kernel, istilah kernel biasa disebut dengan *structuring elements* (elemen pembentuk struktur) merupakan suatu matrik dan pada umunya berukuran kecil (Putra, 2010). *Structuring elements* digunakan untuk membangun gambar dan output dari gambar tergantung dari kecocokan dari *structuring elements* pada gambar yang diberikan (Zhou, et al., 2010). Secara umum *structuring elements* digunakan sebagai berikut :



Gambar 2. 11 Berbagai bentuk structuring elements

(Hidayatullah, 2017)

Pada gambar posisi piksel yang dilingkari merupakan pusat koordinat dari *structuring elements*. Pemrosesan citra secara morphologi dilakukan dengan cara melakukan *passing* pada sebuah *structuring elements* terhadap sebuah citra, dimana *structuring elements* merupakan sebuah matriks yang terdiri dari bilangan “0” dan “1” (Matched Filter dan Operasi Morfologi Untuk Estimasi Derajat Kebengkokan Tulang, 2018). *Passing*  yang dilakukan dengan menyesuaikan hotspot *structuring elements* secara umum dilakukan sebagai berikut:



Gambar 2.12 Penerapan Structuring Elements pada citra

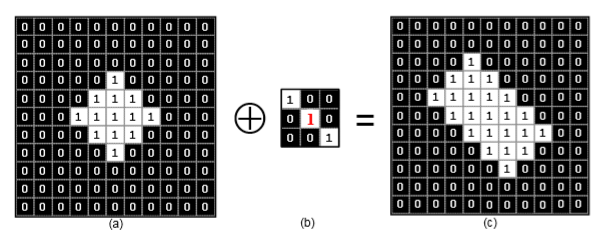
(Kadir, et al., 2013)

### Dilasi

Dilasi (*dilation*) adalah proses morfologi yang ditujukan untuk memperluas area atau ukuran objek (Madenda, 2015)**.** Proses dilasi dilakukan dengan membandingkan setiap piksel citra input dan nilai citra pusat SE dengan cara melapiskan(*superimpose*) SE dan citra sehingga pusat SE tepat terhadap posisi piksel citra yang diproses. Jika paling sedikit ada 1 piksel pada SE sama dengan nilai piksel objek (*foreground*) citra maka piksel input diset nilainya dengan nilai piksel *foreground* dan bila semua piksel yang berhubungan adalah *background* maka input piksel diberi nilai piksel *background.* Proses serupa dilanjutkan dengan translasi SE piksel demi piksel pada citra input. Semakin besar ukuran SE maka semakin besar perubahan yang terjadi. SE berukuran kecil juga dapat memberikan hasil yang sama dengan SE berukuran besar dengan cara melakukan dilasi berulang kali. Efek dilasi adalah dapat memperbesar batas dari objek sehingga telihat semakin besar dan lubang yang terdapat pada objek akan tampak mengecil (Putra, 2010). Berdasarkan operasi matematika dilasi A oleh B, didefenisikan sebagai berikut :

(26)

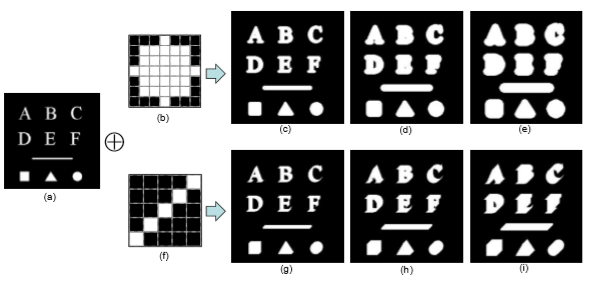
Berdasarkan rumus diatas A merupakan citra asli dan B merupakan *structure element,*



Gambar 2.13 Operasi dilasi

(Zhou, et al., 2010)

Dilasi A oleh B adalah himpunan yang terdiri semua piksel B yang dipantulkan dan menimpah sebagian piksel A dan titik B yang menimpah citra A akan menyebabkan penebalan pada citra A. Titik referensi *structure element* Byang menimpah piksel pada citra A memiliki nilai 1. Berdasarkan rumus, contoh dilasi sebagai berikut:



Gambar 2.14 Contoh Operasi Dilasi

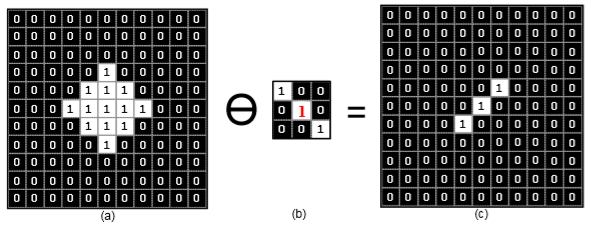
(Zhou, et al., 2010)

### Erosi

Erosi (*Erosion*) adalah proses morfologi yang ditujukan untuk mengikis atau memperkecil luas permukaan ukuran suatu area atau objek (Madenda, 2015)**.** Erosi dilakukan dengan membandingkan setiap piksel citra input dan nilai citra pusat SE dengan cara melapiskan(*superimpose*) SE dan citra sehingga pusat SE tepat terhadap posisi piksel citra yang diproses. Jika semua piksel pada SE tepat sama dengan semua nilai piksel objek(*foreground*) citra maka piksel input diset nilainya dengan nilai piksel *foreground,* bila tidak maka input piksel diberi nilai piksel *background.* Proses serupa dilanjutkan dengan translasi SE piksel demi piksel pada citra input. Proses erosi merupakan kebalikan dari prose dilasi. Jika dalam dilasi menghasilkan objek yang lebih luas maka dalam proses erosi akan menghasilkan objek yang mengecil. Lubang pada objek juga akan tampak membesar seiring mengecilnya batas pada objek tersebut (Putra, 2010).Berdasarkan operasi matematika, erosi A oleh B, didefenisikan sebagai berikut :

(27)

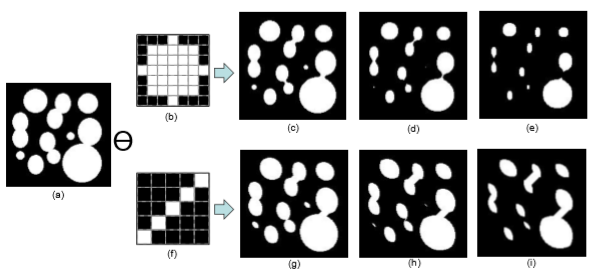
Berdasarkan rumus diatas A merupakan citra asli dan B merupakan *structure element,*



Gambar 2. 15 Operasi erosi

(Zhou, et al., 2010)

Erosi A oleh B adalah himpunan yang terdiri semua piksel B yang dipantulkan terhadap piksel A dan titik B yang menimpah citra A akan menyebabkan pengikisan terhadap citra A . Titik referensi *structure element* Byang menimpah piksel pada citra A tidak memiliki nilai sama dengan 0 maka nilai pada *structure element* B akan dieliminasi sehingga gambar pada citra asli akan semakin mengecil. Berdasarkan rumus, contoh erosi sebagai berikut:



Gambar 2. 16 Operasi erosi

(Zhou, et al., 2010)

## Deteksi Tepi

Deteksi tepi (*edge detection*) adalah suatu proses yang menghasilkan tepi dari objek citra. Dengan tujuan memperbaiki detail dari citra yang kabur,bisa terjadi karena *error* atau adanya efek dari akusisi citra (ALAT BANTU PEMBELAJARAN MATA KULIAH COMPUTER VISION PADA MATERI EDGE BASED SEGMENTASI CITRA BERBASIS MULTIMEDIA, 2013). Pada umumnya tepi objek merupakan pertemuan objek bagian latar belakang ditandai oleh titik yang nilai intensitas keabuan yang sangat berbeda dengan titik yang berada pada sebuah objek (Sutoyo, et al., 2009). Deteksi tepi bertujuan untuk memperoleh tepi objek. Deteksi tepi memanfaatkan perubahan nilai intensitas yang drastis pada batas dua area, tepi mengandung informasi sangat penting dan dapat berupa bentuk ataupun ukuran objek (Kadir, et al., 2013).

Deteksi tepi dibagi menjadi dua golongan. Golongan pertamadisebut deteksi tepi orde pertama, yang bekerja dengan menggunakan turunan atau difrensial orde pertama. Golongan kedua dinamakan deteksi tepi orde kedua. Termasuk dalam kedua kelompok ini ialah :

1. Operator *Roberts*

Operator *roberts* adalah operator pendeteksi tepi yang menggunakan kernel berukuran 2x2 piksel (Sutoyo, et al., 2009). Operator *roberts* memberikan tanggapan lemah terhadap tepi kecuali kalau tepi sangat tajam (Kadir, et al., 2013).

Dimisalkan berikut adalah posisi pada citra f

X x+1

|  |  |
| --- | --- |
|  |  |
|  |  |

Y

Y+1

Dimana nilai operator *roberts* dihitung menggunakan kernel konvolusi sebagai berikut :

|  |  |
| --- | --- |
|  | (28) |

Misalkan f adalah citra yang akan dikenai operator *roberts.* Maka, nilai operator *roberts* pada (y,x) didefinisikan sebagai

|  |  |
| --- | --- |
| r(y,x) | (29) |

Dalam hal ini = f(y,x), = f(y,x+1), =f(y+1,x), dan = f(y+1,x+1).

1. Operator *Prewitt*

Operator *prewitt* dikemukan oleh prewitt pada tahun 1966. Untuk mempercepat komputasi, bagian berniali nol tidak diproses. Oleh karena itu perhitungan operasi *prewitt* ditulis menjadi (Kadir, et al., 2013)

|  |  |
| --- | --- |
|  | (30) |

1. Operator *Sobel*

Operator *sobel* operator pendeteksi tepiyang berfungsi untuk menghitung gradien sehingga perkiraan gradien berada ditengah kernel dan menghindari adanya perhitungan gradien dititik interpolasi (Sutoyo, et al., 2009). Operator *sobel*  lebih sensitif terhadap tepi diagonal daripada tepi vertikal dan horizontal (Kadir, et al., 2013). Operator ini menggunakan kernel yang berukuran 3x3 piksel (x,y) sebagai berikut :

(31)

Secara matematis, gradien pada operator *sobel* dapat dihitung menggunakan persamaan sebagai berikut:

M= (32)

Dimana M adalah besar gradien di titik tengah kernel dan dan dapat dihitung menggunakan persamaan berikut:

= ( + ) - ( + ) (33)

= ( + ) - ( + ) (34)

Dimana c adalah konstanta yang bernilai 2. dan diimplementasikan menjadi kernel berikut:

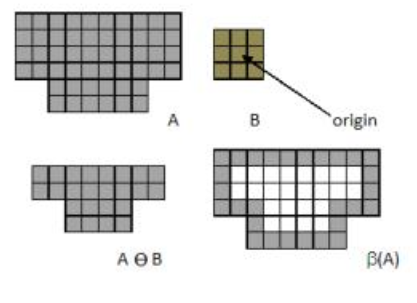
= = (35)

1. Boundary

*Boundary* pada gambar dapat dideteksi dengan mengurangi gambar yang terkikis dari citra utama. Batas ini digunakan kemudian dalam analisis kepala sperma (An efficient for automatic morphological abnormality detection from human sperm images, 2015). *Boundary* himpunan A dinyatakan oleh β(A), didapatkan dengan melakukan pengurangan A oleh B dan kemudian melakukan pengurangan antara A dan hasil erosi. Diformulasikan sebagai berikut (Prasetyo, 2011) :

β(A) = A- (A B) (36)

Dimana nilai B adalah strel yang tepat untuk mengerosi A. Ilustrasi *boundary* dapat dilihat sebagai berikut :



Gambar 2.17 Ilustrasi *boundary*

(Prasetyo, 2011)

1. Operator *Laplacian of Gaussian*

Deteksi tepi orde kedua yang makin kurang sensitif terhadap derau adalah *Laplacian of Gaussian.* Disebabkan penggunaan fungsi *gaussian* yang memuluskan citra dan berdampak terhadap pengurangan derau pada citra (Kadir, et al., 2013). Operator *Laplacian of Gaussian* diperoleh melalui konvolusi dengan :

LoG(y,x) = ] (37)

## Connected Component Labelling

Algoritma *connected component labelling* adalah metode yang dapat digunakan untuk mengklasifikasikan *region* atau objek dalam citra digital. Algoritma ini menerapkan teori *connectivity*. *Connected component labelling* yang digunakan adalah 4-*connected.* Citra yang dapat diolah dengan meggunakan algoritma *connected component labelling* ini adalah citra biner (Menghitung Obyek 2D Menggunakan Connected Component Labelling, 2017). *Connected component labeling* (CCL) digunakan untuk menetapkan label unik untuk semua piksel dari setiap komponen yang terhubung dalam gambar biner. *Connected component labeling* sangat diperlukan untuk membedakan objek yang berbeda dalam gambar biner, dan prasyarat untuk analisis gambar dan pengenalan objek dalam gambar. Oleh karena itu, *connected component labeling* adalah salah satu proses terpenting untuk analisis gambar, pemahaman gambar, pengenalan pola, dan visi komputer. Algoritma pelabelan, digunakan untuk gambar n-D (The connected-component labeling problem: Areview of state-of-the-art algorithm, 2017).

Connected Component Labeling, memindai gambar, dan mengelompokkan pikselnya ke dalam komponen berdasarkan konektivitas piksel, yaitu semua piksel dalam komponen yang terhubung, berbagi nilai intensitas piksel yang serupa, dan dalam beberapa cara terhubung satu sama lain. Biasanya digunakan untuk merujuk pada tugas pengelompokan piksel yang terhubung dalam suatu gambar. Mengekstraksi dan memberi label berbagai komponen yang terpisah dan terhubung dalam suatu gambar, merupakan pusat dari banyak aplikasi analisis gambar otomatis (New Algorithm - Simulation Connected Components Labeling for Binary Images., 2013). Berikut algoritma *connected component labelling* (Menghitung Obyek 2D Menggunakan Connected Component Labelling, 2017).

Terlebih dahulu mempersiapkan 2 array misalnya array *Connected Component* (CC) dan array Temp. Array CC adalah *nested array* yang berfungsi menyimpan array Temp yang menyimpan indeks-indeks *pixel* yang saling bertetangga dan membentuk satu objek. Jadi jumlah objek yang dapat dideteksi pada sebuah citra diambil dari jumlah elemen array CC. sedangkan kumpulan indeks yang merupakan jumlah *pixel* yang membentuk satu objek disimpan pada array Temp. Sehingga dapat dikatakan saru array Temp mewakili satu objek.

Langkah 1 :

* Cek piksel pada array biner secara berurutan
* Jika piksel bernilai *foreground* atau 1 maka ambil indeks piksel tersebut
* Cek piksel berikutnya

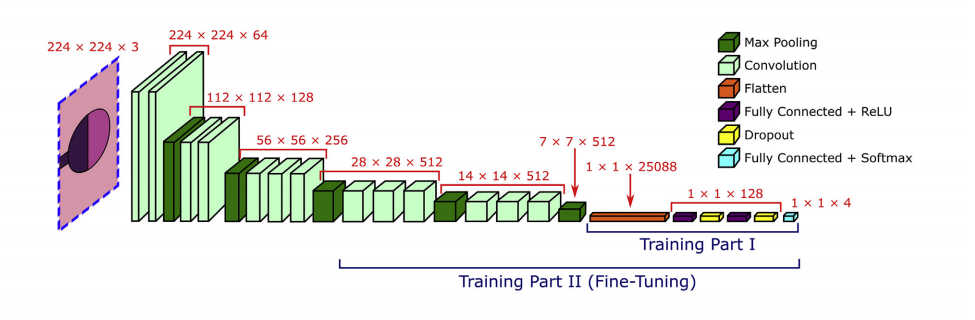
Langkah 2 :

* Buat array baru, misal bernama Temp
* Tambahkan indeks piksel yang diperoleh pada langkah pertama sebagai elemen array Temp.
* Ubah nilai piksel dari 1 ke 0 pada array biner citra (agar tidak terbaca ulang)
* Cek semua 4 tetangga dari piksel yang tersimpan pada array Temp,
* Jika nilai dari piksel tersebut 1 maka tambahkan indeks tetangga tersebut ke array Temp
* Ubah piksel tetangga dan inti ke 0 pada array biner citra
* Bila tidak ditemukan lagi piksel bernilai 1 dari 4 tetangga dari setiap indeks pada array Temp, maka tambahkan atau simpan array Temp ke array *Connected Component*
* Kembali ke langkah pertama poin ketiga

Jika sudah tidak ditemukan lagi piksel-piksel tetangga yang terdapat kedekatan secara 4-connected neighbors, maka lakukan proses merging pada matrix mapping. Periksa kembali piksel pada matrix original dengan melakukan scanning piksel – piksel citra foreground, jika tidak ditemukan, maka selesailah labeling dan merging (Iqbal, 2014).

## Convolution Neural Network

*Convolutional Neural Network* (CNN) merupakan pengembangan dari *multilayer perceptron* (MLP) yang didesain untuk mengolah data dua dimensi dalam bentuk citra. CNN ini termasuk kedalam jenis *Deep Neural Network* karena kedalaman jaringan yang tinggi dan banyak diaplikasikan pada data citra (Klasifikasi Citra Menggunakan Convolutional Neural Network (CNN) pada Caltech 101, 2016). CNN adalah sebuah arsitektur yang dapat dilatih dan terdiri dari beberapa tahap. Masukan (*input*) dan keluaran (*output*) dari setiap tahap adalah terdiri dari beberapa *array* yang biasa disebut *feature map.* Setiap tahap terdiri dari tiga *layer* yaitu konvolusi, fungsi aktivasi *layer* dan *pooling layer*. Berikut adalah jaringan arsitektur *Convolutional Neural Network* (Nada, 2019):



Gambar 2.21Arsitektur *Convolutional Neural Network*

(Sumber : (Deep Learning For The Classification of Human Sperm , 2019))

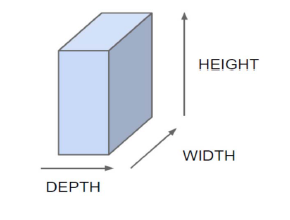
Berdasarkan gambar diatas, tahap pertama pada arsitektur CNN adalah tahap konvolusi. Tahap ini dilakukan dengan menggunakan sebuah kernel dengan ukuran tertentu. Perhitungan jumlah kernel yang dipakai tergantung dari jumlah fitur yang dihasilkan. Lapisan konvolusi melakukan operasi konvolusi pada data input dan menghasilkan output ke lapisan berikutnya setiap simpul hanya menerima sebagian kecil dari lapisan sebelumnya yang memungkinkan input gambar besar untuk diproses secara efektif. Lapisan – lapisan awala ini untuk melakukan ekstrasi fitur dan serangkaian lapisan fully connected berada pada bagian akhir dari jaringan yang melakukan klasifikasi. Kemudian dilanjutkan menuju fungsi aktivasi, biasanya menggunakan fungsi aktivasi ReLU ( *Rectifier Linear Unit* ), Selanjutnya setelah keluar dari proses fungsi aktivasi kemudian melalui proses *pooling*. Proses ini diulang beberapa kali sampai didapatkan peta fitur yang cukup (Deep Learning For The Classification of Human Sperm , 2019).

Langkah-langkah pengerjaan algoritma CNN , yaitu :

1. Berikan input gambar ke dalam lapisan konvolusi
2. Pilih parameter, terapkan filter dengan langkah, padding jika perlu. Lakukan konvolusi pada gambar dan terapkan aktivasi ReLU ke matriks.
3. Lakukan pooling untuk mengurangi ukuran dimensi
4. Tambahkan sebanyak mungkin lapisan konvolusional sampai
5. Ratakan output dan masukkan ke dalam lapisan yang terhubung penuh (FC Layer)
6. Keluarkan kelas menggunakan fungsi aktivasi (Regresi Logistik dengan fungsi biaya) dan mengklasifikasikan gambar (Prabhu, 2018).

### Convolution Layer

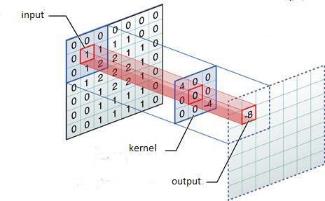
Konvolusi adalah istilah matematis dimana pengaplikasian sebuah fungsi pada *output* fungsi lain secara berulang. Operasi konvolusi merupakan operasi pada dua fungsi argumen bernilai nyata. Operasi ini menerapkan fungsi *output* sebagai *Feature Map* dari input citra (Understanding of a Convolutional Neural Network, 2017). Representasi CNN dapat dituliskan sebagai berikut :

****

Gambar 2.22 Inputan 3 dimensi Representasi dari CNN

(Understanding of a Convolutional Neural Network, 2017)

Operasi konvolusi menggunakan *filter* yang juga disebut sebagai *kernel* untuk mendeteksi fitur- fitur, seperti sudut yang ada di seluruh gambar. *Filter* hanyalah sebuah matriks nilai yang disebut bobot, dilatih untuk mendeteksi fitur tertentu. *Filter* bergerak di setiap bagian gambar untuk memeriksa apakah fitur yang dimaksudkan untuk dideteksi tersedia. Untuk memberikan nilai yang menunjukkan seberapa tepat fitur yang tersedia tersebut melakukan operasi konvolusi (Escontrela, 2018).



Gambar 2.23 Operasi Konvolusi pada Tahap *Convolution Layer*

(Escontrela, 2018)

Operasi konvolusi dapat dituliskan sebagai berikut :

s(t) = (x\*t) (t) = ∑ 𝑥(α) ∗ w(t − α) ∞ α=−∞ (38)

Keterangan :

S(t) = Fungsi hasil operasi konvolusi

X = Input 33

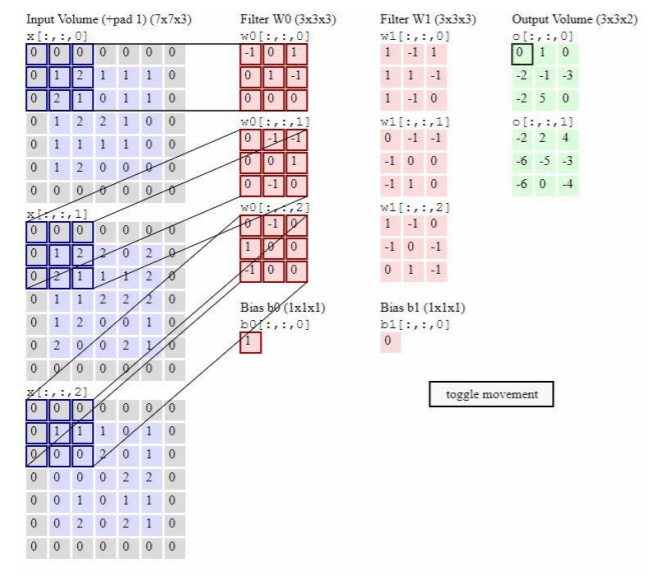
W = bobot (kernel)

Fungsi s(t) memberikan output tunggal berupa feature Map. Argumen pertama adalah input yang merupakan x dan argumen kedua w sebagai kernel atau filter. Apabila dilihat input sebagai citra dua dimensi, maka bisa dikatakan t sebagai piksel dan menggantinya dengan i dan j. Maka dari itu, operasi untuk konvolusi ke input dengan lebih dari satu dimensi dapat menulis sebagai berikut (Nada, 2019):

s(i,j) = (K\*I) (i,j) = ∑∞ ∑∞ 𝐼(𝑖 − 𝑚,𝑗 − 𝑛)𝐾(𝑚, 𝑛) (39)

s(i,j) = (K\*I) (i,j) = ∑∞ ∑∞ 𝐼(𝑖 + 𝑚,𝑗 + 𝑛)𝐾(𝑚, 𝑛) (40)

Berdasarkan kedua persamaan diatas merupakan perhiutngan dasar dalam opersai konvolusi, dengan i dan j adalah sebuah piksel dari citra. Perhitungan tersebut bersifat komulatif dan muncu saat K sebagai kernel, kemudian I sebagai input dan kernel yang dapat dibalik relatif terhadap input (Nada, 2019).



Gambar 2.24 Convolution Layer

(Sena, 2017)

Keterangan :

1. Cara Kerja Layar Convolutional Pertama

Gambar (matrix dengan nilai-nilai pixel) dimasukkan kedalam layer pertama. Pembacaan matriks. masukan dimulai di bagian atas kiri gambar. Berikutnya perangkat lunak memilih matriks yang lebih kecil, yang disebut filter. Kemudian filter menghasilkan layer, bergerak sepanjang gambar yang dimasukkan. Filter bertugas memperbanyak nilai-nilai piksel yang asli (Nada, 2019).

1. Cara kerja Layar Convolutional Kedua

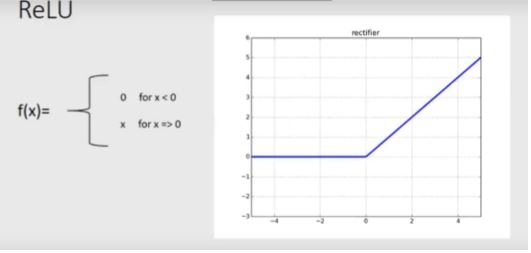
Layer convolution kedua memiliki fungsi aktivasi.

### Fungsi Aktivasi ReLu

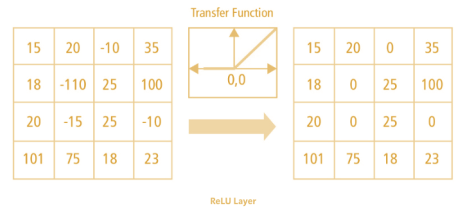
Fungsi aktivasi merupakan salah satu yang penentu performa dari neural network. Ada beberapa jenis fungsi aktivasi yang ada, dalam penelitian ini fungsi aktivasi yang digunakan adalah ReLU . ReLU (Rectification linear unit ) merupakan operasi untuk mengenalkan nonlinearitas dan meningkatkan representasi dari model. Apabila mendapat input positif, maka turunannya adalah nilai input aktivasi itu sendiri, dengan kata lain aktivasi hanya men-threshold pada nilai 0 (IMPLEMENTASI METODE CONVOLUTIONAL NEURAL NETWORK UNTUK KLASIFIKASI TANAMAN PADA CITRA RESOLUSI TINGGI , 2018). Adapun rumus dari ReLU adalah

𝑓𝑥= max (𝑥,0) . (41)

Pada arsitektur CNN, fungsi aktivasi terletak pada perhitungan akhir keluaran feature map atau sesudah proses perhitungan konvolusi atau pooling untuk menghasilkan suatu pola fitur (Convolutional Neural Networks untuk Pengenalan Wajah Secara Real-Time , 2016) :



Gambar 2. 25 *Aktivitas ReLU*



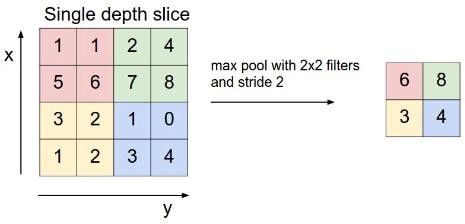
Gambar 2.26 ReLu Operation

(Prabhu, 2018)

### Operasi Max-Pooling

*Pooling* merupakan pengurangan ukuran matriks dengan menggunakan operasi *pooling. Pooling Layer* biasanya berada setelah *convolution layer*. Pada dasarnya *pooling layer* terdiri dari sebuah filter dengan ukuran dan *stride* tertentu yang akan secara bergantian bergeser pada seluruh area *feature map.* Dalam *pooling layer* terdapat dua macam *pooling* yang biasa digunakan yaitu *average pooling* dan *max- pooling*. Nilai yang diambil pada *average pooling* adalah nilai rata-rata, sedangkan pada *max-pooling* adalah nilai maksimal.

Lapisan *Pooling* yang dimasukkan diantara lapisan konvolusi secara berturut-turut dalam arsitektur model CNN dapat secara progresif mengurangi ukuran volume output pada *Feature Map*, sehingga mengurangi jumlah parameter dan perhitungan di jaringan, untuk mengendalikan *Overfitting*. Lapisan pooling bekerja di setiap tumpukan *feature map* dan melakukan pengurangan pada ukurannya. Bentuk lapisan *pooling* umumnya dengan menggunakan filter dengan ukuran 2x2 yang diaplikasikan dengan langkah sebanyak dua dan beroperasi pada setiap irisan dari inputnya. Berikut ini adalah contoh gambar operasi *max-pooling* (Understanding of a Convolutional Neural Network, 2017):



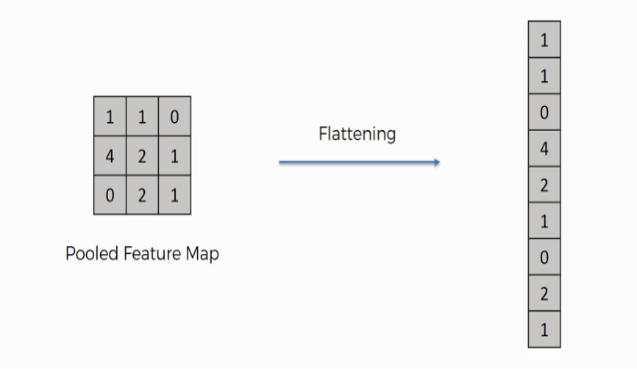
Gambar 2. 27Operasi *Max-Pooling*

(Understanding of a Convolutional Neural Network, 2017)

Berdasarkan gambar diatas menunjukan proses dari *max-pooling*. *Output* dari proses *pooling* adalah sebuah matriks dengan dimensi yang lebih kecil dibandingkan dengan citra awal. Lapisan pooling diatas akan beroperasi pada setiap irisan kedalaman volume input secara bergantian. Jika dilihat dari gambar diatas operasi *max-pooling* dengan menggunakan ukuran filter 2x2. Masukan pada proses tersebut berukuran 4x4, dari masing-masing 4 angka pada input operasi tersebut diambil nilai maksimalnya kemudian dilanjutkan membuat ukuran *output* baru menjadi ukuran 2x2 (Understanding of a Convolutional Neural Network, 2017).

### Fully-Connected Layer

*Fully-Connected Layer* dari *neural network* adalah proses dimana matriks masukan di-*flatten* menjadi fitur vektor dan melewati jaringan *neuron* untuk memprediksi probabilitas keluaran. Gambar berikut ini menjelaskan operasi *flatten* (Escontrela, 2018).

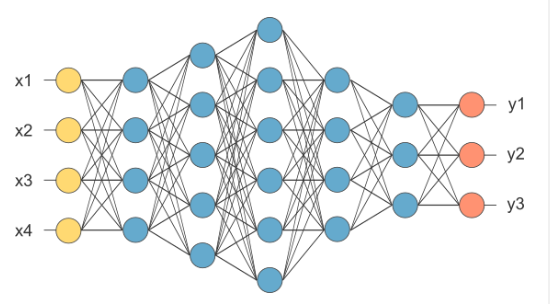


Gambar 2.28 Flatten

(Sumber: (Escontrela, 2018))

Baris digabungkan untuk membentuk fitur vektor yang panjang. Jika terdapat beberapa *layer* masukan, barisnya juga digabungkan untuk membentuk fitur vektor yang lebih panjang. Fitur vektor kemudian melewati beberapa *layer* yang padat. Pada setiap layer yang padat, fitur vektor dikalikan dengan bobot *layer*, dijumlahkan dengan *bias-*nya, dan melewati fungsi aktivasi (Escontrela, 2018).

*Layer* keluaran dari CNN (*Convolutional Neural Network*) bertugas menghasilkan probabilitas setiap kelas (setiap digit) yang diberikan masukan gambar. Untuk mendapatkan probabilitas ini, *layer* yang padat terakhir diinisialisasikan agar mengandung jumlah *neuron* yang sama dengan kelas yang ada. Keluaran dari *layer* yang padat ini kemudian melewati fungsi *Softmax Activation*, yang memetakan semua keluaran *layer* yang padat terkahir ke vektor yang unsur-unsurnya berjumlah satu (Escontrela, 2018).



Gambar 2. 29 Fully Connected Layer

(Prabhu, 2018)

### Evaluasi

Untuk mengetahui performa yang dihasilkan oleh metode dalam mengklasifikasi *sperm head* manusia maka perlu dilakukan evaluasi untuk mengukur kinerja pada sistem. empat metrik dihitung: *True Positive* (TP) berarti sistem memprekdisi jumlah positif yang diidentifikasi dengan benar sebagai positif, *True Negative* (TN) berarti sistem memprekdisi dengan benar yang bernilai negatif, *False Negative* (FN) berarti sistem memprekdiksi negative dan memiliki hasil yang salah, *False Positive* (FP) berarti ketika sistem memprediksi positif dan hasilnya salah. Positif (P) menunjukkan contoh tanpa masalah, dan negatif (N) menunjukkan contoh bermasalah. sebagai hasilnya, metrik yang disebutkan dapat dianggap sebagai (An Efficient Method For Automatic Morphological Abnormality Detection From Human Sperm Images, 2015).

Ada beberapa perhitungan yang dapat digunakan sebagai pengujian pada sistem diantaranya (Detection of Affected Part of Plant Leaves and Classification of Diseases Using CNN Technique, 2018):

* 1. *Accuracy*

*Accuracy* digunakan untuk mengukur tingkat kedekatan pengukuran antara nilai asli dengan nilai yang terdeteksi.

*Accuracy =*  (42)

Keterangan:

TP = *True Positive*

FN = *False Negative*

TN = *True Negative*

FP = *False Positive*

* 1. *Precision*

*Precision* digunakan untuk menentukan tingkat ketepatan antara informasi yang diminta pengguna dengan jawaban yang diberikan oleh sistem.

*Precision =* (43)

* 1. *Recall*

*Recall* digunakan untuk mengukur tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi.

*Recall =*  (44)



# BAB III ANALISIS DAN PERANCANGAN



## Analisis

### Analisis Proses p

#### Konversi Citra Grayscale

Dalam *image processing* banyak dilakukan konversi citra berwarna menjadi grayscale. Tujuan dari konversi citra ini adalah untuk memperkecil memori yang dibutuhkan sehingga dapat mempercepat proses yang dilakukan. Proses ini sangat membantu dalam pemrograman karena dapat memanipulasi bit yang tidak terlalu banyak. Citra berwarna terdiri dari 3 layer matrik yaitu R-layer, G-layer, dan B-layer. Untuk mengubah citra berwarna yang mempunyai nilai matrik masing-masing r, g, dan b menjadi gray scale dengan nilai s, maka konversi dapat dilakukan dengan mengambil rata-rata dari nilai r, g, dan b sehingga dapat dituliskan menjadi :

dimana :

I (i,,j) = Nilai intensitas citra grayscale

R (i,j) = nilai intensitas warna merah dari citra asal

G (i,j) = nilai intensitas warna hijau dari citra asal

B (i,j) = nilai intensitas warna biru dari citra asal

Flowchart :>>>>>>

Berikut contoh dari proses konversi citra RGB menjadi citra *grayscale.* Asumsikan citra adalah 8x8 *pixel* dengan nilai RGB nya sebagai berikut :



*Matrix* citra asli

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| R = 181 | R = 181 | R = 165 | R = 165 | R = 156 | R = 156 | R = 156 | R = 156 |
| G = 57 | G = 74 | G = 57 | G = 57 | G = 49 | G = 49 | G = 49 | G = 49 |
| B = 115 | B = 115 | B = 107 | B = 107 | B = 99 | B = 99 | B = 99 | B = 99 |
| R = 181 | R = 173 | R = 165 | R = 165 | R = 165 | R = 156 | R = 156 | R = 156 |
| G = 57 | G = 66 | G = 57 | G = 57 | G = 57 | G = 49 | G = 49 | G = 49 |
| B = 115 | B = 107 | B = 107 | B = 107 | B = 107 | B = 99 | B = 99 | B = 99 |
| R = 173 | R = 165 | R = 156 | R = 165 | R = 148 | R = 140 | R = 140 | R = 140 |
| G = 66 | G = 57 | G = 49 | G = 57 | G = 41 | G = 33 | G = 33 | G = 33 |
| B = 107 | B = 99 | B = 90 | B = 99 | B = 90 | B = 82 | B = 82 | B = 82 |
| R = 206 | R = 189 | R = 165 | R = 165 | R = 148 | R = 148 | R = 132 | R = 132 |
| G = 99 | G = 82 | G = 57 | G = 57 | G = 41 | G = 41 | G = 24 | G = 24 |
| B = 140 | B = 123 | B = 99 | B = 99 | B = 90 | B = 90 | B = 74 | B = 74 |
| R = 239 | R = 231 | R = 206 | R = 198 | R = 189 | R = 173 | R = 132 | R = 140 |
| G = 132 | G = 123 | G = 99 | G = 90 | G = 82 | G = 66 | G = 49 | G = 33 |
| B = 173 | B = 165 | B = 140 | B = 132 | B = 123 | B = 107 | B = 82 | B = 74 |
| R = 247 | R = 239 | R = 231 | R = 231 | R = 222 | R = 206 | R = 173 | R = 173 |
| G = 140 | G = 132 | G = 123 | G = 123 | G = 115 | G = 99 | G = 90 | G = 66 |
| B = 181 | B = 173 | B = 165 | B = 165 | B = 156 | B = 140 | B = 123 | B = 107 |
| R = 255 | R = 255 | R = 247 | R = 247 | R = 239 | R = 239 | R = 239 | R = 231 |
| G = 156 | G = 156 | G = 148 | G = 148 | G = 140 | G = 132 | G = 132 | G = 123 |
| B = 173 | B = 173 | B = 165 | B = 165 | B = 156 | B = 173 | B = 173 | B = 165 |
| R = 255 | R = 255 | R = 255 | R = 255 | R = 255 | R = 255 | R = 255 | R = 247 |
| G = 156 | G = 156 | G = 156 | G = 156 | G = 156 | G = 148 | G = 148 | G = 140 |
| B = 173 | B = 173 | B = 173 | B = 173 | B = 173 | B = 189 | B = 189 | B = 181 |

Pada *matrix* citra asli diatas *pixel* ke -1 dengan nilai R = 181, G = 57 dan B = 115. Dengan menggunakan rumus *average* untuk mengubah citra RGB menjadi *grayscale* dan perhitungan dilakukan untuk semua *pixel* dengan hasil konversi *matrix* sebagai berikut :

Citra Grayscale

3.1.1.2 Perbaikan citra

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

Flowchart>>>>>>>>>>>

*Matrix* citra *grayscale* yang akan dilakukan perbaikan

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

Dilakukan penentuan *mask filter*,penentuan *mask filter* misalnya dengan ukuran *matrix* 3 x 3. Karena ukuran *mask filter* 3 x 3 maka *matrix* diatas harus diurutkan terlebih dahulu dan dimasukkan kedalam *matrix* berukuran 1 x 9.

*Mask* *filter* 3 x 3 setelah diurutkan ke – 1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 110 | 110 | 115 | 115 | 118 | 118 | 123 |

Dengan menggunakan rumus *median filtering* pada persamaan (…..) sebagai berikut didapat nilai tengah

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

Dari gambar diatas dapat dicari nilai *pixel* yang baru dengan menggunakan rumus *median filtering*. Maka nilai mediannya adalah . Nilai *pixel*  yang baru akan menggantikan nilai *pixel* pada citra awal sehingga warna pada *pixel* sebelumnya akan berbeda. Karena nilai *median* sama dengan posisi tengah pada *mask filter* maka tidak terjadi perubahan dan proses perhitungan *median filtering* menggunakan citra awal.

*Mask* *filter* 3 x 3 setelah diurutkan ke – 2

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 107 | 110 | 110 | 110 | 110 | 115 | 123 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke - 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 3

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 93 | 98 | 101 | 107 | 110 | 110 | 110 | 110 | 110 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke - 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 4

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 93 | 101 | 101 | 101 | 107 | 110 | 110 | 110 |

, pada posisi ke – 4 didapatkan nilai tengahnya

*Mask median filter* ke - 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 5

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 85 | 93 | 101 | 101 | 101 | 101 | 101 | 110 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke - 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 6

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 85 | 85 | 101 | 101 | 101 | 101 | 101 | 101 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 7

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 107 | 110 | 115 | 115 | 118 | 131 | 148 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 8

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 107 | 107 | 107 | 110 | 110 | 115 | 131 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 9

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 93 | 93 | 98 | 107 | 107 | 107 | 110 | 110 | 110 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 9

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 10

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 93 | 93 | 93 | 101 | 107 | 107 | 110 | 110 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke -11

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 85 | 85 | 93 | 93 | 93 | 101 | 101 | 110 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 12

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 77 | 85 | 85 | 85 | 93 | 101 | 101 | 101 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 13

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 107 | 115 | 131 | 148 | 148 | 173 | 181 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 14

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 98 | 107 | 107 | 107 | 107 | 131 | 140 | 148 | 173 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 14

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 15

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 93 | 93 | 98 | 107 | 107 | 107 | 131 | 140 | 148 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 15

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 16

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 85 | 93 | 93 | 93 | 107 | 107 | 115 | 131 | 140 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 16

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 17

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 85 | 85 | 88 | 93 | 93 | 93 | 115 | 131 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 17

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 18

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 77 | 82 | 85 | 85 | 85 | 88 | 93 | 115 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 18

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 19

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 107 | 131 | 148 | 148 | 173 | 173 | 181 | 181 | 189 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 19

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 20

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 107 | 107 | 131 | 140 | 148 | 173 | 173 | 173 | 181 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 20

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 21

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 93 | 107 | 107 | 131 | 140 | 148 | 164 | 173 | 173 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 21

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 22

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 93 | 93 | 107 | 115 | 131 | 140 | 148 | 164 | 173 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 22

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 23

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 88 | 93 | 93 | 115 | 129 | 131 | 148 | 164 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 23

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 24

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 77 | 77 | 82 | 88 | 93 | 115 | 115 | 129 | 148 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 24

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 25

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 148 | 173 | 173 | 181 | 181 | 187 | 189 | 195 | 195 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 25

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 26

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 140 | 148 | 173 | 173 | 173 | 181 | 187 | 187 | 195 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 26

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 27

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 131 | 140 | 148 | 164 | 173 | 173 | 178 | 187 | 187 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 27

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 28

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 115 | 131 | 140 | 148 | 164 | 173 | 178 | 187 | 181 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 28

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 29

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 88 | 115 | 129 | 131 | 148 | 164 | 178 | 181 | 181 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 29

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 30

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 82 | 88 | 115 | 115 | 129 | 148 | 173 | 181 | 181 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 30

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 31

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 173 | 181 | 187 | 189 | 195 | 195 | 195 | 195 | 195 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 31

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 32

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 173 | 173 | 181 | 187 | 187 | 195 | 195 | 195 | 195 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 32

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 33

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 164 | 173 | 173 | 178 | 187 | 187 | 195 | 195 | 195 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 33

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 34

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 148 | 164 | 173 | 178 | 181 | 187 | 195 | 195 | 197 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 34

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 35

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 129 | 148 | 164 | 178 | 181 | 181 | 195 | 197 | 197 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 35

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

*Mask* *filter* 3 x 3 setelah diurutkan ke - 36

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 115 | 129 | 148 | 173 | 181 | 181 | 189 | 197 | 197 |

, pada posisi ke – 5 didapatkan nilai tengahnya

*Mask median filter* ke – 36

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 110 | 101 | 101 | 101 |
| 115 | 107 | 98 | 107 | 93 | 85 | 85 | 85 |
| 148 | 131 | 107 | 107 | 93 | 93 | 77 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 88 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 178 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

Hasil *Median Filter*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 118 | 123 | 110 | 110 | 101 | 101 | 101 | 101 |
| 118 | 115 | 110 | 110 | 101 | 101 | 101 | 101 |
| 115 | 115 | 107 | 107 | 101 | 93 | 85 | 85 |
| 148 | 131 | 107 | 107 | 107 | 93 | 85 | 77 |
| 181 | 173 | 148 | 140 | 131 | 115 | 93 | 82 |
| 189 | 181 | 173 | 173 | 164 | 148 | 129 | 115 |
| 195 | 195 | 187 | 187 | 181 | 181 | 181 | 173 |
| 195 | 195 | 195 | 195 | 195 | 197 | 197 | 189 |

Citra yang akan dilakukan *labelling*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

*Structure element* citra *labelling*

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 1 | 0 |

Cek *labelling* - 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 9

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 14

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 15

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 16

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* - 17

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke -18

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 19

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 20

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling*  ke - 21

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 22

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 23

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 24

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 25

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 26

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 27

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 28

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 29

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 30

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 31

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 32

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 33

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 34

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 35

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 36

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 37

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 38

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 39

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 40

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 41

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 42

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 43

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 44

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 45

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 46

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 47

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 48

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 49

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 50

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 51

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 52

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 53

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 54

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 55

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 56

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 57

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 58

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 59

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 60

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 61

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 62

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 63

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Cek *labelling* ke - 64

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Hasil *Labeling*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Citra yang akan dilakukan Deteksi Tepi menggunakan Sobel

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Deteksi sobel ke – 1

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Dari gambar diatas dapat dicari nilai *pixel* yang baru dengan mengkalikan persamaan rumus deteksi sobel, yaitu :

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

=

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

=   
Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

= ((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke - 2

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

= ((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke - 3

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

= ((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke - 4

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke - 5

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 6

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 7

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 8

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 9

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 10

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 11

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 12

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 13

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (1 x 2) + (0 x 1)) = 0 + 0 + 2 = 2

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (1 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 2 + 0 = 2

Deteksi Tepi Sobel

ke – 14

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 1 = 1

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 1 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 1 + 1 = 2

Deteksi Tepi Sobel

ke – 15

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 16

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 17

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 18

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 19

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (1 x 0) + (0 x 0)) + ((0 x 1) + (1 x 2) + (1 x 1)) = 0 + 2 + 1 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (1 x 0) + (0 x -2)) + ((0 x 1) + (1 x 0) + (1 x - 1)) = 0 + 0 -1 = -1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 3 + (-1) = > untuk nilai (–) akan di absolutkan

= 3 + 1 = 4

Deteksi Tepi Sobel

ke – 20

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 21

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 0 + 1 = 1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 4

Deteksi Tepi Sobel

ke – 22

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (0 x 1)) = 0 + 0 +3= 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 1 + 0 + 0 = 1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 4

Deteksi Tepi Sobel

ke – 23

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 1 = 1

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0+ 1 = 1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 2

Deteksi Tepi Sobel

ke – 24

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((0 x 0) + (0 x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((0 x 2) + (0 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Deteksi Tepi Sobel

ke – 25

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 0 |
| 0 | 1 | 1 |
| 0 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 0 |
| 0 | 1 | 1 |
| 0 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 26

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 27

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 28

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 29

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 30

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (0 x 0) + (0 x 0)) + ((1 x 1) + (1 x 2) + (1 x 1)) = 0 + 0 + 3 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 1 | 1 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (0 x 0) + (0 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 0 + 2 + 1 = 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobel

ke – 31

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (1 x -2) + (1 x -1)) + ((0 x 0) + (1x 0) + (1x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + (-3) + 0 = - 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 1 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (1 x 0) + (1 x -1)) + ((1 x 2) + (1 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = -1 + 2 + 1 = - 3

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 6

Deteksi Tepi Sobelke – 32

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((1 x -1) + (1 x -2) + (1 x -1)) + ((1 x 0) + (1 x 0) + (1 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = -4 + 0 + 0 = -4

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((1 x 1) + (1 x 0) + (1 x -1)) + ((1 x 2) + (1 x 0) + (1 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = -4

Deteksi Tepi Sobel

ke – 33

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((1 x -1) + (1 x -2) + (1 x -1)) + ((1 x 0) + (1 x 0) + (1 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = -4 + 0 + 0 = -4

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 1 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((1 x 1) + (1 x 0) + (1 x -1)) + ((1 x 2) + (1 x 0) + (1 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = -4

Deteksi Tepi Sobel

ke – 34

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((1 x -1) + (1 x -2) + (0 x -1)) + ((1 x 0) + (1 x 0) + (1 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = -3 + 0 + 0 = 3

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((1 x 1) + (1 x 0) + (0 x -1)) + ((1 x 2) + (1 x 0) + (1 x -2)) + ((1 x 1) + (1 x 0) + (1 x - 1)) = 1 + 2 + (-2) = 1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 3 + 1 = 4

Deteksi Tepi Sobelke – 35

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((1 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (1 x 0) + (1x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = -1 + 0 + 0 = -1

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |
| 1 | 1 | 1 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((1 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (1 x 0) + (1 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 1 + 2 + (-2) = 1

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 2

Deteksi Tepi Sobelke – 36

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Persamaan pada kernel Sx :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| -1 | 0 | 1 |
| -2 | 0 | 2 |
| -1 | 0 | 1 |

Χ

= ((0 x -1) + (0 x -2) + (0 x -1)) + ((1 x 0) + (1x 0) + (0 x 0)) + ((0 x 1) + (0 x 2) + (0 x 1)) = 0 + 0 + 0 = 0

Persamaan pada kernel Sy :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 1 | 1 | 0 |
| 0 | 0 | 0 |

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 1 |
| 0 | 0 | 0 |
| -1 | -2 | -1 |

Χ

((0 x 1) + (0 x 0) + (0 x -1)) + ((1 x 2) + (1 x 0) + (0 x -2)) + ((0 x 1) + (0 x 0) + (0 x - 1)) = 0 + 0 + 0 = 0

Untuk mendapatkan kernel baru dengan menjumlahkan nilai dan :

+ = 0

Hasil Deteksi Tepi Sobel

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 2 | 2 | 0 | 0 | 0 | 0 | 0 |
| 0 | 4 | 6 | 4 | 4 | 2 | 0 | 0 |
| 0 | 4 | 4 | 4 | 6 | 6 | 4 | 0 |
| 0 | 6 | 4 | 4 | 4 | 2 | 2 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

# BAB IV HASIL DAN PENGUJIAN



# BAB V KESIMPULAN DAN SARAN



## Kesimpulan

## Saran



Adapun saran yang dapat diberikan adalah sebagai berikut :

# DAFTAR PUSTAKA

**DAFTAR RIWAYAT HIDUP**

Nama :

Umur :

Tempat / Tanggal Lahir :

Jenis Kelamin :

Agama :

Tempat Tinggal :

Pendidikan :

Demikian daftar riwayat hidup ini saya perbuat dengan sesungguhnya.

|  |  |  |
| --- | --- | --- |
|  |  | Hormat saya, |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |