JOINT

Journal of Informatics and Technology, Vol 2, No 4, Tahun 2013, p 1-9
http://ejournal-s1.undip.ac.id/index.php/joint

APLIKASI PENDETEKSI OBJEK LINGKARAN PADA CITRA DENGAN TRANSFORMASI HOUGH

Lutfi Rinanto [1], Aris Sugiharto [2], Indriyati [3]

Jurusan Ilmu Komputer / Informatika Fakultas Sains dan Matematika Universitas Diponegoro

Abstrak

Terdapat berbagai bangun datar seperti segitiga, segiempat, segilima, segienam dan lingkaran. Diantara bangun datar tersebut, lingkaran merupakan bangun datar yang berbeda dengan bangun datar lainnya karena merupakan kurva tertutup yang memiliki jari-jari konstan. Dalam kehidupan nyata banyak objek yang dibentuk dengan dasar lingkaran, seperti rambu-rambu lalu lintas, uang logam, bola, bahkan di dalam organ tubuh manusia seperti iris mata dan sel darah merah. Oleh karena itu, dibutuhkan suatu aplikasi pendeteksi objek lingkaran agar dapat menjadi media yang bermanfaat baik dalam bidang pendidikan maupun kesehatan. Dalam mendeteksi suatu objek lingkaran diperlukan suatu metode yang efektif agar dapat diperoleh hasil yang akurat. Permasalahan yang muncul dalam melakukan proses pendeteksian objek lingkaran pada citra digital adalah bagaimana sebuah metode dapat mendeteksi berbagai objek lingkaran dengan ukuran yang berbeda. Transformasi Hough merupakan metode yang dapat digunakan untuk mendeteksi objek lingkaran pada citra digital dengan hasil yang akurat. Nilai rata-rata hasil proses pendeteksian adalah 94,25%.

Kata kunci: citra digital, deteksi, objek lingkaran, Transformasi Hough.

Abstract

There are various types of plane such as triangles, rectangles, pentagons, hexagons and circles. Among the plane, the circle is a different plane than other plane because it is a closed curve which has a constant radius. In real-life, there are many objects that formed the basis of the circle, such as traffic signs, coins, balls, even inside human organs such as iris and red blood cells. Therefore, required a circular object detection applications in order to be a useful media in both education and health. Detecting an object in a circle required an effective method in order to obtain accurate results. The problems that arise in the process of detection of circular objects in digital images is how a method can detect various types of circular objects of different sizes. Hough transformation is a method that can be used to detect the circular object in a digital image with accurate results. The average value of the results of the detection is 94.25%.

Keywords: digital image, detection, circular object, Hough Transformation.

1. Pendahuluan

Perkembangan teknologi komputer dewasa ini telah mengalami banyak perubahan yang sangat pesat, seiring dengan kebutuhan manusia. Kemajuan tersebut kini telah digunakan secara luas di berbagai bidang, seperti bisnis, kesehatan, pendidikan, dan pekerjaan. Dalam berbagai bidang tersebut, dibutuhkan sarana media yang baik dalam menunjang aktifitas sehari-hari dalam pengelolaan data teks, suara, citra dan video.

Dalam aktifitas sehari-hari, mata digunakan sebagai indra pengelihatan untuk melihat benda/objek berbentuk segiempat, segitiga, dan lingkaran. Lingkaran merupakan kurva tertutup yang memiliki sudut 360 derajat dengan ukuran jari-jari konstan. Dalam kehidupan sehari-hari banyak objek-objek yang dibentuk dengan dasar lingkaran seperti bola, koin, maupun sel darah merah. Oleh karena itu, dibutuhkan suatu sarana media yang bisa digunakan untuk mendeteksi objek-objek berbentuk lingkaran pada suatu citra digital.

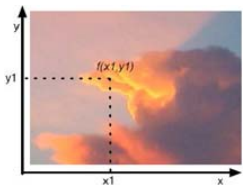
Transformasi Hough (TH) (Hough, 1962) merupakan teknik pengalokasian bentuk-bentuk dalam gambar. Secara khusus, transformasi ini digunakan untuk ekstraksi garis, lingkaran, dan elips. Implementasi TH menjelaskan sebuah pemetaan dari titik-titik gambar menuju sebuah ruang akumulator (ruang Hough). Pemetaan tersebut diperoleh dalam bentuk yang efisien secara matematis, berdasarkan fungsi yang menjelaskan kondisi dari target. TH akan mendeteksi lingkaran selama ada lebih banyak titik-titik di dalam daerah sirkular yang dapat dipaparkan dengan parameter-parameter dari lingkaran target dibandingkan dengan lingkaran-lingkaran yang lainnya. [3]

Pada tugas akhir ini diteliti dalam bidang pengolahan citra digital khususnya dalam bidang pengenalan pola untuk mendeteksi objek lingkaran pada citra dengan menggunakan metode Transformasi Hough. Metode Transformasi Hough digunakan pada sistem ini dengan harapan hasil yang dihasilkan akan akurat dan dapat menjadi suatu sarana media yang dapat dikembangkan menjadi suatu aplikasi yang lebih besar dan bermanfaat bagi masyarakat dalam berbagai bidang.

2. Dasar Teori

2.1. Citra Digital

Citra digital dapat didefinisikan sebagai fungsi dua variabel, f(x,y), dimana x dan y adalah koordinat spasial dan nilai f(x,y) adalah intensitas citra pada koordinat tersebut, hal tersebut diilustrasikan pada Gambar 2.1. Teknologi dasar untuk menciptakan dan menampilkan warna pada citra digital berdasarkan pada penelitian bahwa sebuah warna merupakan kombinasi dari tiga warna dasar, yaitu merah, hijau, dan biru.[9]



Gambar 2.1. Contoh Citra Digital data secara elektronik [1].

2.2. *Grayscalling*

Grayscalling adalah teknik yang digunakan untuk mengubah citra berwana (RGB) menjadi bentuk *grayscale* atau tingkat keabuan (dari hitam ke putih). Dengan pengubahan ini, matriks penyusun citra yang sebelumnya 3 matriks akan berubah menjadi 1 matriks saja. Pengubahan dari citra berwarna ke bentuk *grayscale* biasanya mengikuti aturan sebagai berikut: [6]

$$I(i,j) = \frac{R(i,j) + G(i,j) + B(i,j)}{3} \quad \text{.....2.1}$$

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

2.3. *Sobel Edge Detection*

Metode Sobel merupakan salah satu metode yang dapat digunakan untuk deteksi tepi pada suatu citra digital. Metode Sobel merupakan pengembangan dari metode robert dengan menggunakan filter HPF *(High Pass Filter)* yang diberi satu angka nol penyangga. Metode ini mengambil prinsip dari fungsi laplacian dan gaussian yang dikenal sebagai fungsi untuk membangkitkan HPF *(High Pass Filter)*. Kelebihan dari metode sobel ini adalah kemampuan untuk mengurangi noise sebelum melakukan perhitungan deteksi tepi. [2]

Operator Sobel menggunakan kernel operator gradient sumbu-x dan gradient sumbu-y 3 x 3 sebagai berikut:

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

Sumbu-X

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

Sumbu-Y

Operator Sobel melakukan deteksi tepi dengan memperhatikan tepi vertikal dan horizontal. Gradient Magnitude dari operator Sobel adalah sebagai berikut : [2]

$G_x = [f(i-1,j-1) + 2f(i-1,j) + f(i-1,j+1)] - [f(i+1,j-1) + 2f(i+1,,j) + f(i+1,j+1)]$.....2.2

$Gy = [f(i-1,j-1) + 2f(i,j-1) + f(i+1,j-1)] - [f(i-1,j+1) + 2f(i,j+1) + f(i+1,j+1)]$.....2.3

$$G[f(x,y)] = \sqrt{Gx^2 + Gy^2} \quad \text{.....2.4}$$

2.4. Transformasi Hough

Transformasi Hough (TH) (Hough, 1962) merupakan teknik pengalokasian bentuk-bentuk dalam gambar. Secara khusus, transformasi ini digunakan untuk ekstraksi garis, lingkaran, dan elips. TH kemudian diimplementasikan untuk menemukan garis-garis dalam gambar (Duda, 1972) dan kemudian meluas karena transformasi ini memiliki banyak kelebihan dan banyak potensi untuk pengembangan lebih lanjut. Kelebihan utamanya yaitu dapat memberikan hasil lebih cepat dan sama dengan pencocokan pola. (Princen, 1992), (Sklansky, 1978) (Stockman, 1977).

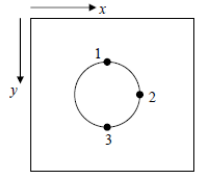
Implementasi TH menjelaskan sebuah pemetaan dari titik-titik gambar menuju sebuah ruang akumulator (ruang Hough). Pemetaan tersebut diperoleh dalam bentuk yang efisien secara matematis, berdasarkan fungsi yang menjelaskan kondisi dari target. Pemetaan ini membutuhkan jauh lebih sedikit sumber perhitungan matematis dibandingkan dengan pencocokan pola. Bagaimanapun, TH masih membutuhkan penyimpanan signifikan dan perhitungan matematis tingkat tinggi. Masalah-masalah ini diselesaikan kemudian, karena mereka memfokuskan untuk pengembangan TH secara kontinu. Bagaimanapun, fakta bahwa TH ekuivalen dengan pencocokan pola telah menjadikannya sebagai salah satu dari teknik-teknik ekstraksi bentuk yang terpopuler yang ada [12].

2.4.1. Transformasi Hough Lingkaran

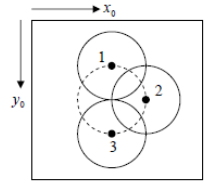
TH dapat diperluas dengan mengganti persamaan kurva pada proses pengenalan. Persamaan kurva dapat diberikan dalam bentuk eksplisit atau parametrik. Dalam bentuk eksplisit, TH dapat didefinisikan dengan mengingat persamaan untuk lingkaran sebagai berikut:

$$(x - x_0)^2 + (y - y_0)^2 = r^2 \quad \text{.....(2.5)}$$

Persamaan ini mendefinisikan posisi titik-titik (x, y) memiliki pusat di daerah (x_0,y_0) dan *radius r*. Persamaan ini dapat pula divisualisasikan dalam dua cara, yaitu sebagai posisi titik-titik (x,y) dalam sebuah gambar atau sebagai posisi titik-titik (x_0,y_0) yang berpusat pada (x,y) dengan *radius r* (pada gambar 2.2).



Gambar 2.2 Citra yang berisi lingakaran



Gambar 2.3. Ruang Akumulator

Persamaan tersebut mendefinisikan titik-titik pada ruang akumulator (pada gambar 2.3) bergantung pada nilai *radius r*. Jejak kurva (atau permukaan) umumnya dikenal sebagai *point spread function* (fungsi penyebaran titik).

Di masing-masing titik tepi, dapat menggambar sebuah lingkaran dengan titik tengah dan jari-jari lingkaran (radius). Lingkaran digambar dalam parameter yaitu sumbu x adalah nilai parameter x_0 dan sumbu y mewakili nilai parameter y_0 dan sumbu z mewaliki parameter radius. Pada koordinat yang memiliki perimeter dari gambar lingkaran, dapat diwakili pada matriks penjumlahan (akumulator) yang mempunyai ukuran yang sama sebagai *parameter space*. Cara ini men-scan setiap titik tepi dalam penggambaran image lingkaran dengan radius dapat meningkatkan nilai pada matrik akumulator.

Pada gambar ilustrasi dapat dilihat bahwa TH untuk lingkaran mampu mentoleransi gangguan-gangguan (noise). Perlu dicatat bahwa tidak lagi mendapati masalah pada penentuan titik awal dan akhir garis, karena lingkaran merupakan sebuah bangun tertutup. Pada contoh ketiga, terdapat banyak titik yang mengimplikasi adanya penambahan waktu proses. TH akan mendeteksi lingkaran selama ada lebih banyak titik-titik di dalam daerah sirkular yang dapat dipaparkan dengan parameter-parameter dari lingkaran target dibandingkan dengan lingkaran-lingkaran yang lainnya. Ini merupakan performa yang tepat sama dengan TH untuk garis, seperti yang diharapkan, dan juga konsisten dengan hasil dari pencocokan pola [12].

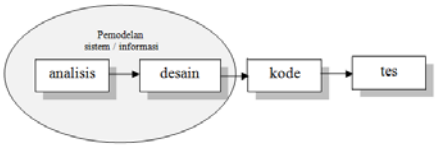
Transformasi Hough merupakan bagian segmentasi citra. Masalah umum yang sering

dihadapi dalam computer vision yaitu menentukan lokasi, nomor atau orientasi objek tertentu dalam foto. Hough transform umum yang dapat digunakan pada berbagai bentuk, meskipun kompleksitas dari transformasi dengan meningkatkan jumlah parameter yang dibutuhkan untuk menjelaskan bentuk.

2.5. Metode Pengembangan Aplikasi

Pengembangan aplikasi hampir menyerupai proses pengembangan aplikasi perangkat lunak umum, yaitu dimulai dari tahap analisis dan diakhiri dengan tahap pengujian. Pengembangan aplikasi ini menggunakan model pengembangan perangkat lunak *sekuensial linier*.

Model *sekuensial linear* terkadang disebut juga "siklus kehidupan klasik" atau "model air terjun". *Sekuensial linear* mengusulkan sebuah pendekatan pengembangan perangkat lunak yang sistematik dan sekuensial mulai dari analisis, desain, kode, pengujian, dan pemeliharaan [8]. Gambar 2.4 memberikan ilustrasi model sekuensial linier dalam pengembangan perangkat lunak.



Gambar 2.4. Model Sekuensial Linear

Dalam bukunya, Pressman menjelaskan bahwa model sekuensial linier meliputi aktifitas sebagai berikut [8]:

1. Analisis kebutuhan perangkat lunak

Untuk memahami sifat program yang dibangun, perekayasa perangkat lunak (analis) harus memahami domain informasi, tingkah laku, unjuk kerja, dan antar muka *(interface)* yang diperlukan. Kebutuhan baik untuk sistem maupun perangkat lunak didokumentasikan dan dilihat lagi dengan pelanggan.

2. Desain

Desain perangkat lunak sebenarnya adalah proses multi langkah yang berfokus pada empat atribut sebuah program yang berbeda, yaitu struktur data, arsitektur perangkat lunak, representasi interface, dan detail (algoritma) prosedural. Proses desain menerjemahkan syarat/kebutuhan ke dalam sebuah representasi perangkat lunak yang dapat diperkirakan demi kualitas sebelum dimulai pemunculan kode.

3. Generasi kode

Desain harus diterjemahkan kedalam bentuk mesin yang bisa dibaca. Langkah pembuatan kode melakukan tugas ini. Jika desain dilakukan dengan cara yang lengkap, pembuatan kode dapat diselesaikan secara mekanis.

4. Pengujian

Sekali kode dibuat, pengujian program dimulai. Proses pengujian berfokus pada logika internal perangkat lunak, memastikan bahwa semua pernyataan sudah diuji, dan pada eksternal fungsional yaitu mengarahkan pengujian untuk menemukan kesalahan-kesalahan dan memastikan bahwa input yang dibatasi akan memberikan hasil aktual yang sesuai dengan hasil yang dibutuhkan.

2.6. *MATLAB*

MATLAB *(Matrix Laboratory)* adalah sebuah program untuk analisis dan komputasi numerik dan merupakan suatu bahasa pemrograman matematika lanjutan yang dibentuk dengan dasar pemikiran menggunakan sifat dan bentuk matriks. MATLAB memiliki kemampuan mengintegrasikan komputasi, visualisasi, dan pemrograman [9]. Pada awalnya, program ini merupakan interface untuk koleksi rutin-rutin numerik dari proyek LINPACK dan EISPACK, dan dikembangkan menggunkan bahasa fortran namun sekarang merupakan produk komersial dari perusahaan *Mathworks Inc.* yang dalam perkembangan selanjutnya dikembangkan menggunakan bahasa C++ dan assembler (utamanya untuk fungsi-fungsi dasar MATLAB) [7].

MATLAB telah berkembang menjadi sebuah lingkup pemrograman yang canggih yang berisi fungsi-fungsi built-in untuk melakukan tugas pengolahan sinyal, aljabar linier, dan kalkulasi matematis lainnya. MATLAB juga berisi *toolbox* yang berisi fungsi - fungsi tambahan untuk aplikasi khusus. MATLAB bersifat *extensible*, dalam arti bahwa seorang pengguna dapat menulis fungsi baru untuk ditambahkan pada *library* ketika fungsi-fungsi *built-in* yang tersedia tidak dapat melakukan tugas tertentu. MATLAB *(Matrix Laboratory)* yang merupakan bahasa pemrograman tingkat tinggi berbasis pada matriks sering digunakan untuk teknik komputasi numerik, yang digunakan untuk menyelesaikan masalah-masalah yang melibatkan operasi matematika elemen, matrik, optimasi, aproksimasi, dan lain-lain.

Sehingga MATLAB banyak digunakan pada :
a. Matematika dan komputasi.
b. Pengembangan dan algoritma.
c. Pemrograman modeling, simulasi, dan pembuatan *prototype*.
d. Analisa data, eksplorasi dan visualisasi.
e. Analisis numerik dan statistik.
f. Pengembangan aplikasi teknik.

Fitur-fitur MATLAB sudah banyak dikembangkan, dan lebih dikenal dengan nama *toolbox*. Sangat penting bagi seorang pengguna Matlab, *toolbox* mana yang mandukung untuk *learn* dan *apply* technologi yang sedang dipelajarinya. *Toolbox-toolbox* ini merupakan kumpulan dari fungsi-fungsi MATLAB (M-files) yang telah dikembangkan ke suatu lingkungan kerja MATLAB untuk memecahkan masalah dalam kelas particular.

Area-area yang sudah bisa dipecahkan dengan *toolbox* saat ini meliputi pengolahan sinyal, sistem kontrol, *neural networks, fuzzy logic, wavelets*, dan lain-lain. Selain *toolbox*, matlab juga dilengakapi dengan *Simulink* yang sangat *powerfull* untuk mensimulasikan dan menyelesaikan masalah-masalah yang berhubungan dengan pemodelan matematika. Dengan *Simulink*, dapat dengan mudah mengecek kestabilan suatu model Matematika yang ada [6].

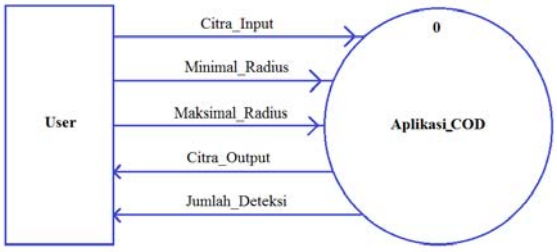
3. Analisis dan Perancangan

3.1. Definisi Kebutuhan Perangkat Lunak

Subbab ini dipaparkan mengenai definisi kebutuhan aplikasi. Kebutuhan atau *requirement* bertujuan untuk mendefinisikan lingkungan proyek, menangkap kebutuhan yang aplikasi terhadap *user*.

3.2. *Data Context Diagram*

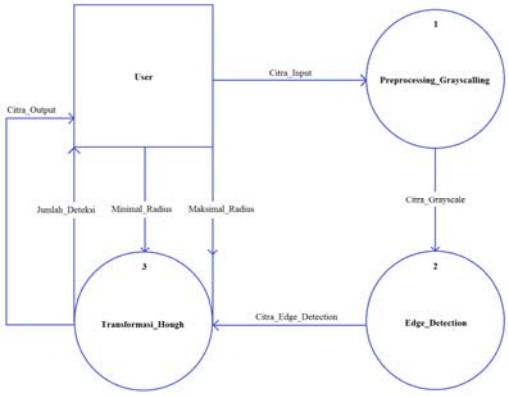
Gambar 3.1 adalah *data context diagram* pada aplikasi COD *(Circular Object Detection).*



Gambar 3.1. *Data Context Diagram*

3.3. *Data Flow Diagram Level I*

Data Flow Diagram Level 1 merupakan turunan *(breakdown)* dari *Data Context Diagram (DCD) / Data Flow Diagram Level 0* yang menjelaskan aliran data lebih rinci pada setiap proses, seperti pada gambar 3.2.



Gambar 3.5. DFD Level 1

Pada DFD level 1, dipecah menjadi 3 proses yang lebih rinci lagi yaitu :

a. ***Preprocessing Grayscalling***, yaitu proses untuk konversi citra RGB menjadi citra *grayscale*. Proses ini mendapat masukan berupa citra_input dari *user*. Keluaran dari proses ini yaitu berupa citra_grayscale.
b. ***Edge Detection***, yaitu proses deteksi tepi dengan menggunakan *Sobel Edge Detection*. Proses ini mendapat masukan berupa citra_grayscale yang didapat dari proses *preprocessing_grayscalling*. Keluaran dari proses ini yaitu berupa citra_edge_detection.
c. **Transformasi Hough**, yaitu proses pendeteksian objek lingkaran. Proses ini mendapat masukan citra_edge_detection dari proses *edge_detection*. Selain itu proses ini juga menerima 2 parameter masukan, yaitu minimal_radius dan maksimal_radius dari pengguna. Proses ini memberikan keluaran berupa citra_output dan jumlah_deteksi kepada *user*.

3.4. Perancangan sistem

Subbab ini dipaparkan mengenai perancangan proses/*flowchart* dari aplikasi COD.

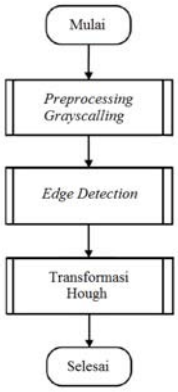
3.4.1. Perancangan Proses

Perancangan proses merupakan diagram alur (*flowchart*) dari aplikasi COD *(Circular Object Detection)*. Terdapat 3 proses utama yang ada dalam proses aplikasi COD, yaitu:

a. Proses *preprocessing grayscalling*, dapat dilihat pada gambar 3.4.

b. Proses *edge detection*, dapat dilihat pada gambar 3.5.
c. Proses Transformasi Hough, dapat dilihat pada gambar 3.6.

Sedangkan *flowchart* secara global dari aplikasi COD dapat dilihat pada gambar 3.3.

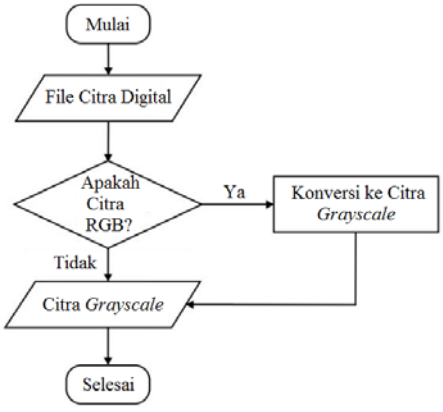


Gambar 3.6. *Global flowchart* proses COD

3.4.2. *Preprocessing Grayscalling*

Preprocessing Grayscalling merupakan proses untuk mengkonversi citra RGB menjadi citra *grayscale*. Adapun algoritma dari proses *preprocessing* citra digital yaitu:

a. Mengecek apakah citra masukan merupakan citra RGB atau tidak.
b. Jika merupakan citra RGB maka dikonversi ke citra *grayscale*. Jika bukan, maka langsung ke proses selanjutnya.

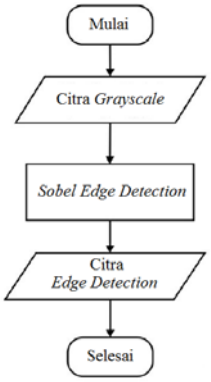


Gambar 3.7. *Flowchart* proses *preprocessing grayscalling* citra

3.4.3. *Edge Detection*

Edge detection merupakan proses untuk mendeteksi tepi-tepi dari setiap piksel yang terdapat pada citra, tujuannya adalah untuk menandai bagian yang menjadi detail citra. Metode yang digunakan adalah *Sobel Edge Detection*. Adapun algoritma dari *Edge Detetction* adalah sebagai berikut:

a. Masukkan citra *grayscale* sebagai *file* citra masukan.
b. Melakukan proses *Edge Detection* (deteksi tepi) dengan metode *Sobel*.
c. *File output* berupa citra *edge detection* hasil proses deteksi tepi.

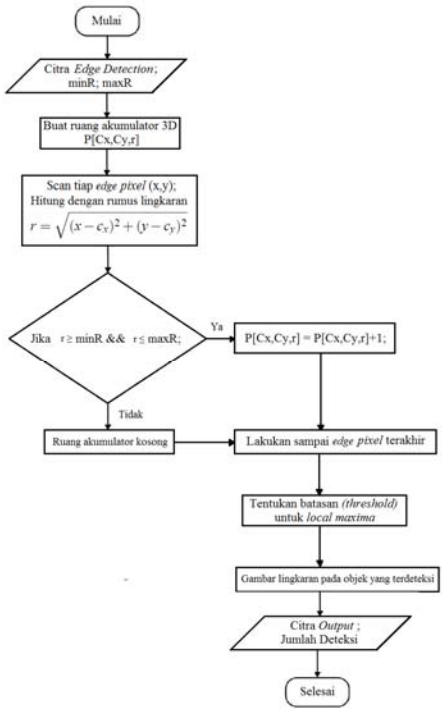


Gambar 3.8. *Flowchart* proses *Edge Detection*.

3.4.4. Transformasi Hough

Transformasi Hough merupakan proses untuk mendeteksi objek lingkaran yang terdapat pada citra. Adapun algoritma dari Transformasi Hough adalah sebagai berikut:

a. Masukkan citra *edge detection* sebagai *file* citra masukan dan minR,MaxR sebagai parameter masukan.
b. Membuat ruang akumulator 3D, misal P[Cx,Cy,r] dengan Cx adalah *center* lingkaran sumbu-x, Cy adalah *center* lingkaran sumbu-y, r adalah radius.
c. *Scan* tiap *pixel* (x,y) dan *pixel* (Cx,Cy) kemudian hitung radius dengan rumus lingkaran.
d. Hitunglah, jika r≥minR dan r≤maxR, maka tambahkan satu nilai ruang akumulator sehingga P[Cx,Cy,r] = P[Cx,Cy,r]+1. Lakukan sampai *pixel* terakhir.
e. Tentukan batasan *(threshold)* untuk menentukan *local maxima*.
f. *File output* berupa citra hasil proses deteksi dan jumlah hasil deteksi.



Gambar 3.9. *Flowchart* proses Transformasi Hough.

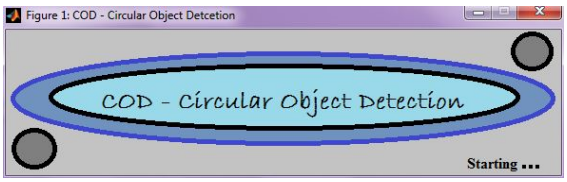
4. Implementasi dan Pengujian

Bab ini berisi tahap implementasi dan pengujian dalam pembangunan aplikasi COD *(Circular Object Detection)*.

4.1. Implmentasi *Interface*

4.1.1. *Form* Pembuka *(Splash Screen)*

Form pembuka *(splash screen)* ini terdapat informasi nama dan logo aplikasi, yaitu *Circular Object Detection (COD)*. Tampilan *form* pembuka dapat dilihat pada Gambar 4.1.

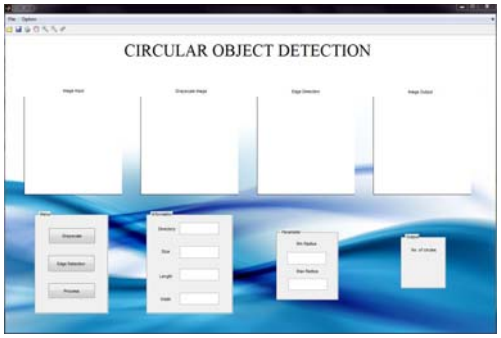


Gambar 4.10. *Form* Pembuka

4.1.2. Form Utama

Form utama ini terdapat informasi meliputi nama aplikasi, 4 sumbu citra, yaitu sumbu citra masukan, sumbu citra *grayscale*, sumbu citra *Edge Detection*, dan sumbu citra keluaran. Terdapat 2 menu, yaitu *File* yang berisi manajemen file dan *Option* yang dapat digunakan untuk bantuan penggunaan aplikasi COD. Terdapat toolbar yang berisi *Open File, Save, Print, Pan, Zoom In, Zoom Out, dan Imtool*. Terdapat 3 Panel Button, yaitu *Grayscale, Edge Detection,* dan *Process*. Terdapat Panel Parameter Input yang meliputi *Min. Radius* dan *Max. Radius* dan terdapat Panel Output, yaitu *No. of circles* untuk jumlah hasil deteksi objek.

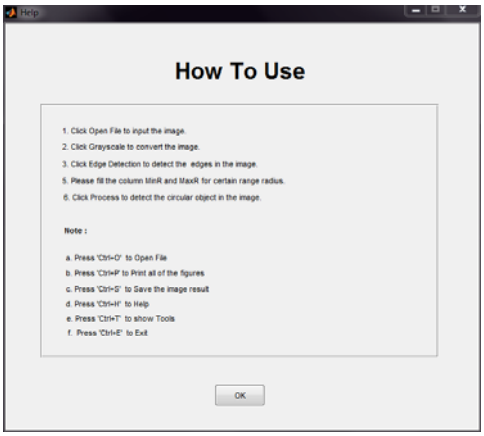
Form utama ini merupakan inti dari tahapan-tahapan dalam pendeteksian objek lingkaran. Pada *form* utama ini akan ditampilkan informasi tentang citra masukan seperti ukuran, panjang, dan lebar citra. Setiap ada kesalahan maka aplikasi COD ini akan menampilkan pesan *error*. Jadi, *user* harus mengikuti manual prosedur yang terdapat pada menu bantuan. *Form* utama ini dibuat sederhana agar *user* mudah dalam menggunakannya. Tampilan menu utama dapat dilihat pada Gambar 4.2.



Gambar 4.11 *Form* Utama

4.1.3. Menu Bantuan

Menu ini mempuyai tujuan untuk menampilkan cara penggunaan dari aplikasi COD. Menu ini digunakan untuk membantu para pengguna *(user)* yang masih belum mengerti tentang cara menjalankan aplikasi ini. Pada menu ini dijelaskan tentang tahapan-tahapan proses dari awal sampai akhir. Berikut tampilan menu bantuan pada gambar 4.3.



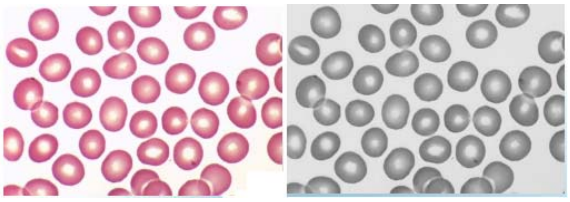
Gambar 4.12. Menu Bantuan

4.2. Implementasi Fungsi

Pada tahap implementasi fungsi, dilakukan 3 proses, yaitu *grayscalling, sobel edge detection,* dan Transformasi Hough.

4.2.1. *Grayscalling*

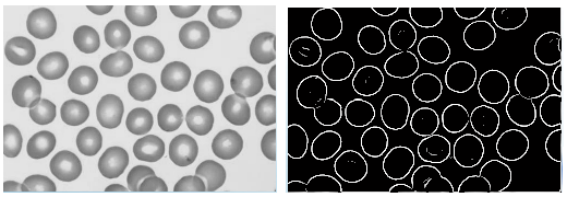
Pada proses *grayscalling,* dilakukan konversi citra RGB menjadi citra *grayscale.* Hasilnya dapat dilihat pada gambar 4.4.



Gambar 4.13. Konversi citra RGB menjadi *grayscale*

4.2.2. *Edge Detection*

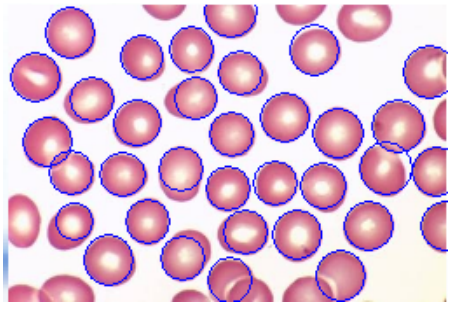
Pada proses *edge detection,* dilakukan proses pendeteksian tepi pada citra *grayscale* dengan metode *Sobel.* Hasilnya dapat dilihat pada gambar 4.5.



Gambar 4.14. Konversi citra *grayscale* menjadi citra deteksi tepi.

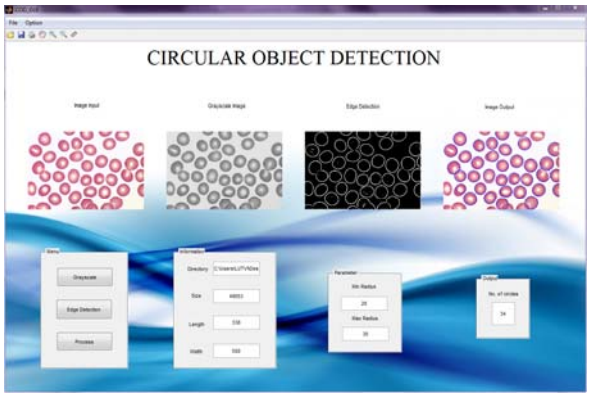
4.2.3. Transformasi Hough

Pada proses terakhir untuk mendeteksi objek lingkaran pada citra, digunakan metode Transformasi Hough. Hasilnya dapat dilihat pada gambar 4.6.



Gambar 4.15. Citra hasil pendeteksian dengan Transformasi Hough.

Setelah dilakukan proses Transformasi Hough, makan aplikasi akan menghasilkan citra keluaran dan jumlah hasil deteksi. Pada citra sel darah merah *(red blood cell)* didapatkan hasil deteksi sebanyak 34 objek dari total sebanyak 44 objek. Hasil proses aplikasi COD dapat dilihat pada gambar 4.7



4.3. Pengujian

4.3.1. Identifikasi Hasil Pengujian

Pengujian dilakukan sebanyak 6 tahap dengan total 2 citra yang berbeda. Setelah dihitung masing-masing persentase keberhasilan tiap tahap, maka dapat dihitung total rata-rata persentase keberhasilan yang dapat dilihat pada tabel 4.1.

Tabel 4.1. Rata-rata persentase hasil 6 pengujian

Pengujian	Rata-rata Persentase Keberhasilan
Pertama	100%
Kedua	78%
Ketiga	100%
Keempat	100%
Kelima	100%
Keenam	87,5%
Total rata-rata persentase keberhasilan	**94,25%**

4.3.2. Analisis Hasil Uji

Dari tabel 4.4 dapat diketahui bahwa rata-rata persentase keberhasilan deteksi pada pengujian pertama sebesar 100%, pada pengujian kedua sebesar 78%, pada pengujian ketiga sebesar 100%, pengujian keempat sebesar 100%, pengujian kelima sebesar 100%, dan pengujian keenam sebesar 87,5%. Sedangkan total rata-rata persentase keberhasilan pengenalan secara menyeluruh dari pengujian pertama sampai dengan pengujian keenam sebesar 94,25%. Tingkat keberhasilan deteksi dengan metode Transformasi Hough yang dihasilkan

sangat akurat meskipun jenis, ukuran, jumlah, jarak, dan sudut objek yang digunakan sebagai masukan berbeda-beda. Namun, pada kemiringan sudut 30^0 aplikasi COD tidak dapat mendeteksi objek lingkaran sama sekali.

Pada penelitian ini, ditetapkan nilai batas ambang *(threshold) local maxima* sebesar 0.5 atau lebih. Nilai ini digunakan untuk menentukan penggambaran konstruksi lingkaran setelah semua piksel discan dan dihitung dalam ruang akumulator. Selain itu, jika terdapat dua lingkaran dengan perbedaan jarak 50 piksel pada satu objek yang sama maka akan dianggap sebagai satu objek yang sama. Nilai-nilai ini ditentukan agar dapat meminimalisasi *error* dan menghasilkan persentasi keakuratan deteksi. Penentuan nilai-nilai ini didasari oleh rujukan *journal* yang terdapat pada daftar pustaka nomor ke-4. [4]

5. Kesimpulan dan Saran

5.1. Kesimpulan

Kesimpulan yang dapat diambil dalam pembuatan tugas akhir ini adalah:

1. Pada tugas akhir ini telah dihasilkan sebuah aplikasi pendeteksi objek lingkaran pada citra dengan Transformasi Hough.
2. Aplikasi ini adalah salah satu contoh aplikasi dalam bidang pengolahan citra digital yang berguna untuk mendeteksi objek-objek lingkaran pada suatu citra.
3. Pada penelitian ini, dihasilkan persentase keberhasilan pendeteksian objek lingkaran dengan tingkat keakuratan sebesar 94,25%.

5.2. Saran

Penelitian ini masih dapat dikembangkan lebih lanjut. Beberapa saran untuk mengembangkan penelitian ini adalah sebagai berikut :

1. Lakukan tahap *resizing* dan *sharpening* pada proses *preprocessing* agar proses deteksi berlangsung lebih cepat dan hasil jadi lebih akurat.
2. Aplikasi ini dapat dikembangkan menjadi sistem berbasis *real-time/video* dalam bidang *computer vision/robotics*, jadi tidak terbatas pada citra saja.

DAFTAR PUSTAKA

[1] Anfiandi, Irvan, *"Efficient Randomized Algorithm untuk Mendeteksi Lingkaran pada Citra"*, diakses dari http://www.stts.edu/browse/detailAbstrak?nrp=204115163 pada tanggal 01 April 2013, pukul 11.20 WIB .

[2] Anonim, "*Makalah Deteksi Tepi*", diakses dari http://komputernrecipe.files.wordpress.com/2010/06/makalah-citra.doc pada tanggal 01 April 2013, pukul 11.20 WIB.

[3] Fatta, Hanif., 2007, "*Konversi Format Citra RGB ke Format Grayscale Menggunakan Visual Basic"*, diakses dari http://p3m.amikom.ac.id/p3m/51%20-%20KONVERSI%20FORMAT%20CITRA%20RGB%20KE%20FORMAT%20GRAYSCALE.pdf, diakses pada tanggal 14 Juni 2012, pukul 14.50 WIB.

[4] Ioannou, Dimitrios, 1988, *"Circle recognition through a 2D Hough Transform and radius histograming"*, diakses dari http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.6622&rep=rep1&type=pdf, diakses pada tanggal 30 Juli 2013, pukul 12.25 WIB

[5] Ladjamudin, Al Bahra Bin, 2006, "*Rekayasa Perangkat Lunak*", Yogyakarta: Graha Ilmu

[6] Mabrur, Andik., 2011, "*Pengolahan Citra Digital menggunakan MATLAB*", Tulungagung.

[7] Moler, 2004, "*MATLAB About*", diakses dari *http://www.mathworks.com*, pada tanggal 30 Juli 2012, pukul 14.50 WIB.

[8] Pressman, Roger S. 2002. *"Rekayasa Perangkat Lunak"* : Pendekatan Praktisi (Buku 1)/Roger S. Presman; Ed.:I. Diterjemahkan oleh : CN Harnaningrum, Yogyakarta : Penerbit Andi.

[9] Sugiharto, Aris, 2006, "*Pemrograman GUI dengan MATLAB*", Penerbit Andi, Yogyakarta.

[10] Utami, Ayu Satyari., 2011, *"Perancangan Perangkat Lunak Sistem Temu Balik Citra Menggunakan Jarak Histogram dengan Model Warna YIQ"*, Medan: Universitas Sumatera Utara.

[11] Widodo, Yanu, 2007, "Penggunaan *Color* Histogram dalam *Image Retrieval*", diakses dari http://ilmukomputer.org/wp-content/uploads/2009/10/yanuwid-cbir.pdf, diakses pada tanggal 27 Juni 2012, pukul 20.00 WIB.

[12] Wirian, Yosef Bernadus, 2007, diakses dari http://thesis.binus.ac.id/doc/Bab2/2007-2-00445-MTIF_Bab%202.pdf, diakses pada tanggal 14 Oktober 2012, pukul 14.30 WIB