MENENTUKAN LUAS OBJEK CITRA DENGAN TEKNIK DETEKSI TEPI

Determining The Extents Of Image Objects Using Edge Detection Technique

Rany Zuriatna Utami1[1], I Made Budi Suksmadana2[1], Bulkis Kanata3[1]

ABSTRAK

Pada penelitian ini dilakukan perhitungan luas objek sebuah citra dari hasil pengambilan gambar RGB menggunakan USB webcam yang langsung dihubungkan pada PC. Penentuan luas objek dilakukan dengan proses segmentasi yang menerapkan metode deteksi tepi Canny. Untuk menentukan luas objek, proses segmentasi dapat dilakukan dengan berbagai cara diantaranya melakukan threshold, deteksi tepi dan menghilangkan objek-objek selain objek yang ingin diketahui nilainya.

Proses pengolahan citra dilakukan menggunakan software MATLAB dengan dua kondisi pengambilan gambar serta letak objek (kondisi) yang berbeda yaitu pada bagian kiri atas, pada bagian pusat dan pada bagian kanan bawah dalam sebuah citra.

Dari hasil penelitian diperoleh prosentasi keberhasilan perhitungan luas untuk objek segitiga sebesar 80,9% (perbedaan 19,1%), lingkaran 95,5% (perbedaan 4,50%) dan persegi 94,98% (perbedaan 5,02%), hal ini menunjukkan perhitungan yang baik.

***Kata Kunci**: citra, threshold, deteksi tepi, luas objek*

ABSTRACT

This research has been carried out extensive calculations object an image of a captured image using a USB webcam RGB directly connected to the PC. Determination the object ekstents is done by segmentation process by applying the Canny edge detection method. To determine the extents of the object, the segmentation process can be done in various ways including doing threshold, edge detection and eliminate objects other than the object that we want to know its value.

Image processing is done using MATLAB with two shooting conditions and object location (condition) that is different is in the top left, at the center and at the bottom right of an image.

From the results obtained by the percentage of success for the object triangle area calculation by 80.9% (19.1% difference), Circle 95.5% (difference 4.50%) and 94.98% square (difference 5.02%), it shows good calculation.

***Keywords**: image, threshold, edge detection, object ekstents*

PENDAHULUAN

Seringkali dalam sebuah citra terdapat beberapa objek di dalamnya dan objek dalam citra tersebut ingin diketahui nilai panjang, lebar, tinggi, volume atau luasannya untuk keperluan tertentu. Untuk menganalisa suatu objek di dalam citra misal menentukan luasan objek tersebut, maka terlebih dahulu dilakukan pemisahan antara objek citra yang satu dengan yang lainnya. Hal ini dapat dilakukan dengan proses segmentasi yang dapat dilakukan dengan berbagai cara diantaranya dengan *threshold*, deteksi tepi dan menghilangkan objek-objek selain objek yang ingin diketahui nilainya.

Pada penelitian ini dilakukan penentuan luas permukaan objek dengan deteksi tepi Canny pada citra RGB dari hasil pengambilan gambar dengan menggunakan *USB (Universal Serial Bus) webcam (website camera*) yang langsung dihubungkan pada *PC (personal computer)*. Tipe USB webcam yang digunakan CMOS VGA.

Citra adalah suatu gambaran, kemiripan atau imitasi dari suatu objek. Citra terbagi menjadi dua yaitu citra analog dan citra digital. Citra analog adalah citra yang bersifat kontinyu seperti gambar pada monitor televisi, foto sinar X, hasil *CT Scan*. Sedangkan pada citra digital adalah citra yang dapat diolah oleh komputer (Sutoyo, T. et al. 2009:9).

[1].Jurusan Teknik Elektro, Fakultas Teknik Universitas Mataram, Nusa Tenggara Barat, Indonesia
ranyzur@gmail.com1[1], ugikanata@te.ftunram.ac.id2[1], mdbudi@yahoo.com3[1]

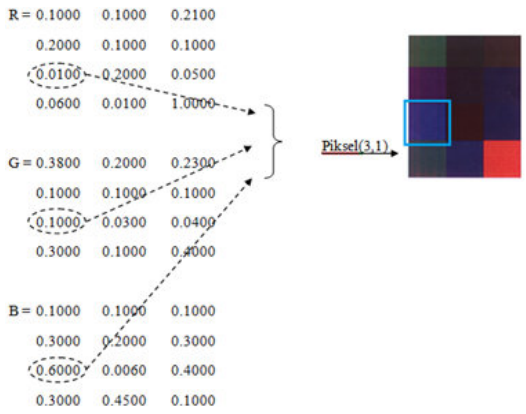
Sebuah citra digital dapat diwakili oleh sebuah matriks yang terdiri dari M kolom N baris, dimana perpotongan antara kolom dan baris disebut piksel (piksel=*picture element*), yaitu elemen terkecil dari sebuah citra. Piksel mempunyai dua parameter, yaitu koordinat dan intensitas atau warna. Nilai yang terdapat pada koordinat (x,y) adalah f(x,y), yaitu besar intensitas atau warna dari piksel di titik itu. Oleh sebab itu, sebuah citra digital dapat ditulis dalam bentuk matriks berikut.

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M) \\ f(1,0) & f(1,1) & \dots & f(1,M) \\ \vdots & \vdots & \dots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad \text{..... (1)}$$

(Sutoyo, T. et al. 2009:20).

Citra RGB merupakan cara standar untuk mempresentasikan warna data. Dalam MATLAB, komponen-komponen *red*, *green* dan *blue* dari sebuah citra RGB menempati tiga matriks intensitas secara terpisah dan masing-masing memiliki jumlah baris dan kolom yang sama. Intensitas piksel (warna piksel) merupakan gabungan dari masing-masing komponen R, G dan B sesuai dengan lokasinya.

Contoh: komponen R, G dan B sebuah citra ukuran 4 x 3 pada Gambar 1 adalah sebagai berikut.



Gambar 1 Piksel RGB

Intensitas piksel pada posisi (3,1) merupakan gabungan intensitas pada R(3,1), G(3,1) dan B(3,1) yakni gabungan 0.0100, 0.1000 dan 0.6000 yang menghasilkan warna biru (Kanata, B. 2009:9).

Pengolahan citra sangat erat hubungannya dengan ilmu pengenalan pola, yang secara umum bertujuan mengenali suatu objek dengan cara mengekstraksi informasi penting yang terdapat dalam suatu citra. Pada pengolahan citra digital, proses penampilan data yang diperoleh dari sebuah citra sangat penting karena bagaimanapun juga citra digital hasil olahan harus dapat dinilai oleh mata manusia melalui suatu penampil (*display*). *Display* yang digunakan biasanya berupa *graphic monitor* atau suatu *graphic printer* atau *plotter*. Namun sebelum suatu data dapat ditampilkan dari sebuah citra, diperlukan suatu proses terlebih dahulu yaitu perubahan citra dan pengolahan data.

Digitizer yaitu mengubah citra masukan menjadi sinyal listrik dan kemudian mencuplik sinyal listrik tersebut dengan menggunakan *A/D Converter* (*Analog to Digital Converter*). *Digitizer* ini dapat berupa *scanner* atau kamera video yang mengubah citra kontinu ke dalam suatu representasi numerik, sehingga citra ini dapat diproses oleh komputer digital (Wijaya, M. dan Prijono, A. 2007:25). Kemudian proses pengolahan data dilakukan oleh komputer. Citra digital yang dihasilkan pada proses *digitizer* umumnya berbentuk empat persegi panjang dan dimensi ukurannya dinyatakan sebagai tinggi (N) x lebar (M) (lebar x panjang). Citra digital yang tingginya N, lebarnya M dan memiliki *L* skala keabuan dapat dianggap sebagai fungsi :

$$\text{f(x,y)} \begin{cases} 0 \le x \le M \\ 0 \le y \le N \\ 0 \le f \le L \end{cases} \quad \text{.....(2)}$$

Histogram Citra. Histogram citra adalah grafik yang menggambarkan penyebaran nilai-nilai intensitas piksel suatu citra atau bagian tertentu dalam citra. Dari histogram dapat diketahui frekuensi kemunculan relatif dari intensitas pada citra tersebut, dapat menunjukkan banyak hal tentang kecerahan (*brightness*) dan kontras dari gambar.

Segmentasi. Faktor kunci dalam mengekstraksi ciri adalah kemampuan mendeteksi keberadaan tepi (*edge*) dari objek di dalam citra. Setelah tepi objek diketahui, langkah selanjutnya dalam analisa citra adalah segmentasi, yaitu mereduksi citra menjadi objek atau region, misalnya memisahkan objek-objek yang berbeda dengan mengekstraksi batas-batas objek (*boundary*).

Thresholding. *Thresholding* merupakan parameter yang digunakan untuk membedakan

objek dengan latar objek berdasarkan skala keabuannya. Cara untuk membedakan objek dengan latar objek yaitu dengan memberi batas T pada histogramnya. Jika sebuah titik terletak kurang dari batas T maka akan bernilai 0, sebaliknya jika titik terletak lebih dari batas T maka akan bernilai 1.

Deteksi Tepi Canny. Penelitian ini digunakan jenis deteksi tepi Canny. Deteksi Canny digunakan karena selain menentukan tepi-tepi yang baik arah horizontal maupun arah vertikal, juga mampu menemukan tepi-tepi yang lemah pada citra kemudian menghubungkan pada tepi-tepi yang kuat tersebut.

Berikut adalah langkah-langkah yang dilakukan dalam pendeteksian tepi sebuah objek citra menggunakan metode Canny:

1. Penghalusan Citra. Tidak dapat dihindari bahwa semua gambar yang diambil menggunakan kamera akan terdapat pula noise yang menyertai pada gambar. Untuk mengurangi kesalahan pada pendektesian tepi maka noise juga harus dikurangi. Citra dihaluskan menggunakan filter Gaussian dengan standar deviasai, σ, untuk mengurangi noise.

$\frac{1}{159}$

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

Gambar 2 Matriks Gaussian

Untuk melakukan filter Gaussian, matriks 5x5 tersebut dilewatkan di atas citra. Setiap piksel dilewatkan berulang sebagai jumlah dari nilai-nilai piksel sesuai ketetanggaan dari matriks 5x5 tersebut dibagi nilai total keseluruhan matriks 5x5 tersebut.

2. Gradien. Setelah penghalusan citra dan pengurangan noise, langkah selanjutnya adalah menemukan tepi yang kuat dengan menentukan gradien citranya. Metode Sobel melakukan pengukuran gradien spasial dua dimensi pada citra, kemudian perkiraan mutlak besarnya gradien atau kuatnya tepi pada setiap titik dapat ditemukan. Metode Sobel menggunakan konvolusi sepasang matriks 3x3, dimana sebuah matriks 3x3 memperkirakan gradien dalam arah-x (kolom) atau Gx dan matriks 3x3 lainnya memperkirakan gradien dalam arah-y (baris) atau Gy. Diperlihatkan sebagai berikut:

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

Gx

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

Gy

Gambar 3 Gradien Arah x dan y

Besarnya gradien atau kuatnya tepi kemudian didekati dengan menggunakan persamaan:

|G|=|Gx|+|Gy|…….....….....(3)

3. Arah Tepi. Arah tepi dihitung menggunakan gradien dalam arah x dan y. Namun kesalahan akan dihasilkan ketika penjumlahan arah x sama dengan nol. Jadi harus terdapat seperangkat pembatas setiap kali hal ini terjadi. Setiap kali gradien dalam arah x sama dengan nol, arah tepi harus sama dengan 90^o atau 0^o tergantung pada nilai gradien dalam arah y yang sama. Jika Gy memiliki nilai nol, maka arah tepi akan sama dengan 0^o atau 90^o. Persamaan untuk menemukan arah tepi adalah:

□ = arctan —— .....(4)

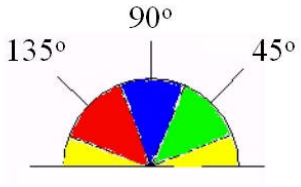
4. Orientasi Tepi. Setelah arah tepi diketahui, selanjutnya adalah menghubungkan arah tepi ke arah yang dapat ditelusuri dalam citra. Jadi jika piksel pada citra 5x5 selaras dengan:

x x x x x
x x x x x
x x **a** x x

x x x x x

x x x x x

Kemudian dengan melihat piksel **a**, hanya terdapat empat kemungkinan arah saat menjelaskan piksel-piksel sekitarnya yaitu 0^o (pada arah horizontal), 45^o (sepanjang diagonal positif), 90^o (pada arah vertikal) atau 135^o (sepanjang diagonal negatif). Orientasi tepi harus diselesaikan berdasarkan salah satu arah terdekat dari empat arah tersebut.



Gambar 4 Orientasi Tepi

Setiap arah tepi yang berada dalam kisaran kuning (0^o–22.5^o dan 157.5^o–180^o) diatur menjadi 0^o. Setiap arah tepi yang berada dalam kisaran hijau (22.5^o–67.5^o) diatur menjadi 45^o. Setiap arah tepi yang berada dalam kisaran biru (67.5^o–112.5^o) diatur menjadi 90^o. Setiap arah tepi yang berada dalam kisaran merah (112.5^o–157.5^o) diatur menjadi 135^o.

5. Pendeteksian Sepanjang Tepi. Langkah berikutnya adalah menelusuri sepanjang tepi berdasarkan pada perhitungan kekuatan gradien sebelumnya dan arah tepi. Setiap piksel dilakukan berulang menggunakan dua keadaan. Jika piksel yang dilalui memiliki kekuatan atau nilai lebih tinggi dari *threshold* yang ditentukan, maka saklar dieksekusi. Penentuan aktifnya saklar ditentukan oleh arah tepi piksel saat itu.

   Arah tepi yang telah diketahui memiliki baris dan kolom. Jika memiliki arah tepi yang sama dan kekuatan gradien lebih rendah dari *threshold*-nya, piksel diatur berwarna putih yang juga berlaku untuk piksel sepanjang tepi. Selanjutnya setiap terdeteksi peningkatan tepi yang secara signifikan, akan ditetapkan berwarna putih sementara piksel lainnya diatur berwarna hitam sampai menemukan piksel yang dianggap tepi lainnya pada citra tersebut.
6. *Hysterisis. Hysterisis* digunakan sebagai sarana untuk menghilangkan noise. Noise yang dimaksud adalah garis yang melewati bentuk garis yang seharusnya yang disebabkan oleh keluaran yang berfluktuasi di atas dan di bawah *threshold*. Jika ambang tunggal, T1, diterapkan pada citra dan tepi memiliki kuat yang rata-rata sama dengan T1 dimana memiliki *noise* maka akan ada kasus yaitu tepi menukik di *threshold*-nya. Begitu pula jika noise berada di atas *threshold*, akan membuat garis tepi tampak putus-putus.

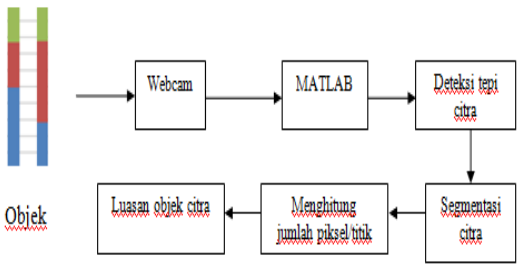
   Untuk menghindari atau mengurangi tepi yang menghasilkan *noise*, *hysterisis* menggunakan dua *threshold*, T1 dan T2, tinggi dan rendah dengan T1<T2. Setiap piksel dalam citra memiliki nilai lebih besar dari T1 maka akan dianggap menjadi piksel tepi lemah dan otomatis ditandai sebagai tepi. Kemudian setiap piksel yang terhubung ke tepi piksel ini dan yang memiliki nilai lebih besar dari T2 juga terpilih sebagai piksel tepi kuat dan ditandai sebagai tepi kuat (Green, Bill. 2002).

METODE PENELITIAN

Alat Penelitian. Alat yang digunakan dalam penelitian ini terdiri atas:

1. Satu unit PC (*personal computer*) yang dilengkapi dengan perangkat lunak (*software*) dengan spesifikasi sebagai berikut:
   - *Operating system* (OS) : *Windows 7 Starter*
   - Program aplikasi : MATLAB versi 7.0.4
2. Satu unit *USB webcam*
3. Satu set penyangga *USB webcam*, dengan lensa *USB webcam* yang berjarak 31.5 cm dari permukaan objek citra dan dilengkapi dengan pencahayaan dengan spesifikasi lampu neon daya 5 W, tegangan 220-240V, frekuensi 50-60Hz dan arus 35 mA.

Blok Diagram Sistem



Gambar 5 Bagan Proses Penentuan Luasan Objek Citra

HASIL DAN PEMBAHASAN

Pengambilan Gambar Objek Citra. Mengaktif-kan *webcam* dengan menggunakan perintah berikut:

```
% aktivasi webcam
clear all; close all; clc;


info = imaqhwinfo % jenis webcam, winvideo
jenis = imaqhwinfo('winvideo') % webcam apa saja yg terbaca
obj = videoinput('winvideo',2) % usb webcam tersedia
usb_webcam = imaqhwinfo(obj) % spesifikasi usb webcam
preview(obj) % video
```

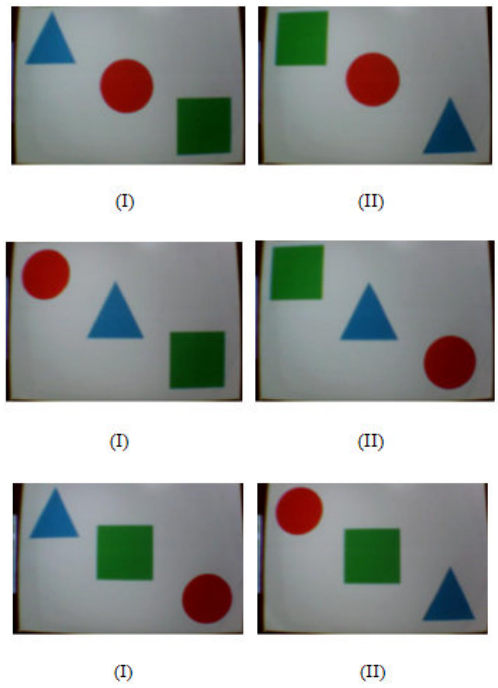
Menampilkan figur video *webcam* yaitu dengan perintah berikut:

```
% Setting video agar menampilkan gambar
% sesuai warna citranya dan full frame
clc; close all; clear all;


imaqreset;


vid = videoinput('winvideo',2,'YUY2_320x240');
set(vid,'ReturnedColorSpace','rgb');
set(vid,'TriggerRepeat',Inf);
figure; % ensure smooth display
set(gcf,'doublebuffer','on');
start(vid)


threshold = 0;
while(vid.FramesAcquired<=10)
    data = getdata(vid,1);
    imshow(data);
```

```
    set(gcf,'Units','normal')
  set(gca,'Position',[0 0 1 1.12])


end
delete(vid)
clear vid
```

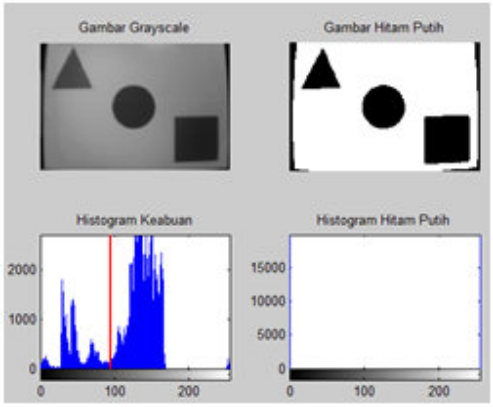
Proses Pengolahan Citra. Pengambilan gambar dengan *webcam* dilakukan pada jarak 31.5 cm dari permukaan objek sebanyak lima kali untuk masing-masing gambar. Pada masing-masing gambar, pusat objeknya berbeda-beda yaitu lingkaran, segitiga dan persegi. Disimulasikan pengambilan gambar berada dalam sebuah ruang semi tertutup untuk menghindari pembiasan agar cahaya lebih fokus dan merata di area gambar atau citra.

Diperoleh hasil pengambilan gambar sebagai berikut:



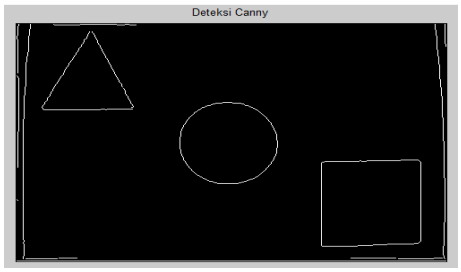
Gambar 6 Hasil Pengambilan Gambar menggunakan *USB webcam* dengan lingkaran, segitiga dan persegisebagai pusat objek dengan dua kondisi

Citra yang telah diambil diubah menjadi citra keabuan kemudian dibuat histogramnya.



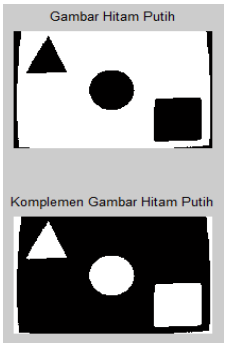
Gambar 7 Gambar citra keabuan, hitam putih dan histogram dengan lingkaran sebagai pusat objek pada Kondisi pertama

Dari gambar keabuan, dapat ditentukan tepi-tepi bentuk objek tersebut menggunakan deteksi tepi Canny dengan perintah `edge(nama_file,'canny')`, dan untuk membuat gambar menjadi hitam putih sesuai tipe gambar yaitu uint8, digunakan perintah `uint8(zeros(size(nama_file)))`.



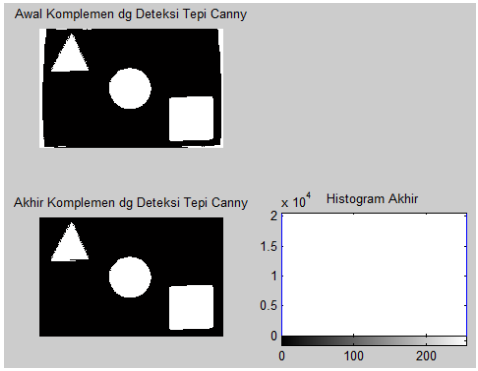
Gambar 8 Hasil deteksi tepi Canny pada kondisi pertama

Untuk mendapatkan latar objek berwarna hitam, maka pada gambar hitam putih dilakukan pembalikan warna atau komplemen warna dengan perintah `imcomplement`. Hal ini bertujuan untuk menggabungkan gambar hitam putih dengan gambar hasil deteksi tepi agar dapat dilakukan segmentasi.



Gambar 9 Hasil Komplemen pada kondisi Pertama

Dilakukan penjumlahan atau peng-gabungan gambar dan dapat dilihat beberapa garis terbentuk yang berasal dari hasil deteksi tepi. Oleh karena itu, dilakukan segmentasi atau "pembersihan" yaitu menghilangkan objek-objek yang tidak diinginkan dengan cara mengubah-nya menjadi seperti warna latarnya yaitu hitam.

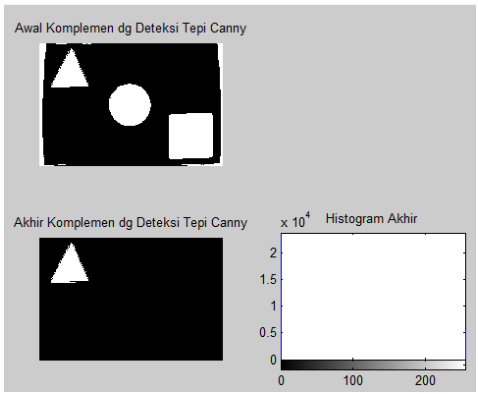


Gambar 10 Hasil penjumlahan dan setelah dilakukan pembersihan area pada kondisi pertamaserta histogram

Menghitung Luasan Objek Citra. Luasan asli citra adalah 626.24 cm^2, luas objek segitiga 32 cm^2, luas lingkaran 34.2257 cm^2 dan luas persegi adalah 49 cm^2.

Dilakukan *cropping* gambar, dalam hal objek pertama, agar dapat dipisahkan kemudian dihitung jumlah piksel objek tersebut, maka digunakan perintah:

```
% objek pertama
tmp(1:7,:)=0; % Merubah nilai menjadi 0
untuk baris 1-7
tmp(128:326,:)=0;
tmp(:,1:25)=0; % Merubah nilai menjadi 0
untuk kolom 1-25
tmp(:,150:484)=0;
0 warna hitam dan 1 warna putih.
```



Gambar 11 Segmentasi untuk segitiga dan histogramnya pada kondisi pertama

Menggunakan perintah berikut agar jumlah piksel dapat dihitung dan diperoleh nilai luasan untuk objek pertama:

```
% Menghitung jumlah titik putih setelah
dilakukan pembersihan
[count_tmp,x_hp] = imhist(tmp);
jumlah_titik_putih=count_tmp(256)
luas_objek=(luas_citra*jumlah_titik_putih)/j
umlah_seluruh_titik
```

Diperoleh jumlah piksel untuk segitiga pada kondisi pertama dengan lingkaran sebagai pusat objek sebanyak 5577 titik yang dihitung menggunakan persamaan perbandingan, sehingga diperoleh luasannya yaitu 22,16 cm^2.

Dilakukan cara yang sama untuk objek kedua dan ketiga pada kondisi pertama dan kedua, serta untuk pusat objek yang berbeda, yang dapat dilihat pada tabel 1.

Tabel.1 Luas rata-rata objek

OBJEK	Asli	LUAS RATA-RATA(cm^2)		
		Perhitungan		
		kondisi-1	Kondisis-2	Kondisi-3
Segi tiga	32	22,16	27,32	28,22
Lingkaran	34,23	31,68	38,26	37,58
Persegi	49	45,38	55,30	54,08

Rata-rata luasan untuk masing-masing objek pada ketiga kondisi sebagai berikut:
Segi tiga 25,9 cm^2, Lingkaran 35,84 cm^2 dan Persegi 51,59 cm^2. Tingkat keberhasilan perhitungan dengan Matlab untuk objek segitiga 80,9% (perbedaan 19,1%), Lingkaran 95,5% (perbedaan 4,50%) dan persegi 94,98% (perbedaan 5,02%), hal ini menunjukkan perhitungan yang baik.

KESIMPULAN

Berdasarkan hasil perhitungan luas secara manual (Asli) dan menggunakan MATLAB, dari lima kali pengambilan gambar untuk citra dengan pusat objek (kondisi) yang berbeda diperoleh tingkat kesuksesan perhitungan segitiga 80,9% (perbedaan 19,1%), Lingkaran 95,5% (perbedaan 4,50%) dan persegi 94,98% (perbedaan 5,02%), hal ini menunjukkan perhitungan yang baik.

DAFTAR PUSTAKA

Kanata, B. 2009. "*Diktat Kuliah Pengolahan Citra (Edisi I)"*. Program Studi Teknik Elektro, Fakultas Teknik, Universitas Mataram. Mataram

Prasetyo, E. 2011. "*Pengolahan Citra Digital dan Aplikasinya Menggunakan Matlab"*. Penerbit ANDI. Yogyakarta

Sutoyo, T. et al. 2009. "*Teori Pengolahan Citra Digital"*. Penerbit ANDI Yogyakarta

Wijaya, M. dan Prijono, A. 2007. "*Pengolahan Citra Digital Menggunakan MatLab*." Penerbit INFORMATIKA. Bandung

Young, I. T. Gerbrands, J. J. and van Vliet, L. J. 1998. *Fundamentals of Image Processing*. Delft University of Technology. Den Haag

Anonim. 2013. "*Find Edges in Grayscale Image MATLAB*," diunduh dari: http://www.mathworks.com/help/images/ref/edge.html (diakses pada 21 Februari 2013, pukul 10:04:40 WITA)

Green, Bill. "2002. *Canny Edge Detection Tutorial*," diunduh dari: http://dasl.mem.drexel.edu/alumni/bGreen/www.pages.drexel.edu/_weg22/can_tut.html (diakses pada 08 Desember 2013, pukul 09:51:09 WITA)