LAPORAN

ANALISIS PROGRAM PENGOLAHAN CITRA DIGITAL

"Object Detection, Object Tracking dan Estimasi Jarak menggunakan Single Camera dengan Segmentasi HSV"

Disusun Oleh :

Jan Putra Bahtra Agung S. Pelawi (1221018)

Dosen Pembimbing :

Wahyu Setyo Pambudi, ST., MT

PROGRAM STUDI TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

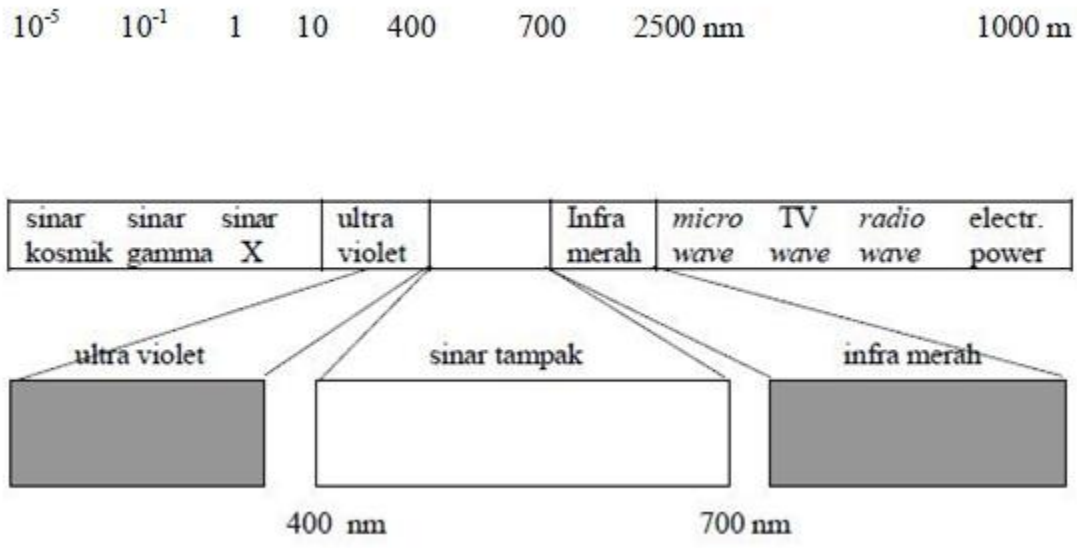
UNIVERSITAS INTERNASIONAL BATAM

2014

Bab 1
Landasan Teori

1.1 HSV

Warna yang diterima oleh mata dari sebuah objek ditentukan oleh warna sinar yang dipantulkan oleh objek tersebut. Sebagai contoh, suatu objek berwarna hijau karena objek tersebut memantulkan sinar biru dengan panjang gelombang 450 sampai 490 nanometer (nm). Warna sinar yang direspon oleh mata adalah sinar tampak (visible spectrum) dengan panjang gelombang berkisar dari 400 (biru) sampai 700 nm (merah).



Gambar 1.1 Spektrum Warna

Warna-warna yang diterima oleh mata manusia merupakan hasil kombinasi cahaya dengan panjang gelombang berbeda. Penelitian memperlihatkan bahwa kombinasi warna yang memberikan rentang warna yang paling lebar adalah *red* (*R*), *green* (*G*), dan *blue* (*B*). Ketiga warna tersebut dinamakan **warna pokok** (*primaries*), dan sering disingkat sebagai warna dasar *RGB.* Warna-warna lain dapat diperoleh dengan mencampurkan ketiga warna pokok tersebut dengan` perbandingan tertentu .

Selain RGB, warna juga dapat dimodelkan berdasarkan atribut warnanya. Setiap warna memiliki 3 buah atribut, yaitu intensity (I), hue (H), dan saturation (S).

a. Value/ Intensity/Brigthness/Luminance

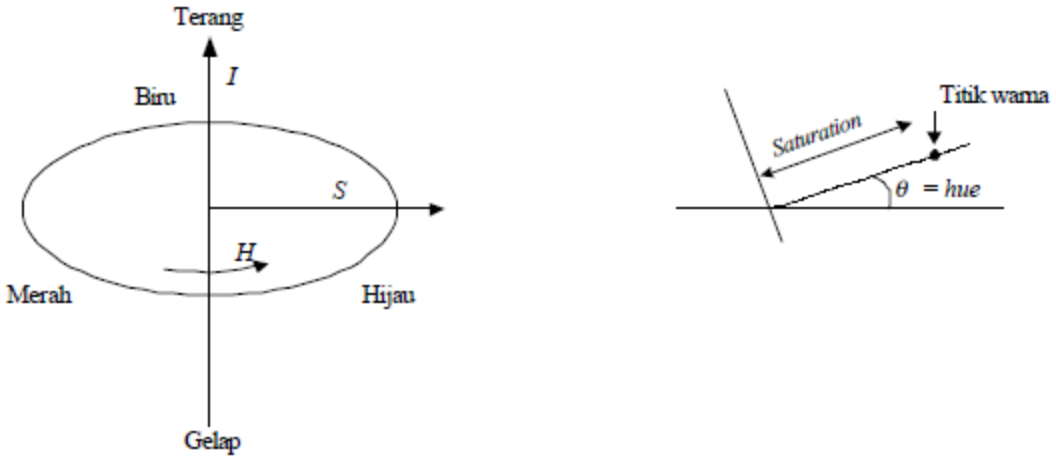
Atribut yang menyatakan banyaknya cahaya yang diterima oleh mata tanpa mempedulikan warna. Kisaran nilainya adalah antara gelap (hitam) dan terang (putih).

b. Hue

Menyatakan warna sebenarnya, seperti merah, violet, dan kuning. Hue digunakan untuk membedakan warna-warna dan menentukan kemerahan (redness), kehijauan (greenness), dsb, dari cahaya. Hue berasosiasi dengan panjang gelombang cahaya, dan bila kita menyebut warna merah, violet, atau kuning, kita sebenarnya menspesifikasikan hue-nya.

c. Saturation

Menyatakan tingkat kemurnian warna cahaya, yaitu mengindikasikan seberapa banyak warna putih diberikan pada warna. Sebagai contoh, warna merah adalah 100% warna jenuh (saturated color), sedangkan warna pink adalah warna merah dengan tingkat kejenuhan sangat rendah (karena ada warna putih di dalamnya). Jadi, jika hue menyatakan warna sebenarnya, maka saturation menyatakan seberapa dalam warna tersebut. Dalam praktek, hue dikuantisasi dengan nilai dari 0 sampai 255; 0 menyatakan merah, lalu memutar nilai-nilai spektrum tersebut kembali lagi ke 0 untuk menyatakan merah lagi. Ini dapat dipandang sebagai sudut dari 0° sampai 360°. Jika suatu warna mempunyai saturation = 0, maka warna tersebut tanpa hue, yaitu dibuat dari warna putih saja. Jika saturation = 255, maka tidak ada warna putih yang ditambahkan pada warna tersebut. Saturation dapat digambarkan sebagai panjang garis dari titik pusat lingkaran ke titik warna. Intensity nilainya dari gelap sampai terang (dalam praktek, gelap = 0, terang = 255). Intensity/Value dapat digambarkan sebagai garis vertikal yang menembus pusat lingkaran. Ketiga atribut warna (V, H, dan S) digambarkan dalam model HSV (ada juga yang menyebutnya model HIS) yang diperlihatkan pada Gambar 1.1



Gambar 1.2 HSV atau HSI

Nilai atribut- atribut tersebut dapat diperoleh dari perbandingan nilai RGBnya. Berikut persamaan konversi RGB ke HSV

$$H = \begin{cases} 60° \; x \left(\dfrac{(G-B)}{\delta}\right) & if \; MAX = R \\ 60° \; x \left(\dfrac{(B-R)}{\delta} + 2\right) & if \; MAX = G \\ 60° \; x \left(\dfrac{(R-G)}{\delta} + 2\right) & if \; MAX = B \\ not \; defined \; if \; MAX = 0 \end{cases} \quad (1.1)$$

$$S = \begin{cases} \dfrac{\delta}{MAX} & If \; MAX \neq 0 \\ 0 & if \; MAX = 0 \end{cases} \quad (1.2)$$

$$V = MAX \quad (1.3)$$

Dimana: $\delta = \max(R, G, B) - \min(R, G, B)$

$MAX = \max(R, G, B)$

1.2 Threshold

Thresholding merupakan salah satu teknik segmentasi yang baik digunakan untuk citra dengan perbedaan nilai intensitas yang signifikan antara latar belakang dan objek utama (Katz,2000). Dalam pelaksanaannya Thresholding membutuhkan suatu nilai yang digunakan sebagai nilai pembatas antara objek utama dengan latar belakang, dan nilai tersebut dinamakan dengan threshold.

Thresholding digunakan untuk mempartisi citra dengan mengatur nilai intensitas semua piksel yang lebih besar dari nilai threshold T sebagai latar depan dan yang lebih kecil dari nilai threshold T sebagai latar belakang. Biasanya pengaturan nilai threshold dilakukan berdasarkan histogram grayscale .

1.3 Estimasi Jarak

Untuk estimasi jarak digunakan metode moore-penrose pseudo inverse. Untuk estimasi ini digunakan persamaan berikut

$$D = c2\ x\ A^2 + c1\ x\ A + c0 \qquad (1.4)$$

Dimana : D = estimasi jarak (cm)

A = radius lingkaran (dalam piksel)

c1,c2 dan c0 merrupakan nilai polynominal dan nilainya dihitung dengan persamaan sebagai berikut.

$$C = W^{\dagger} D \qquad (1.5)$$

Dimana C adalah variable polynominal matriks 1x3 yang akan dicari , dan $W^{\dagger}$ merupakan pseudo invers matrix n x m (n = 3 dan m matriks bergantung sample data) dan D adalah nilai estimasi jarak (cm) dan berbentuk matriks 1 x banyaknya data.

Bab 2

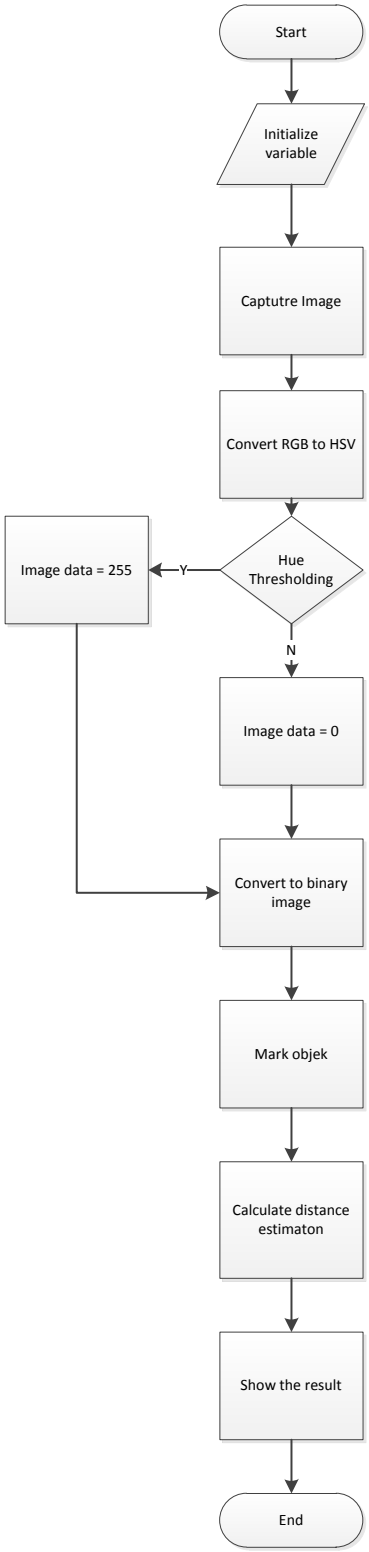
Metode Peneltian

Metode penelitian yang digunakan dalam kegiatan kali ini adalah dengan membuat program Image Processing untuk melakukan fungsi Objek Detection, Object tracking dan perhitungan estimasi jarak menggunakan EmguCV & Visual C#. Program ini dibuat bertujuan sebagai media untuk pengambilan data yang digunakan untuk melakukan analisa. Berikut Algoritma, Flowchart dan Penjelasan program tersebut.

2.1 Algoritma

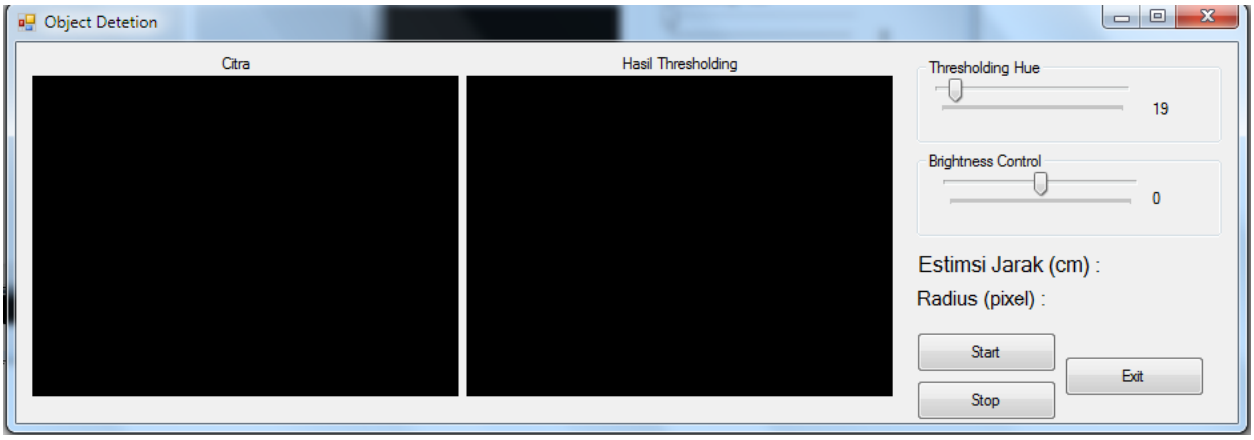
- Jalankan Program
- Inisialisasi variable yang digunakan
- Menangkap informasi citra menggunakan single camera(webcam)
- Melakukan konversi citra RGB (hasil tangkapan camera) menjadi HSV
- Melakukan thresholding Hue, jika nilai berada pada range hue yang diinginkan maka nilai piksel 255, jika tidak maka nilai piksel 0
- Melakukan konversi binary image hasil thresholding
- Memberi penandaan kepada objek yang terdeteksi
- Menghitung estimasi jarak
- Menampilkan hasil
- Berhenti

2.2 Flowchart



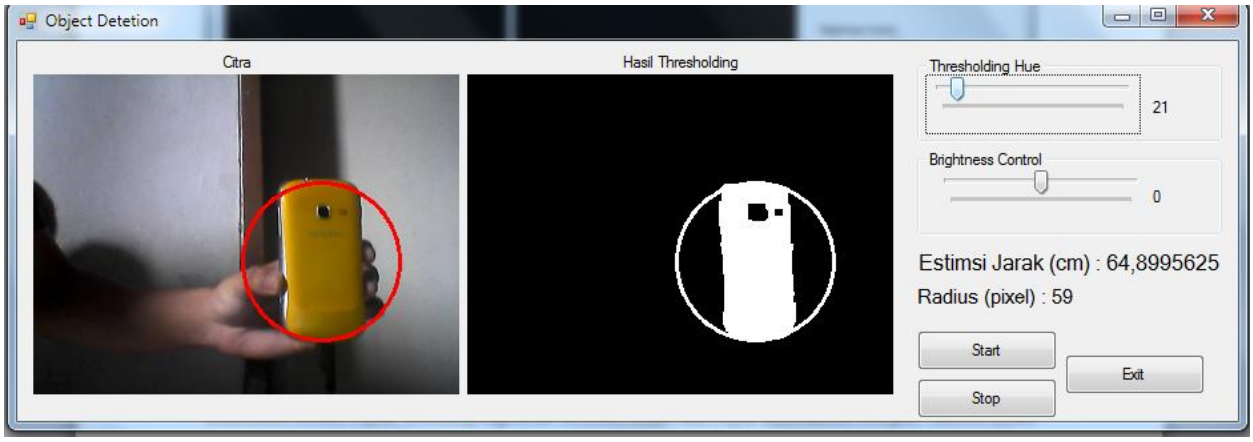
2.3 Penjelasan Program

Berikut tampilan program yang dibuat.



Gambar 2.1 Tampilan Program

Cara kerja program, klik button start untuk memulai program menangkap citra, kemudian tentukan nilai hue yang digunakan untuk melakukan deteksi objek sesuai dengan warna. Sedangkan Brghtness control digunakan untuk mengatur tingkat kecerahan, agar mendapat hasil thresholding yang baik. Brightness control ini digunakan untuk kondisi dimana hasil thresholding kurang smepurna untuk dilakukan deteksi objek. Setelah pengaturan tersebut, maka saat objek dilketakkan didepan camera maka objek akan terdeteksi dan ditandai dengan lingkaran yang mengikuti objek tersebut. Untuk hasil estimasi jarak akan terlihat label diatas button start.



Gambar2.2 Tampilan program saat dijalankan

Berikut beberapa variable yang harus dideklarasikan terlebih dahulu secara public karena variable berikut merupakan variable yang sering digunakan.

```
Capture capture = new Capture();
 int x, y,r;
int xpos0,ypos0, temp11, temp12;
int blue, red, green;
Image<Bgr, Byte> ColoredIMG = new Image<Bgr, byte>(320, 240);
Image<Hsv, Byte> HSVIMG = new Image<Hsv, byte>(320, 240);
Image<Gray, Byte> GrayIMG = new Image<Gray, byte>(320, 240);
```

Dari variable yang deklarasikan diatas terdapat 3 variable berjenis Image, yaitu ColoredIMG, GrayIMG dan HSVIMG. ColoredIMG bertipe RGB untuk menyimpan hasil capture dan GrayIMG bertipe Grayscale digunakan untuk menyimpan binary image hasil thresholding hue dan HSVIMG bertipe HSVuntuk menyimpan hasil konversi RGB ke HSV dan digunakan untuk segmentasi dalam objek detection.

Untuk konversi HSV dilakukan menggunakan coding berikut:

```
ColoredIMG = capture.QueryFrame().Flip(Emgu.CV.CvEnum.FLIP.HORIZONTAL);
ColoredIMG = ColoredIMG.Resize(320, 240, Emgu.CV.CvEnum.INTER.CV_INTER_LINEAR,
true);

HSVIMG = ColoredIMG.Convert<Hsv, Byte>();
```

Coding ini bertujuan mengkonversi RGB hasil capture yang disimpan pada variable ColoredIMG.

Untuk deteksi objek berdasarkan data hue, digunakan coding constructor seperti dibawah ini. Hasil dari koding tersebut adalah Binary image yang dihasilkan dari thresholding nilai hue yang sesuai dengan warna objek. Hasil citra tersebut akan memisahkan objek yang ingin dideteksi dengan objek lainnya dan background. Objek yang terdeteksi akan berwarna putih dan yang lainnya akan berwana hitam.

```
void Detect(Image<Hsv, Byte> input, Image<Gray, Byte> output)
        {
            int DHue;
            int VHue = trackBar1.Value;
            int Gray;

            for (y = 0; y < input.Height; y++)
            {
                for (x = 0; x < input.Width; x++)
                {
                    DHue = input.Data[y, x, 0];
                    if (DHue > VHue-5 &&  DHue < VHue+5 )
                    {
                        Gray = 255;

                    }
                    else
                    {
                        Gray = 0;
                    }

                    output.Data[y, x, 0] = Convert.ToByte(Gray);
                }
            }
            output._Erode(3);
            output._Dilate(1);
        }
```

Selanjutnya hasil dari thresholding akan dilakukan tracking dengan cara membuat lingkaran sebagai penanda objek. Tracking dilakukan dengan menghitung luas pixel yang berwarna putih pada binary immage yang dihasilkan dan kemudian menentukan titik tengah objek, sebgai titik pusat lingkaran penanda. Berikut contructor untuk tracking tersebut.

```
void Tracker(Image<Gray, Byte> input, Image<Bgr, Byte> input1)
      {
          int gray, i;
          int[] xpos = new int[200000];
          int[] ypos = new int[200000];
          i = 0;
          PointF center = new PointF();
          for (y = 0; y < input.Height; y++)
          {
              for (x = 0; x < input.Width; x++)
              {
                  gray = input.Data[y, x, 0];
                  if (gray == 255)
                  {
                      xpos[i] = x;
                      ypos[i] = y;

                      i = i + 1;
                  }
              }
          }
              if (i != 0)
              {
                  i = i - 1;
              }
              else
              {
                  i = 1;
              }

             temp11 = (xpos[i] - xpos[0])/2;
            temp12 =  ((ypos[i] - ypos[0]))/2;
              xpos0 = xpos[0] + temp11;
              ypos0 = ypos[0]+ temp12;
              center.X = xpos0;
              center.Y = ypos0;
              r = temp12;
              if (r != 0)
              {
                  input.Draw(new CircleF(center, r), new Gray(255), 2);
                  input1.Draw(new CircleF(center, r), new Bgr(Color.Red), 2);

              }
```

Untuk beberapa kondisi diperlukan pengaturan kecrahan maka, diperlukan coding untuk mengatur kecerahan sebagai berikut.

```
void BrightnessOP(Image<Bgr, byte> input, Image<Bgr, byte> output,  int k)
{
   int Bred, Bblue, Bgreen;

            for (x = 0; x < input.Height; x++)
            {
                for (y = 0; y < input.Width; y++)
                {
                    blue = input.Data[x, y, 0];
                    green = input.Data[x, y, 1];
                    red = input.Data[x, y, 2];

                    Bblue = blue + k;
                    Bgreen = green + k;
                    Bred = red + k;


                    if (Bblue > 255)
                    {
                        Bblue = 255;
                    }
                    else if (Bblue < 0)
                    {
                        Bblue = 0;
                    }

                    if (Bgreen > 255)
                    {
                        Bgreen = 255;
                    }
                    else if (Bgreen < 0)
                    {
                        Bgreen = 0;
                    }

                    if (Bred > 255)
                    {
                        Bred = 255;
                    }
                    else if (Bred < 0)
                    {
                        Bred = 0;
                    }

                    output.Data[x, y, 0] = Convert.ToByte(Bblue);
                    output.Data[x, y, 1] = Convert.ToByte(Bgreen);
                    output.Data[x, y, 2] = Convert.ToByte(Bred);
                }
```

Sedangkan untuk perhitungan estimasi jarak digunakan persamaan (1.4), berikut fuction tersebut. Untuk koefisen yang dipergunnakan akan dibahas pada bab 3.

```
double distance(double r)
        {
            r = r / 2;
            double c2,c1,c0;
            c2 = 0.00105;
            c1 =-1.18260;
            c0 = 98.8725;
            double hasil;
            hasil = c2 * (r * r);
            hasil = hasil + (c1 * r);
            hasil = hasil + c0;
            return hasil;
         }
```

Keseluruhan Constructor dan function diatas diletakan pada timer untuk menghasilkan citra secara real time, berikut codingnya:

```
private void timer1_Tick_1(object sender, EventArgs e)
        {
            ColoredIMG =
capture.QueryFrame().Flip(Emgu.CV.CvEnum.FLIP.HORIZONTAL);
            ColoredIMG = ColoredIMG.Resize(320, 240,
Emgu.CV.CvEnum.INTER.CV_INTER_LINEAR, true);

            BrightnessOP(ColoredIMG, ColoredIMG, trackBar2.Value);

            HSVIMG = ColoredIMG.Convert<Hsv, Byte>();
            imageBox1.Image = ColoredIMG;
            Detect(HSVIMG, GrayIMG);
            Tracker(GrayIMG, ColoredIMG);
            imageBox2.Image = GrayIMG;
            if (r != 0)
            {
                label3.Text = "Estimsi Jarak (cm) :" + " " +
Convert.ToString(distance(r)) + " " + "cm";
        }
            label4.Text = "Radius (pixel) :" + " " + Convert.ToString(r);
        }
```

Bab 3

Analisa dan Pembahasan

Deteksi Objek

Pada kegiatan yang dilakukan, yaitu deteksi objek dan tracking, digunakan objek berupa back case handphone dengan nilai Hue berkisar 16 – 30. Pada range ini warna yang dapat ditangkap adalah warna kuning kejinggaan. Warna ini dipilih karena warna ini sangat mencolok dan kontras jika dibandingkan warna objek disekitar pengetesan. Posisi objek ditentukan tegak seperti gambar dibawah.

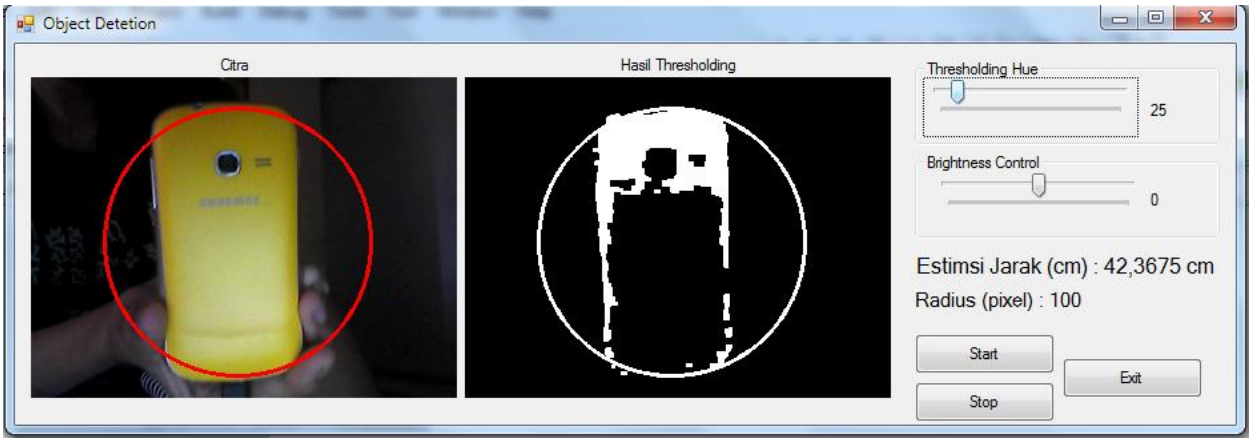
Gambar 3.1 Objek yang digunakan

Untuk deteksi objek, dipergunakan metode segmentasi HSV. Hal ini karena dengan menggunakan citra HSV, warna dapat diklasifikasikan dengan satu nilai saja, yaitu nilai Hue, karena nilai ini merupakan representasi panjang gelombang cahaya, berbeda dengan RGB yang memerlukan 3 warna dasar. Hal ini mempermudah untuk dilakukan thresholding, sehingga jika nilai hue piksel citra objek sesuai pada range nilai hue warna yang ditentukan sebagai nilai thresholdnya maka nilai piksel diubah menjadi 255 dan jika diluar range tersebut maka nilai piksel menjadi 0. Hasil proses thresholding ini menghasilkan binary image, dengan objek yang

ingin dideteksi menjadi berwarna putih dan objek lain serta backgraound berwarna hitam. Hasil ini berhasil mengidentifikasi objek yang dideteksi dan objek lain serta background. Untuk menunjukkan hasil deteksi objek, maka hasil binary image ini diolah untuk menghasilkan object tracking untuk menandai objek dan memberitahu bahwa objek telah terdeteksi.

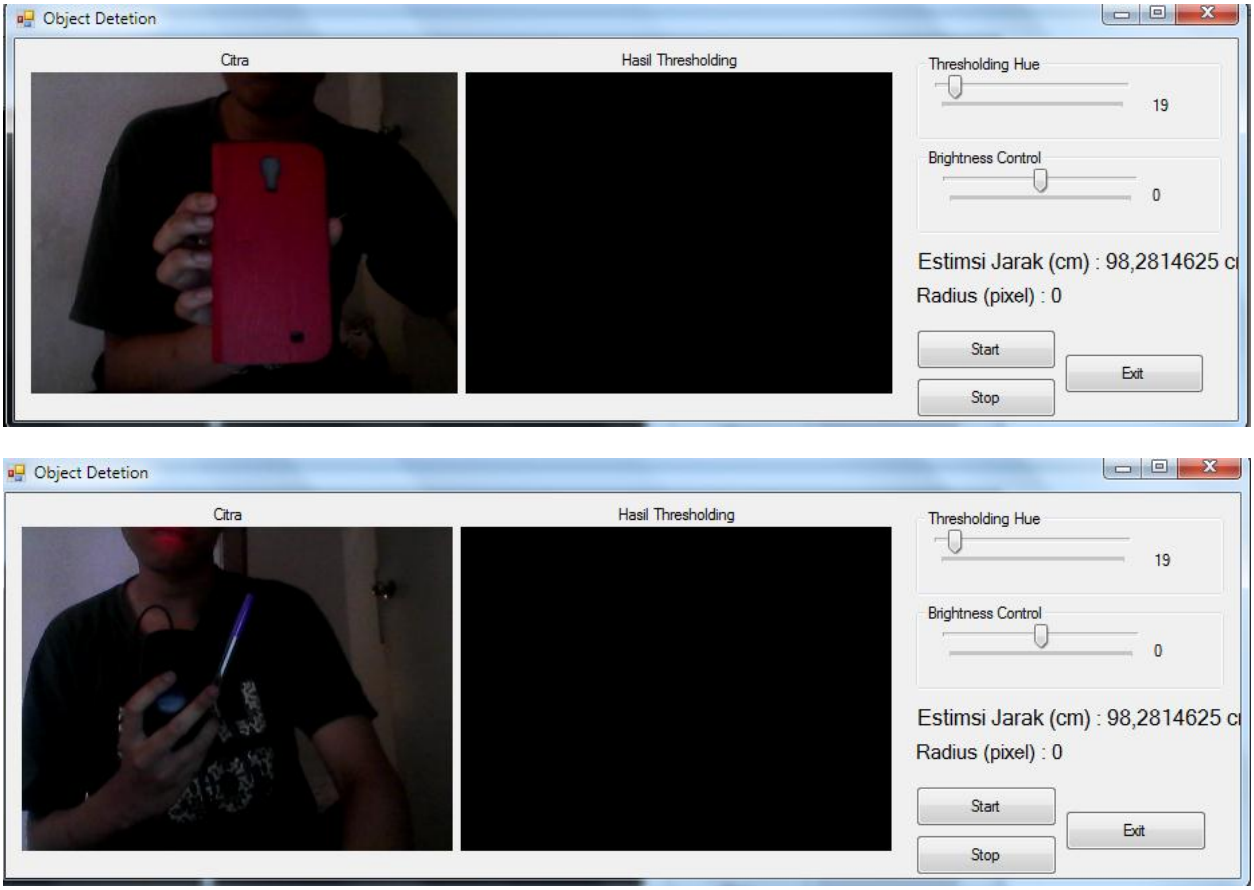
Sedangkan untuk tracking objek, hasil binary image tersebut diberi tanda dengan lingkaran pada daerah dengan nilai piksel 255(warna putih). Untuk memnentukan tanda ini, maka harus ditentukan titik pusat lingkaran dan radius lingkaran penanda. Untuk titik pusat dihitung dengan mencari selisih koordinat pixel terendah dan tertinggi dibagi 2 dan ditambahkan pada koordinat pixel terendah untuk pada daerah dengan nilai piksel 255. Untuk radius, karena objek dikondisi tegak, maka untuk men-*cover* seluruh objek, maka nilai radius dihitung dengan mencari selisih nilai koordinat pada sumbu y dibagi 2 pada daerah dengan nilai piksel 255. Koordinat dan radius ini digunakan untuk menentukan lingkaran pada binary image maupun citra hasil tangkapan camera.

Kombinasi dari segmentasi HSV dan object tracking inilah yang menghasilkan objek jika berada didepan camera akan terdeteksi dengan ciri terdapat lingkaran yang mengikut posisi dari objek.tersebut. Berikut gambar jika objek berupa back case handphone berwarna kuning diletakkan di depan camera.



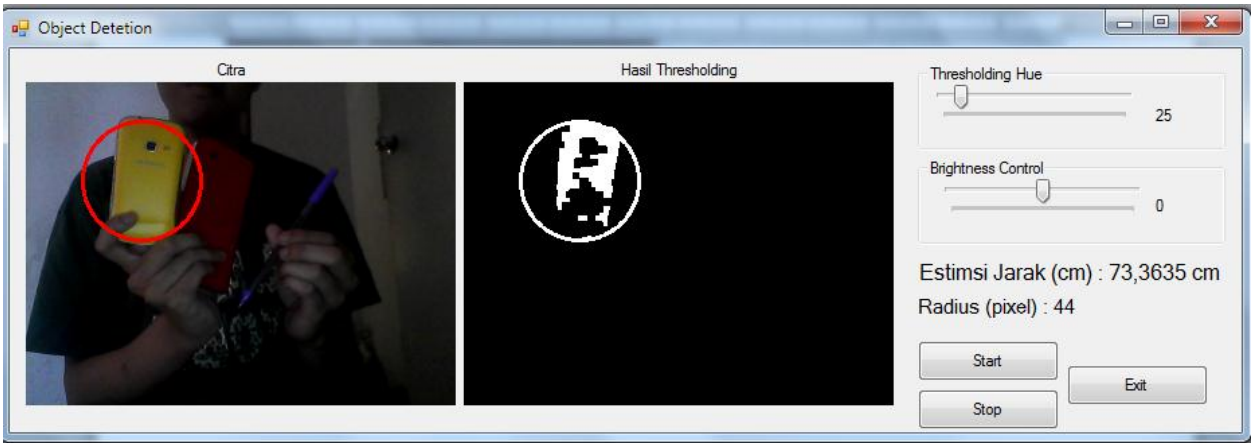
Gambar 3.1 Saat objek berwarna kuning berada di depan camera

Dilakukan percobaan dengan objek lain tidak berwarna kuning dan dengan jumlah lebih dari 1 (2), maka tidak ada objek yang terdeteksi. Berikut gambar hasil percobaan.



Gambar 3.3 Percobaan dengan objek tidak berwarna kuning

Dilakukan juga percobaan dengan beberapa objek, dengan salah satu diantaranya objek berwarna kuning, maka objek yang berwarna kuning tersebut terdeteksi. Berikut gambar hasil percobaan.



Gambar 3.4 Percobaan dengan banyak objek

Percobaan juga dilakukan untuk 2 objek dengan warna yang hamper sama. Untuk kasus yang ini kedua objek dianggap sebagai sebuah objek karena untuk menentukan penanda lingkaran berdasarkan daerah piksel bernilai 255, dan kedua objek memiliki daerah dengan nilai tersebut, sehingga kedua objek masuk dalam lingkaran penanda.

Gambar 3.5 Percobaan dengan 2 objek berwarna sama

Estimasi Jarak menggunakan single camera

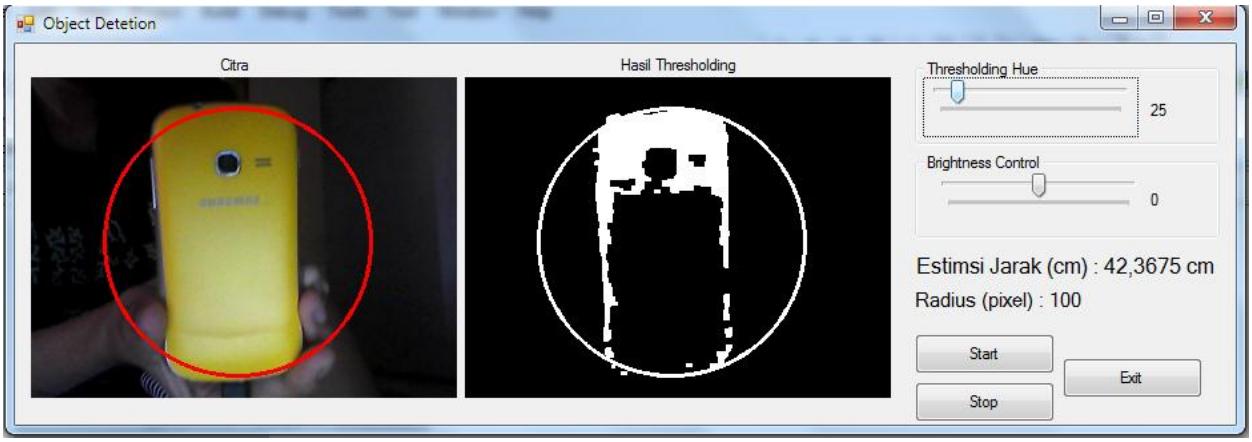
Untuk estimasi jarak digunakan persamaan (1.4) dengan nilai c0,c1,c2 dihitung dengan persamaan (1.5). Berikut perhitungann matriksnya berdasarkan data yang didapat.

$$\begin{pmatrix} c0 \\ c1 \\ c2 \end{pmatrix} = \begin{pmatrix} 1 & 100 & 10000 \\ 1 & 64 & 4096 \\ 1 & 55 & 3025 \\ 1 & 47 & 2209 \\ 1 & 37 & 1369 \\ 1 & 31 & 961 \\ 1 & 25 & 625 \\ 1 & 22 & 484 \end{pmatrix} \begin{pmatrix} 20 \\ 30 \\ 35 \\ 40 \\ 50 \\ 60 \\ 70 \\ 80 \end{pmatrix}$$

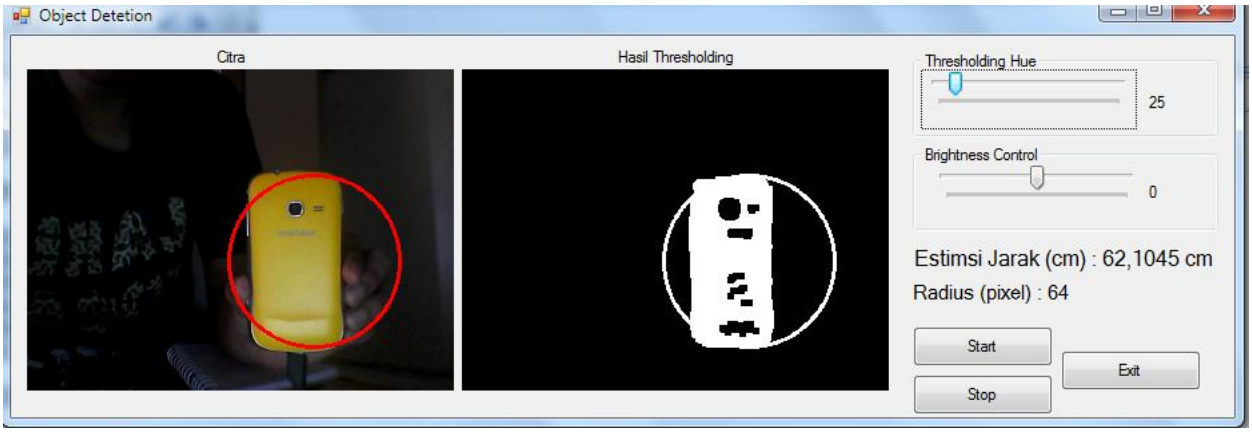
Dari perhitungan diatas didapat c0 = 98,8725; c1=-1,18260; dan c2 = 0,00105. Nilai ini dimasukan kedalam persamaan (1.5). sehingga persamaan yang dipergunaan dalam program menjadi:

$$D = 0{,}00105\ x\ A^2 - 1{,}18260\ x\ A + 98{,}875 \qquad (3.1)$$

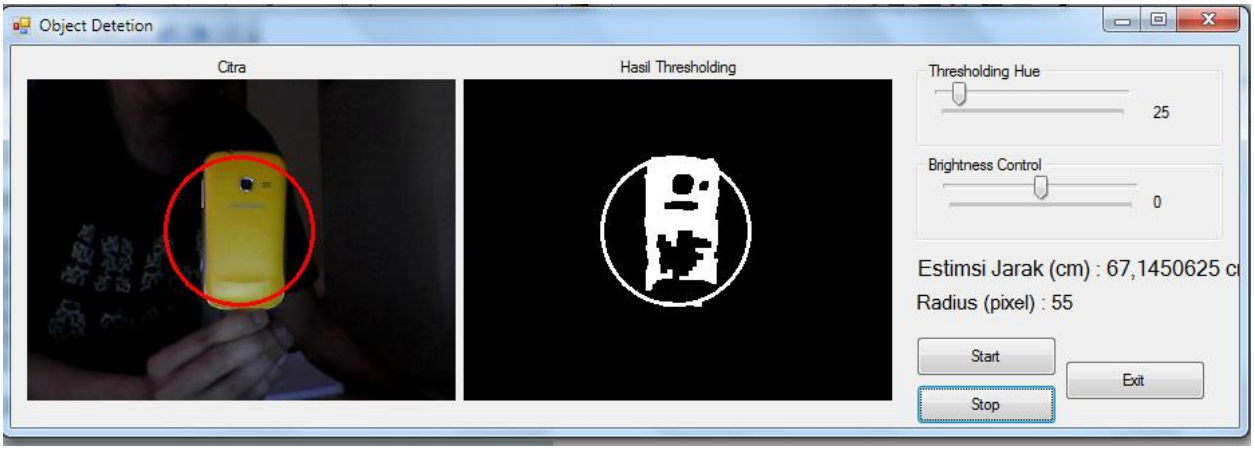
Percobaan dilakukan dengan objek yang sama dengan melakukan deteksi objek, yaitu back case handphone berwarna kuning. Beikut gambar hasil estimasi jarak.



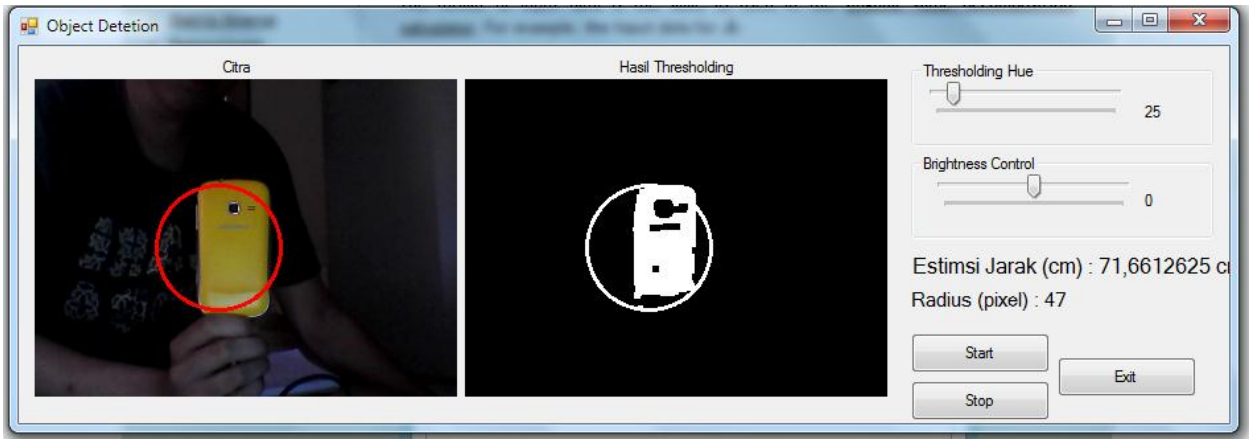
(a)



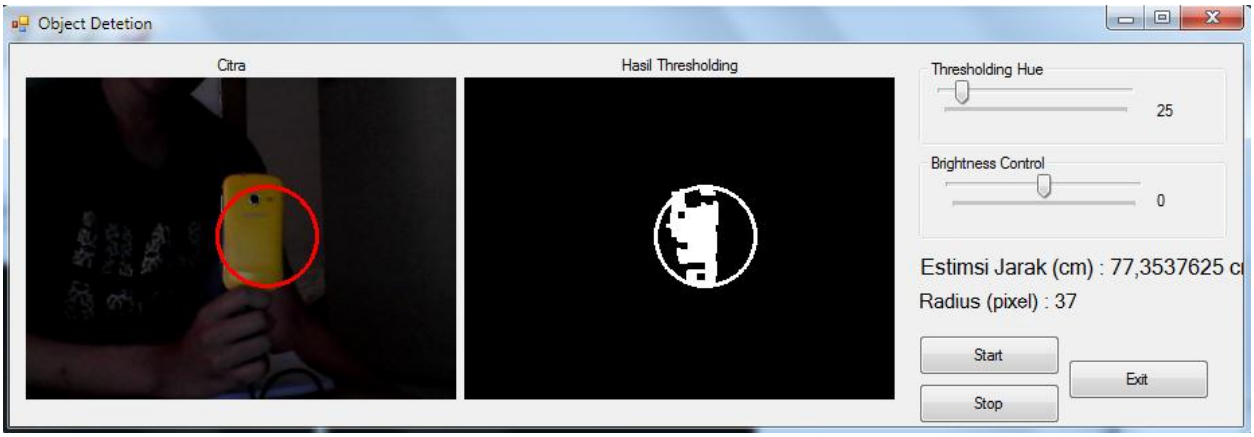
(b)



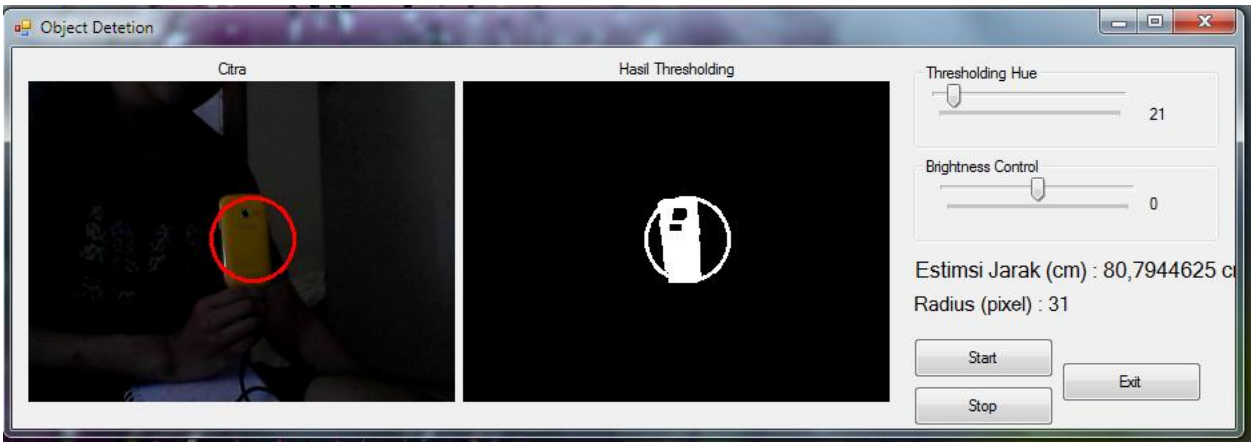
(c)


Object Detetion
Citra
Hasil Thresholding
Thresholding Hue
25
Brightness Control
0
Estimsi Jarak (cm) : 71,6612625 c
Radius (pixel) : 47
Start
Stop
Exit

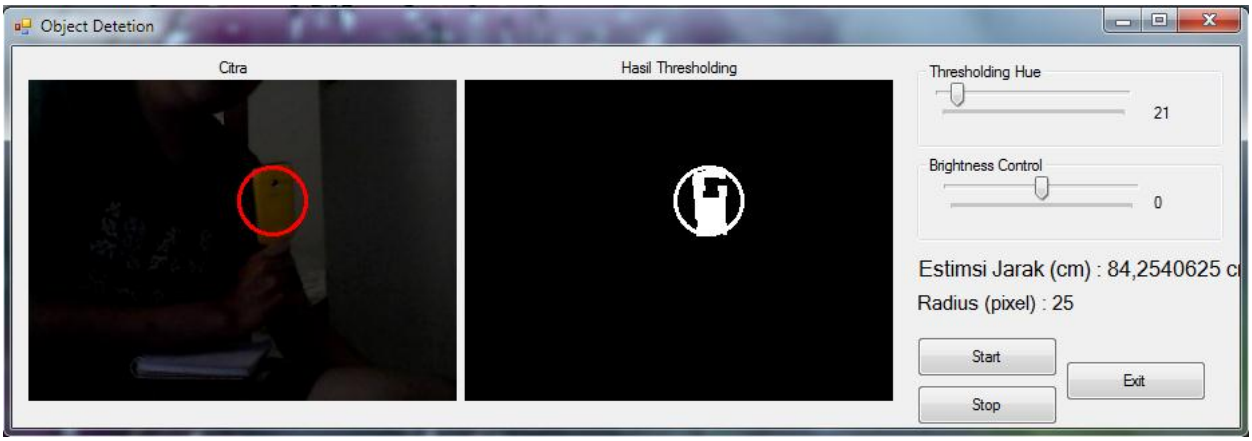
(d)


Object Detetion
Citra
Hasil Thresholding
Thresholding Hue
25
Brightness Control
0
Estimsi Jarak (cm) : 77,3537625 c
Radius (pixel) : 37
Start
Stop
Exit

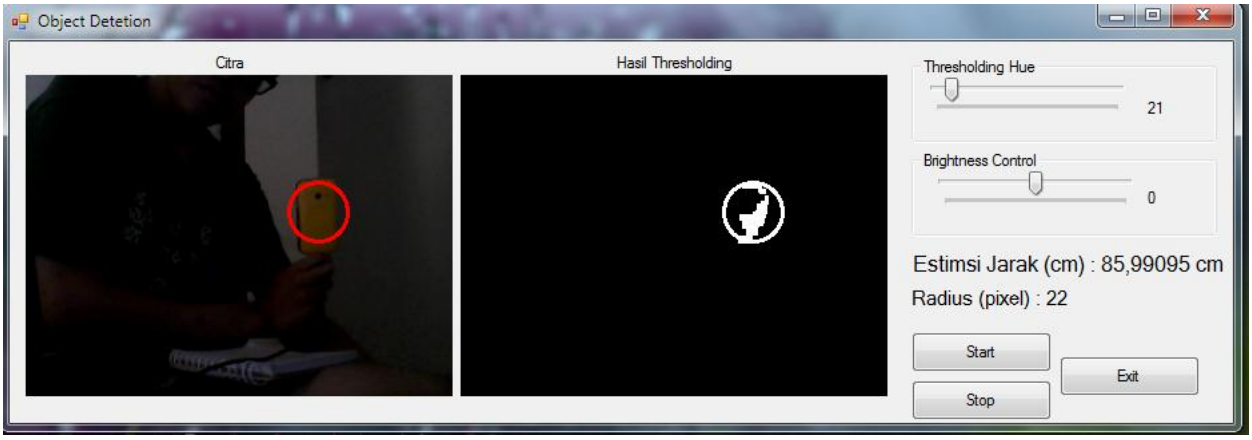
(e)


Object Detetion
Citra
Hasil Thresholding
Thresholding Hue
21
Brightness Control
0
Estimsi Jarak (cm) : 80,7944625 c
Radius (pixel) : 31
Start
Stop
Exit

(f)



(g)



(h)

Gambar 3.6 a,b,c,d,e,f,g,h Percobaan estimasi dengan berbagai jarak uji

Dari persamaan (1.5), estimasi jarak objek ditentunkan berdasarkan besar radius dalam piksel lingkaran yang menandai objek, semakin besar radiusnya maka semakin dekat jaraknya dan semakin kecil radiusnya maka semakin besar jarak benda. Radius ini digunakan sebagai representasi tinggi objek. Dari hasil estimasi diatas maka dapat diperbandingkan dengan jarak sebenarnya yang telah diukur. Berikut tabel data tersebut.

Tabel 3.1 Hasil Perbandingan Estimasi Jarak terhadap jarak sebenarnya

Jarak sebenarnya (cm)	Radius (piksel)	Estimasi jarak (cm)	Error (%)
20	100	42,3675	52,794
30	64	62,1045	51,69432
35	55	67,145	47,874
40	47	71,6613	44,18187
50	37	77,3537	35,36185
60	31	80,80	25,74257
70	25	84,254	16,91789
80	22	85,99	6,965926

Dari data diatas dapat dilihat bahwa terdapat nilai error untuk setiap nilai estimasi jarak dibanding dengan jarak sebenarnya. Besar error ini juga berbeda untuk setiap nilainya dengan rentang error 6,97% - 52,794%. Dari data diatas, saat jarak 80 cm dapat dibaca dengan pendekatan yang paling baik yaitu 85,99 cm dengan nilai error 6,97% , sedangkan untuk jarak 20 cm dibaca dengan pendekatan paling buruk yaitu 42,3675 cm dengan nilai error 52,8%. Dari data tersebut pada jarak tertentu, hasil estimasi menunjukkan nilai yang mendekati nilai sebenarnya namun pada jarak tertentu juga menghasilkan nilai yang berbeda jauh. Dari data tersebut dapt dilihat semakin besar jarak dari camera semakin kecil nilai errornya, dan semakin dekat jarak dari camera semakin besar nilai errornya.

Bab 4

Kesimpulan

Dari kegiatan yang dilakukan dapat disimpulkan bahwa, Segemntasi HSV yang dilakukan dengan thresholding Hue (dalam percobaan ini menggunakan nilai antara 16-30) dapat digunakan untuk memisahkan objek tertentu terhadap objek lainnya dan sensitif terhapad warna (bergantung terhadap warna) sehingga dapat digunakan sebagai detector objek yang dibedakan melalui warna. Dalam kegiatan ini dipergunakan warna kuning, sehingga yang terdeteksi hanya yang berwarna kuning, tetapi tidak dapat mendeteksi objek berwarna lain.

Estimasi jarak menggunakan single camera dilakukan dengan perbandingan radius lingkaran yang men-*cover* objek sebagai representasi tinggi objek (setelah dilakukan perhitungan dengan pseudoinverse) dengan jaraknya. Metode ini tidak dapat digunakan untuk mengestimasi jarak karena memiliki rentang nilai error yang besar, yaitu berkisar 6,97% - 52,794%. Dalam percobaan yang dilakukan nilai error tertinggi didapat saat objek berjarak 20 cm dengan pembacaaan estimasi 52,794 cm dengan error 52,794% dan nilai error terendah didapat saat objek berjarak 80 cm dengan pembacaan estimasi 85,99 dengan error 6,97%. Semakin jauh jarak dari camera nilai error semakin kecil, sebaliknya semakin dekat jarak dengan camera semakin besar nilai errornya.

Daftar Pustaka

Munir, Rinaldi.(2004),"*Pengolahan Citra Digital dengan pendekatan Algoritmik"*, Penerbit Informatika, Bandung.

Pambudi , Wahyu Setyo. Salamah, Irma. Tompunu , Alan Novi.(2011) , " Deteksi dan Estimasi Jarak Obyek Menggunakan Single Camera Dengan Model Segmentasi HSV", Seminar Nasional Teknoin 2011, ISBN: 978-979-96964-8-9.