



**UAS KOMPUTER VISION
MEMBUAT RESUME 3 JURNAL TERKAIT COMPUTER VISION**

Nama : Nur ahmad riski
Kelas : 4G
Nim : 18041019
MK : UAS KOMPUTER VISON

**POLITEKNIK HARAPAN BERSAMA TEGAL
TAHUN 2020-2021**

JURNAL 1

PENYAKIT KULIT BERBASIS COMPUTER VISION MELALUI DETEKSI TEPI

Hapnes Toba¹, Antonius Hendrik², Riskadewi³

¹Program Studi D3 Teknik Informatika, ^{2,3}Jurusan S1 Teknik
Informatika Fakultas Teknologi Informasi, Universitas Kristen
Maranatha

Jl. Suria Sumantri No. 65, Bandung 40164

¹hapnestoba@it.maranatha.edu, ^{2,3}antonriskalive@gmail.com

Abstrak

Penglihatan manusia dapat melakukan hal-hal yang menakjubkan seperti mengenali objek, navigasi dalam menghindari rintangan, ataupun mengenali *mood* di dalam sebuah adegan. Lain halnya dengan komputer yang memerlukan sensor guna menerima persepsi dari lingkungan dan program komputer yang berfungsi sebagai pemroses data dari sensor tersebut. *Computer vision* merupakan sebuah konsep yang memanfaatkan teknik-teknik pemrosesan citra untuk membuat keputusan berdasarkan citra yang didapat dari sensor. Dalam penelitian ini dikembangkan sebuah basisdata yang terintegrasi dengan kamera sebagai sensor untuk mengenali berbagai penyakit kulit melalui deteksi tepi. Untuk mendeteksi tepi dari satu atau lebih objek, digunakan operator Canny, Prewitt, Sobel, dan Roberts. Hasil deteksi tepi tersebut kemudian akan dicocokkan dengan fitur-fitur yang tersimpan dalam basisdata untuk menentukan penyakit kulit yang teridentifikasi. Perangkat lunak untuk deteksi tepi diimplementasikan dengan Microsoft Visual Studio 2010 dan OpenCV 2.4. Hasil penelitian menunjukkan bahwa untuk dapat mengenali penyakit kulit secara lebih baik diperlukan pengurangan *noise* dengan menggunakan filter Gaussian dan pemecahan (*split*) citra warna ke dalam masing-masing saluran warna (merah, hijau, dan biru) dengan ukuran 8 bit. Hasil evaluasi pencocokan citra dengan metode *cross-correlation* menunjukkan bahwa operator Canny adalah operator yang paling memenuhi kriteria penandaan tepi, yaitu: tingkat kesalahan yang rendah, lokasi yang benar, dan waktu respon yang minimum.

1. Pendahuluan

Penglihatan manusia dapat melakukan hal-hal yang menakjubkan seperti mengenali orang/objek, navigasi dalam menghindari rintangan, ataupun mengenali *mood* di dalam sebuah adegan. Lain halnya dengan mesin, yang dalam konteks ini adalah komputer. Untuk melakukan mimikri terhadap penglihatan manusia, komputer memerlukan sensor yang berfungsi layaknya mata pada manusia dan program komputer yang berfungsi sebagai pemroses data dari sensor. *Computer vision* merupakan ilmu yang menggunakan image processing untuk membuat keputusan berdasarkan citra yang didapat dari sensor [5, 9, 10]. Dengan kata lain, *computer vision* bertujuan untuk membangun sebuah mesin pandai yang dapat “melihat”. Kerangka kerja umum yang biasa dilakukan dalam *computer vision* adalah: proses akuisisi citra, pra pemrosesan, ekstraksi fitur, deteksi atau segmentasi citra, pemrosesan tingkat tinggi, dan terakhir pengambilan keputusan.

Tahap akuisisi citra adalah tahap untuk mendapatkan citra dari sensor. Tahap pra pemrosesan adalah tahap pemrosesan awal terhadap citra untuk memperbaiki kualitas citra, misalnya pengurangan *noise* (informasi yang salah pada citra)

dan *contrast enhancement* (perbaikan kontras pada citra). Tahap ekstraksi fitur adalah tahap ekstraksi fitur dari citra, misalnya titik, garis, dan tepi (*edge*). Tahap deteksi/segmentasi adalah tahap mendeteksi perbedaan kecerahan pada citra untuk mendapatkan lokasi atau posisi dari suatu objek. Kemudian, dilakukan pengenalan bentuk berdasarkan kriteria dan deskripsi objek yang telah ditentukan sebelumnya. Pemrosesan tingkat tinggi adalah tahap pemrosesan terhadap sebagian kecil dari data, misalnya sebagian dari titik dalam bagian citra tertentu. Contoh pemrosesan tingkat tinggi adalah *image recognition* (pengenalan citra) dan *image registration* (registrasi citra). Pengambilan keputusan adalah tahap dimana pengambilan keputusan akhir dibutuhkan pada aplikasi tertentu, misalnya pengambilan keputusan lulus/tidak lulus pada aplikasi otomatisasi inspeksi (*pass/fail on automatic inspection applications*) [5, 9, 10].

Terdapat beberapa penelitian *computer vision* untuk mendeteksi penyakit kulit, misalnya untuk mendeteksi penyakit *dermatitis*, *eczema*, dan *utricaria*. Namun pada umumnya metode yang digunakan adalah *texture features* yang merupakan metode analisis tekstur berbasis statistik [2, 6]. Masih sedikit sekali penelitian yang memanfaatkan

deteksi tepi sebagai fitur pengenalan. Menurut hipotesis kami, deteksi tepi dapat dimanfaatkan untuk mengenali bagian kulit yang sehat dan tidak setelah melalui pemrosesan citra. Dalam penelitian ini ditekankan pada proses untuk menemukan metode akuisisi citra yang baik sampai dengan menghasilkan kumpulan fitur untuk memperoleh operator deteksi tepi dengan performa terbaik sehingga dapat dipakai untuk mengenali obyek.

Permasalahan yang akan ditelaah dalam penelitian ini adalah sebagai berikut:

1. Prosedur apa saja yang perlu dilakukan untuk mendapatkan citra digital yang dapat diolah untuk *computer vision*?
2. Prosedur pra-pemrosesan apa saja yang harus dilakukan pada citra digital?
3. Hal apa saja yang dapat menentukan performa operator deteksi tepi?
4. Bagaimana membangun basis data citra untuk penyakit kulit?

Kerangka kerja dalam penelitian ini dapat dilihat Gambar 1. Proses dalam penelitian adalah: mengambil citra dari *webcam*, pra-pemrosesan, dan mendeteksi tepi dari citra. Mula-mula citra diambil dari *webcam*. Kemudian akan dilakukan pra- pemrosesan, yaitu pengurangan *noise*. Tahap selanjutnya adalah proses deteksi tepi pada citra tersebut dengan menggunakan operator Sobel, Prewit, dan Roberts. Hasil dari masing-masing operator akan dibandingkan untuk menentukan operator mana yang paling memenuhi kriteria penandaan tepi untuk mengenali penyakit kulit.

Setelah itu akan dibuat basis data citra yang berisi jenis penyakit dan hasil deteksi tepi dengan operator yang optimal. Basisdata citra berisi sejumlah citra penyakit kulit yang diambil melalui mesin pencari citra dari Google untuk 16 macam penyakit kulit sebagaimana diberikan pada Gambar

1. Untuk setiap penyakit diambil 30 citra dengan resolusi minimal 800x600 *dot-per-inch* (dpi).

2. Operator Deteksi Tepi dan Perangkat

Gradien dari sebuah fungsi citra adalah dasar dari banyak operator deteksi tepi klasik. Pada prakteknya, operator deteksi tepi hanya berbeda pada tipe filter yang dipakai untuk mengestimasi komponen gradien dan cara mengkombinasikan komponen-komponen tersebut. Kekuatan dari titik-titik tepi dan arah dari tepi dimuat pada fungsi gradien dan dapat dengan mudah dihitung dari komponen berarah.

2.1 Operator Prewitt dan Sobel

Operator Prewitt dan Sobel menggunakan filter linear yang memperluas ketiga baris dan kolom yang berdekatan untuk melawan sensitifitas *noise* dari operator gradien sederhana (satu garis/kolom) [2, 7]. Operator Prewitt menggunakan filter pada persamaan 1 untuk menghitung rata-rata komponen gradien yang melewati sepanjang baris atau kolom

yang bertetangga. P_x melakukan penghalusan (menggunakan filter kotak) terhadap tiga baris sebelum menghitung gradien x , dan P_y melakukan penghalusan terhadap tiga kolom sebelum menghitung gradien y .

$$H_x^P = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \text{ dan } H_y^P = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad (1)$$

Filter untuk operator Sobel hampir identik, tetapi bagian penghalusan menetapkan bobot yang lebih tinggi untuk baris dan kolom yang berada di tengah. Operator Sobel menggunakan filter pada persamaan 2.

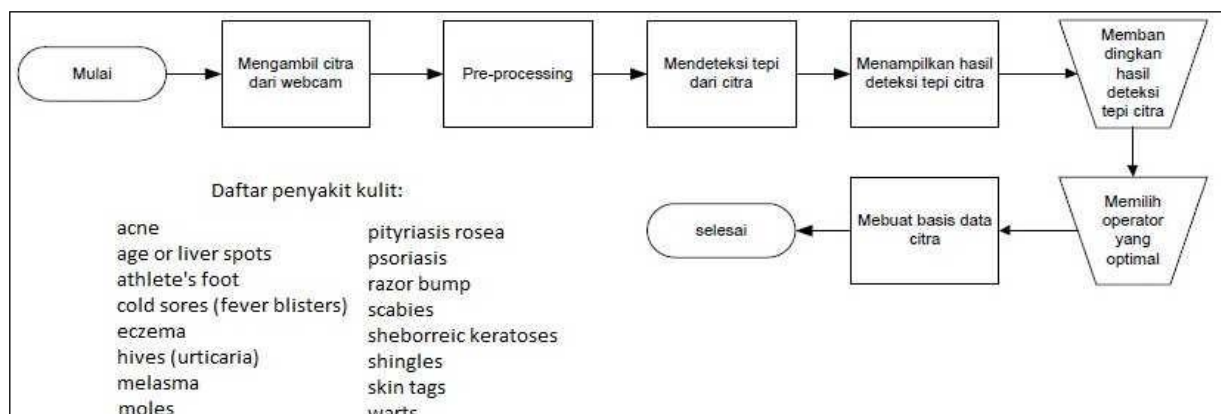
$$H_x^S = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \text{ dan } H_y^S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (2)$$

2.1 Operator Roberts

Operator Roberts merupakan filter yang paling sederhana dan paling tua. Operator Roberts menggunakan filter matriks berukuran 2x2 untuk mengestimasi arah gradien sepanjang diagonal citra. Operator Roberts menggunakan filter pada persamaan 3 [2, 7].

$$H_1^R = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \text{ dan } H_2^R = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad (3)$$

Gambar 1. Metodologi Penelitian dan Daftar Penyakit Kuli



Filter Roberts dapat merespons tepi diagonal tetapi tidak memilih orientasi, kedua filter menunjukkan hasil yang kuat, yang melingkupi sudut dengan jangkauan yang luas. Kekuatan tepi dihitung dengan mengukur panjang vektor 2D yang dihasilkan, serupa dengan perhitungan gradien, tetapi dengan komponennya dirotasikan 45° .

2.2 Operator Canny

Operator Canny adalah operator deteksi tepi yang menggunakan algoritma banyak tahap (*multi-stage*) untuk mendeteksi banyak tepi dari suatu citra. Tahapan umum dalam algoritma Canny meliputi: pengurangan *noise*, mencari intensitas gradien dari citra, menerapkan *non-maximum suppression*, dan menelusuri tepi pada citra dan menentukan ambang histeresis (*hysteresis thresholding*) [2, 7].

2.3 OpenCV

OpenCV (*Open Source Computer Vision*) adalah sebuah pustaka perangkat lunak *computer vision* dan *machine learning* yang bersifat terbuka [8]. OpenCV dirancang sebagai infrastruktur umum untuk aplikasi *computer vision* dan untuk mempercepat penggunaan persepsi mesin (*machine perception*) dalam produk komersial. OpenCV berlisensi BSD (*Berkeley Software Distribution*), sehingga memudahkan bagi pelaku bisnis dan akademisi untuk memanfaatkan dan mengubah kode.

Pustaka OpenCV memiliki lebih dari 2.500 algoritma optimal, yang mencakup sekumpulan algoritma *computer vision* dan pembelajaran mesin bertipe klasik maupun terkini. Algoritma-algoritma ini dapat digunakan untuk berbagai proses dalam *computer vision*, seperti:

- mendeteksi dan mengenali wajah;
- mengidentifikasi objek;
- mengklasifikasikan tindakan manusia dalam video;
- melacak gerakan kamera;
- melacak obyek yang bergerak;
- ekstrak model 3D dari obyek;

- menghasilkan 3D *point clouds* dari kamera stereo;
- menggabungkan citra untuk menghasilkan citra dengan resolusi tinggi dari seluruh adegan;
- menemukan citra yang sama dari basisdata citra;
- menghapus mata merah dari citra yang diambil menggunakan lampu kilat;
- mengikuti gerakan mata;
- mengenali pemandangan dan membuat penanda (*marker*) untuk melapisi (*overlay*) penanda dengan *augmented reality*, dan lain-lain.

3. Perancangan Sistem

Perancangan, skenario dan pemodelan sistem dapat dilihat pada diagram *use case* dan diagram kelas pada Gambar 2(a) dan 2(b). Skenario pemanfaatan sistem secara garis besar adalah sebagai berikut:

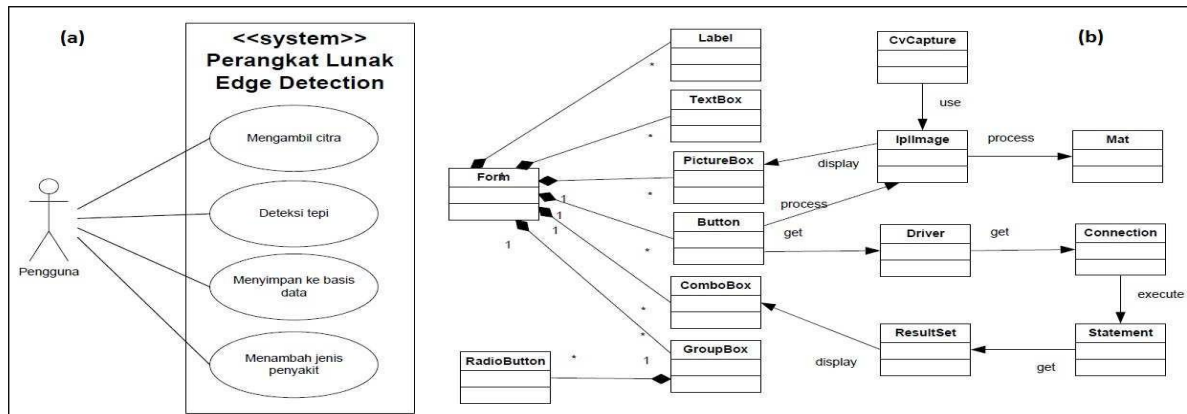
3.1 Mengambil Citra

Fitur ini ditujukan untuk mengambil citra melalui *webcam* yang difungsikan sebagai sensor, layaknya mata pada manusia. Hasil citra di-*capture* tersebut akan dapat disimpan dalam format jpeg dan dapat dicocokkan dengan data penyakit pada basisdata. Lebih jauh, fitur ini di masa depan diharapkan dapat dijadikan juga sebagai cara untuk mendapatkan citra, dan menggantikan peran citra statis yang digunakan dalam penelitian kali ini.

3.2 Deteksi Tepi

Fitur ini ditujukan untuk melakukan deteksi tepi berdasarkan masukan yang diterima melalui *webcam* atau dari gambar statis. Kerangka kerja *computer vision* secara lengkap dilakukan dalam sub-sistem ini, yaitu pra-pemrosesan, pemilihan filter deteksi tepi (bandingkan juga dengan penjelasan pada bagian 2 di atas), penggabungan kembali saluran warna dari hasil deteksi tepi, serta menampilkan hasilnya.

Gambar 2. Skenario Sistem (a) dan Diagram Kelas (b)



Tabel 1. Contoh metadata penyakit kulit dalam basisdata

<i>Disease Name</i>	<i>Age</i>	<i>Gender</i>	<i>History</i>	<i>Site</i>	<i>Lesion</i>	<i>Surface</i>	<i>Color</i>	<i>Itchiness</i>
<i>Eczema</i>	<i>infant</i>	<i>both</i>	<i>family</i>	<i>hair line, eyes region, nose-cheek region, mouth region, jaw region, neck region, elbow flexure, hand, knee, feet</i>		<i>scaly</i>	<i>brown, red</i>	<i>itchy</i>
<i>Moles</i>	<i>all</i>	<i>both</i>	<i>personal</i>	<i>all, except nail</i>	<i>flat, raised solid</i>		<i>black, blue, brown, red, white</i>	<i>non-itchy</i>
<i>Shingles (Herpes Zoster)</i>	<i>all</i>	<i>both</i>	<i>personal</i>	<i>hair line, eyes region, nose-cheek region</i>	<i>fluid filled</i>	<i>crust</i>		<i>itchy</i>

1.1 Menyimpan ke Basisdata

Fitur ini ditujukan untuk melakukan pengayaan data penyakit kulit di dalam basisdata. Pengguna akan dapat menyimpan informasi sesuai hasil operator deteksi tepi ditambah dengan metadata (lihat juga penjelasan pada bagian 3.4) tentang penyakit kulit yang dimaksudkan dalam citra. Contoh tampilan deteksi tepi dan masukan metadata dapat dilihat pada Lampiran A dan B.

1.2 Menambahkan Jenis Penyakit

Fitur ini ditujukan untuk mengisi metadata penting terkait suatu penyakit kulit, yang akan disimpan bersama dengan data citra yang diperoleh pada bagian 3.3. Berbekal pada metadata ini, deteksi penyakit kulit diharapkan akan lebih akurat dalam operasional di masa mendatang, karena akan dapat difungsikan sebagai tabel keputusan. Adapun contoh metadata yang dimaksudkan untuk tiga penyakit kulit dapat dilihat pada Tabel 1 [1, 4].

4. Eksperimen Performa

Eksperimen bertujuan menguji basis data citra yang dihasilkan oleh masing-masing operator deteksi tepi. Eksperimen dilakukan dengan membandingkan hasil deteksi tepi citra dalam basisdata citra dengan hasil deteksi tepi dari *sample* citra. Citra yang dipakai untuk eksperimen beserta jenis modifikasinya dapat dilihat pada Tabel 2. Dalam eksperimen, modifikasi citra diasumsikan sebagai *noise* yang berasal dari citra sesungguhnya yang diambil dari kamera. Perbandingan dilakukan dengan metode *Cross-Correlation* [3].

Cross-Correlation adalah ukuran kemiripan dari dua buah gelombang (*waveform*) sebagai fungsi *time lag* yang diaplikasikan pada salah satu gelombang. Nama lain dari metode *Cross-Correlation* adalah *Sliding Dot Product* atau *Sliding Inner Product*.

Tabel 2. Hasil pengujian dengan *Cross-Correlation* pada basisdata untuk 10 penyakit kulit pada setiap operator

No.	Nama Penyakit	Modifikasi Citra	Canny	Prewit t	Roberts	Sobel
1.	<i>Acne</i> = jerawat	<i>Monochromatic noise</i>	95,13 %	20,50 %	16,34%	2,00%
2.	<i>Age or Liver Spots</i> = bintik karena usia	<i>Exposure</i>	88,74 %	23,50 %	16,07%	2,04%
3.	<i>Athlete's Foot</i> = jamur pada kaki	<i>Blur</i>	85,64 %	33,67 %	25,13%	4,30%
4.	<i>Cold Sores (Fever Blisters)</i> = sariawan	<i>Crop</i>	87,02 %	23,05 %	16,39%	3,12%
5.	<i>Eczema</i> = eksim	<i>Distort</i>	77,03 %	24,02 %	16,48%	2,66%
6.	<i>Hives (Urticaria)</i> = gatal-gatal	<i>Gray scale</i>	99,42 %	24,35 %	16,57%	3,75%
7.	<i>Melasma (Pregnancy Mask)</i> = guratan pada perut saat mengandung	<i>Invert color</i>	84,48 %	23,05 %	17,73%	2,26%
8.	<i>Moles</i> = tahi lalat	<i>Colored noise</i>	74,19 %	20,83 %	15,23%	2,34%
9.	<i>Pityriasis Rosea/Rosacea</i> = kulit kemerahan	<i>Contrast increasing</i>	87,87 %	18,10 %	14,54%	2,11%
10.	<i>Razor Bump</i> = kerusakan kulit karena mencukur	<i>Sharpen</i>	79,20 %	19,59 %	16,00%	2,08%

Sebagai fungsi diskret, *Cross-Correlation* didefinisikan dalam persamaan 4.

$$(f \star g) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau \quad (4)$$

Dimana f dan g adalah fungsi yang bertipe real, yang memiliki perbedaan hanya pada pergeseran sepanjang sumbu x . *Cross-Correlation* dipakai untuk mencari seberapa banyak g yang harus berpindah di sepanjang sumbu x agar f dan g menjadi identik. Rumus di atas pada dasarnya menggeser fungsi g sepanjang sumbu x , kemudian menghitung nilai integral untuk setiap operasi perkalian pada setiap piksel. Jika terjadi kecocokan (*match*), nilai $f \star g$ akan mencapai nilai maksimum, yaitu 100%.

5. Evaluasi Performa

Dari hasil pada Tabel 2, dihasilkan beberapa pembelajaran sebagai berikut:

- Sesuai dengan kompleksitas dan proses yang ada, algoritma Canny menghasilkan kecocokan yang paling tinggi, dan Sobel terendah.
- Perbedaan nilai akurasi antar penyakit tidak berbeda jauh untuk setiap operator. Hal ini mengindikasikan bahwa karakteristik 'tepi' antar penyakit tidak berbeda secara signifikan.
- Karakteristik penyakit kulit dalam basisdata didominasi oleh penyakit yang mementingkan perubahan warna dibandingkan bentuk, seperti: bintik-bintik kecil yang memiliki tepi sangat mirip antara satu dengan lainnya, sehingga sulit dideteksi dengan algoritma deteksi tepi dalam satu tahap. Menurut analisis kami, inilah salah satu alasan yang menyebabkan algoritma Canny berhasil mendapatkan kecocokan lebih baik dibanding algoritma lainnya.
- Karakteristik umum dimiliki paling nyata pada penyakit '*hives*', hal ini ditunjukkan dengan kenyataan bahwa dari 10 penyakit yang diuji pada Tabel 2, ada 8 penyakit yang mengarah pada '*hives*', dan hanya 2 penyakit yang tepat dideteksi, yaitu: '*age spot*', dan '*hives*'.

6. Kesimpulan dan Keberlanjutan

Kesimpulan yang dapat ditarik melalui penelitian ini adalah sebagai berikut:

1. Penelitian ini telah menghasilkan sistem yang bertujuan untuk menambahkan pengetahuan dan pengolahan citra hasil deteksi tepi ke dalam basisdata citra.
2. Sistem telah dapat mengintegrasikan perangkat lunak dan keras (*webcam*) yang bersifat generik, tidak bergantung pada jenis ataupun *driver* tertentu.
3. Basisdata citra untuk penyakit kulit telah berhasil dibangun dengan menggunakan sistem basisdata MySQL. Basisdata ini berisi pengetahuan (metadata) tentang penyakit kulit dan pengolahan citra hasil deteksi tepi.

4. Prosedur pra-pemrosesan yang perlu dilakukan pada pemrosesan citra digital adalah:
 - a. Pengurangan *noise* dengan menggunakan filter Gaussian;
 - b. Pemecahan (*split*) citra warna ke dalam setiap komponen warna (merah, hijau, dan biru) masing-masing dengan ukuran 8 bit.
 5. Hal-hal yang dapat menentukan performa operator deteksi tepi adalah tingkat kekaburan citra dan pengurangan *noise*.
 6. Operator Canny adalah operator yang paling memenuhi kriteria penandaan tepi, yaitu: tingkat kesalahan yang rendah, lokasi yang benar, dan respon minimum, serta memiliki tingkat kecocokan paling baik.
- Saran pengembangan untuk tahap penelitian selanjutnya adalah:
1. Penggunaan kamera yang memiliki resolusi yang lebih baik akan menghasilkan citra yang memiliki ketajaman yang lebih baik.
 2. Memperkaya fitur deteksi tepi dengan fitur-fitur kontras serta warna untuk jenis-jenis penyakit yang lebih mengedepankan perubahan warna dibandingkan bentuk ataupun tekstur, seperti: *age/liver spot*, *hives*, *razor bump*, dan *rosacea*, sehingga dapat memperbaiki performa pencocokan.
 3. Penggunaan metode pencocokan yang mampu mendeteksi *key point interest*.

Daftar Pustaka:

- [1] Ashton, R. & Leppard, B., 2004, *Differential Diagnosis in Dermatology 3rd ed.*, Abington: Radcliffe Publishing Ltd.
- [2] Bin, L. & Mehdi, S.Y., 2012, *Comparison for Image Edge Detection Algorithms*, IOSR Journal of Computer Engineering.
- [3] Bracewell, R., 1965, *Pentagram Notation for Cross Correlation*, The Fourier Transform and Its Applications, New York: McGraw-Hill.
- [4] Buxton, P.K, 2003, *ABC of Dermatology 4th ed.*, London: BMJ Publishing Group.
- [5] Forsyth, D.A. & Ponce, J., 2011, *Computer Vision: A Modern Approach 2nd ed.*, Pearson Education, Ltd.
- [6] Mittra, A.K. & Parekh, R., 2011, *Automated Detection of Skin Diseases Using Texture Features*, International Journal of Engineering Science and Technology.
- [7] Munir, R., 2004, *Pengolahan Citra Digital*, Bandung: Penerbit Informatika.
- [8] OpenCV, 2012, *Open Source Computer Vision*, <http://opencv.org/>, diakses: November 2013.
- [9] Shapiro, L.G. & Stockman, G.C., 2001, *Computer Vision*, Prentice Hall
- [10] Szeliski, R., 2011, *Computer Vision: Algorithms and Applications*, Springer.

JURNAL 2

SEGMENTASI CITRA UNTUK DETEKSI OBJEK WARNA PADA APLIKASI PENGAMBILAN BENTUK CITRA RECTANGLE

Asep Nana H^[1], M. Ichwan^[1], I Made Santika Putra

^[1]Jurusan Teknik Informatika, Fakultas Teknologi Industri
Institut Teknologi Nasional Bandung

asepnana@itenas.ac.id, ichwan@itenas.ac.id, imadesantikaputra@gmail.com

ABSTRAK

Pengambilan citra terkadang tidak sesuai dengan yang dibutuhkan, sehingga diperlukan proses pengolahan citra untuk mendapatkan bentuk citra yang diinginkan. Dengan menanamkan kecerdasan pada sebuah kamera, pengambilan suatu citra dengan bentuk tertentu dapat dilakukan sehingga mengurangi beban waktu dan memfasilitasi pengguna dalam proses pengambilan gambar. Segmentasi citra merupakan proses pengambilan informasi dari citra dalam pencarian citra yang serupa seperti warna. Warna dapat dijadikan input dalam penggambaran daerah yang diinginkan (*Region Of Interest*) melalui proses deteksi warna dan *tracking* warna, sehingga dapat dilakukan pengambilan gambar dalam bentuk tertentu. Object Tracking adalah proses mengikuti suatu objek yang bergerak dan berpindah posisi. Computer Vision didefinisikan sebagai salah satu cabang ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali obyek yang diamati. Dalam mengenali obyek yang diamati dilakukan proses segmentasi citra dengan menggunakan euclidean color filtering, grayscale dan deteksi blob. Dalam implementasi segmentasi citra untuk deteksi objek warna untuk pengambilan bentuk citra rectangle, digunakan metode segmentasi dan center of gravity. Hasil dari pengujian sistem ini adalah objek warna dapat terdeteksi dan dijejaki dengan baik dalam jarak terbaik antara 40cm-80cm dan dengan intensitas cahaya terbaik antara 22lx-242lx.

Kata Kunci : *Pengolahan Citra, Computer Vision, Deteksi Warna, Segmentasi, Objek Tracking, Capture image, Center of Gravity.*

Latar Belakang

Webcam merupakan sebuah *device* yang dapat digunakan sebagai sensor dalam mendeteksi sebuah benda bergerak melalui proses pengolahan citra. Webcam juga dapat digunakan dalam pengambilan gambar (*capture image*).

Object Tracking adalah proses mengikuti suatu objek yang bergerak dan berpindah posisi. *Computer Vision* didefinisikan sebagai salah satu cabang ilmu pengetahuan yang mempelajari bagaimana komputer dapat mengenali objek yang diamati. Dalam proses pengenalan objek dan deteksi objek diperlukan pemisahan segmen tertentu pada suatu citra yang dikenal dengan proses segmentasi.

Segmentasi citra merupakan bagian dari proses pengolahan citra. Kegunaan segmentasi menurut Forsyth dan Ponce (2003) adalah pengambilan informasi dari citra seperti pencarian bagian mesin, pencarian manusia dan pencarian citra yang serupa. Secara umum pendekatan segmentasi citra yang sering digunakan adalah melalui pendekatan intensitas, pendekatan warna dan pendekatan bentuk (Rujikietgumjorn, 2008). Salah satu produk teknologi dalam proses pengolahan citra adalah *Aforge.Net Framework*.

Pada penelitian ini sistem yang dibangun adalah suatu aplikasi yang dapat mengambil sebuah bentuk citra *rectangle* dari hasil *Region Of Interest* (ROI) berdasarkan objek warna yang dideteksi dan dilacak. Pada Penelitian ini digunakan beberapa metode pelacakan (*tracking*) yakni dengan mengkombinasikan metode segmentasi dan *center of gravity* (COG) yang diharapkan dapat melacak pergerakan dari objek warna yang dideteksi.

Rumusan Masalah

Rumusan permasalahan yang ada pada penelitian ini adalah :

1. Bagaimana sistem dapat mendeteksi dan melacak pergerakan objek warna yang ditempelkan di jari pengguna.
2. Bagaimana sistem dapat menentukan poin 1 dan poin 2 dari objek warna yang bergerak.
3. Bagaimana sistem dapat membentuk garis *rectangular* berdasarkan poin 1 dan poin 2 dari pergerakan objek warna.
4. Bagaimana sistem dapat mengambil bentuk citra berdasarkan pembentukan garis *rectangular*.

Tujuan Penelitian

Tujuan dari kegiatan penelitian ini adalah mendeteksi adanya warna yang bergerak dalam penentuan letak dengan menggunakan metode pelacakan yakni metode segmentasi warna dan *center of gravity (COG)* sehingga dapat menyeleksi suatu citra tertentu berdasarkan daerah yang diinginkan (*Region Of Interest*).

Batasan Masalah

Batasan masalah adalah sebagai berikut :

1. Aplikasi yang dibuat adalah aplikasi berbasis *desktop*.
2. Menggunakan webcam untuk menangkap pergerakan objek warna.
3. Warna yang digunakan pada objek yang di *tracking* adalah warna yang berbeda dengan latar warna *background* dan objek yang diseleksi serta di *capture*.
4. Garis yang terbentuk dari proses *tracking* warna berdasarkan koordinat awal dan koordinat akhir adalah garis dengan bentuk persegi (*rectangular*).
5. Resolusi dari keluaran bentuk citra yang dihasilkan bergantung pada kamera webcam yang digunakan.
6. Metode yang digunakan metode pelacakan yakni mengkombinasikan metode segmentasi warna dan *center of gravity (COG)*.
7. Proses segmentasi warna menggunakan framework *aforge.Net*

Framework Aforge.Net ^[6]

AForge.NET merupakan framework C# terbuka yang dirancang untuk pengembang dan peneliti di bidang Computer Vision dan *Artificial Intelligence* (Kecerdasan Buatan) - Pengolahan Gambar, Jaringan Saraf Tiruan, Algoritma Genetika, Logika Fuzzy, Pembelajaran Mesin, Robotika, dll. Kerangka ini terdiri oleh set perpustakaan dan contoh aplikasi, yang menunjukkan karakteristiknya:

- AForge.Imaging - library dengan rutinitas pengolahan citra dan filter;
- AForge.Vision - library *computer vision*;
- AForge.Video - set library untuk pemrosesan video;
- AForge.Neuro - library perhitungan jaringan saraf tiruan;
- AForge.Genetic - library pemrograman evolusi;
- AForge.Fuzzy - library perhitungan *fuzzy*;
- AForge.Robotics - library yang memberikan dukungan pada beberapa robotika kit;
- AForge.MachineLearning - library pembelajaran mesin; dan lain-lain.

Euclidean Color Filtering

Euclidean Color Filtering adalah metode yang berguna untuk menemukan sebuah warna yang terdapat pada sebuah gambar. Filter ini memfilter piksel-piksel pada gambar yang berada di dalam/di luar dari lingkup RGB (Red Green Blue) dengan pusat dan radius tertentu. Filter tersebut membiarkan piksel-piksel dengan warna yang berada di dalam/di luar dari lingkup yang telah ditentukan dan mengisi sisanya dengan warna tertentu.

Grayscale

GrayScale adalah suatu citra dimana nilai dari setiap pixel merupakan sampel tunggal. Citra yang ditampilkan adalah citra keabuan dimana intensitasnya berada pada interval 0 – 255, warna hitam

(0) pada bagian yang intensitasnya

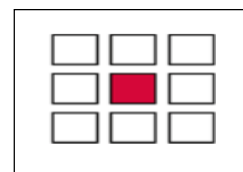
terlemah dan warna putih(255) pada intensitas terkuat. Proses konversi dari citra berwarna menjadi Grayscale menggunakan koefisien dari ITU Recommendation BT.709.

$$\text{Grayscale} = 0.2125 * \text{red} + 0.7154 * \text{green} + 0.0721 * \text{blue}.$$

Persamaan Konversi Grayscale(1)

Blob ^[4]

Blob merupakan sekumpulan piksel-piksel yang memiliki hubungan tetangga. Proses perhitungan *blob* dapat dilakukan dengan melakukan analisis piksel yang bertetangga. Piksel bertetangga pada sebuah piksel ditentukan sebagai piksel yang berjarak 1 dari piksel asal. Proses perhitungan *blob* akan memanfaatkan relasi piksel *8-neighbors*. Gambar 1 merupakan gambaran sederhana dari relasi *8-neighbors*.



Gambar 1. Relasi 8-neighbors

P (x-1, y-1)	P(x, y-1)	P (x+1, y-1)
P (x-1, y)	P (x,y)	P(x+1, y)
P (x-1, y+1)	P(x, y+1)	P (x+1, y+1)

Relasi 8-neighbors

Proses pemetaan objek akan menelusuri tiap piksel pada setiap baris yang ada dan memberikan label

pada piksel yang memiliki nilai warna selain hitam (RGB = 0 0 0). Setiap piksel yang memiliki relasi hubungan *8-neighbors* akan diberikan label yang sama.

Pusat Massa Obyek (*Center Of Gravity*)^[9]

Metode Center of Gravity adalah metode yang dipergunakan untuk menentukan titik keseimbangan dari grafik yang merupakan hasil dari proses pengolahan citra. Persamaan 3

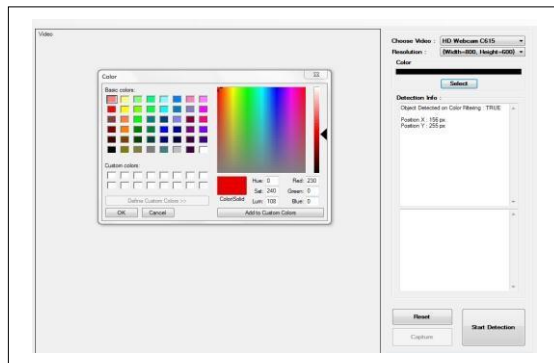
Gambar 3. FlowChart Sistem

Berdasarkan sistem kerja aplikasi seperti terlihat pada gambar 3, terdapat beberapa tahapan dalam implementasi segmentasi citra untuk deteksi objek warna pada pengambilan citra *rectangle*, berikut adalah tahapan-tahapan dari gambar 3 :

1. Mulai

Merupakan *state* awal memulai aplikasi.

2. Pemilihan Warna



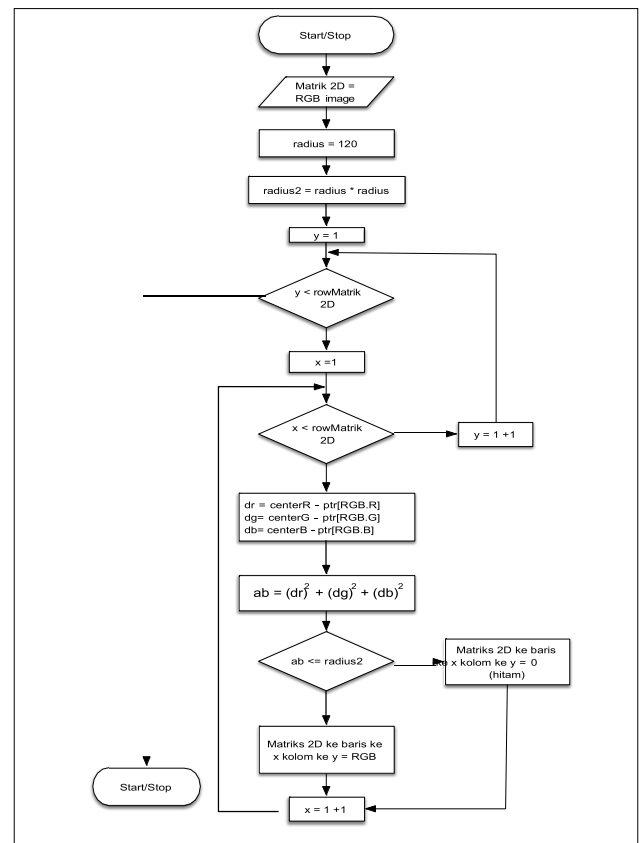
Gambar 4. *Pallate* Warna untuk pemilihan warna

Pada tahap pemilihan warna dilakukan sebagai masukan kepada sistem

untuk mendeteksi warna yang akan dideteksi dan dijejaki. Proses pemilihan warna dapat dilihat pada gambar 4.

3. Pengaktifan kamera webcam

Pada tahap proses pengaktifan perangkat kamera webcam dilakukan sebagai media untuk mendeteksi objek. Pengaktifan kamera webcam dilakukan dengan menekan tombol *start detection* pada aplikasi. Pada gambar 5 merupakan kondisi ketika media webcam aktif.



Gambar 5. Kamera Webcam Aktif

4. Input Objek Warna

Pada tahap ini dilakukan input objek warna dari warna yang sebelumnya dipilih pada tahap 1. Gambar 6 adalah proses input objek warna.

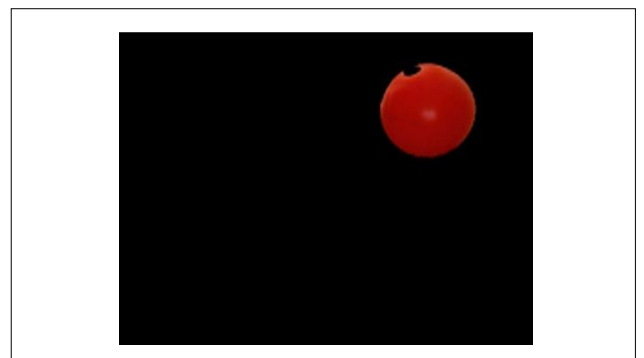


Gambar 6. Input Objek Warna

5. Segmentasi objek warna terhadap ruang warna RGB dengan Euclidean Color Filtering

Pada tahap ini dilakukan proses *filtering* warna menggunakan *euclidean color filtering* berdasarkan pemilihan warna pada tahap 1. Proses *filtering* menggunakan *euclidean color filtering* ini adalah memisahkan objek warna dari background dimana background akan dijadikan warna hitam seperti terlihat pada gambar 8. Proses euclidean dapat dilihat pada gambar 7.

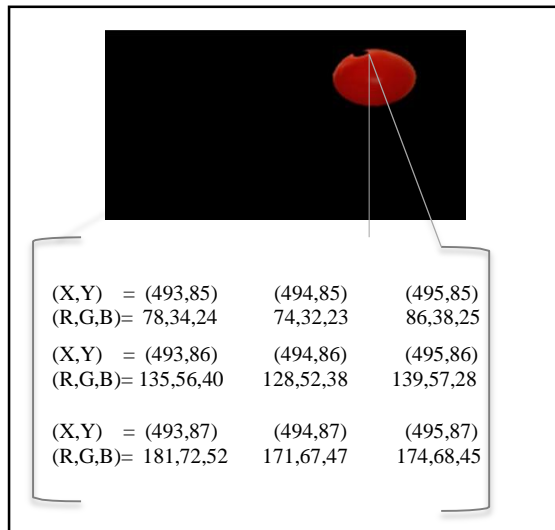
Gambar 7. Flowchart Proses Euclidean Color Filtering



Gambar 8. Input citra setelah dilakukan euclidean color filtering

6. Konversi Warna RGB ke *Grayscale*

Setelah dilakukan pemisahan objek warna dengan menggunakan *euclidean color filtering*, state selanjutnya adalah mengubah nilai warna citra rgb ke dalam ruang warna *grayscale* (keabuan). Proses mengubah nilai warna citra rgb ke dalam ruang warna *grayscale* menggunakan persamaan 1. ilustrasi perubahan nilai warna citra rgb dapat dilihat pada gambar

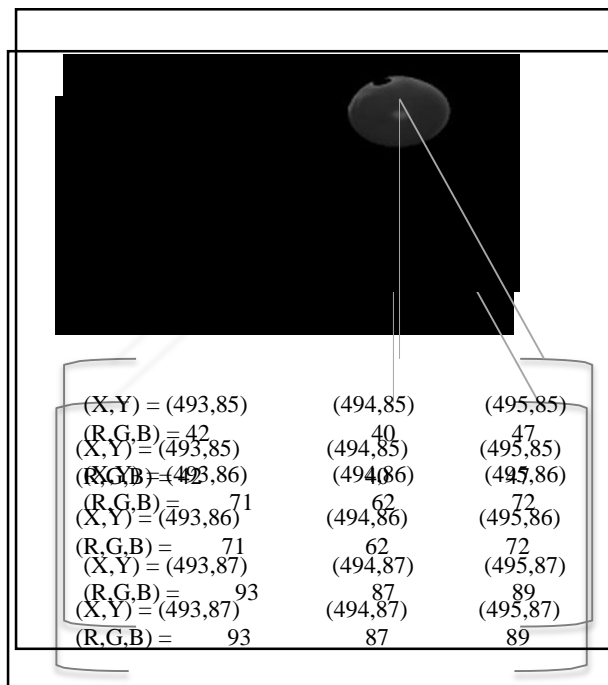


c. Koordinat (495,87)

Grayscale = $(0.2125 \times R) + (0.7154 \times G) + (0.0721 \times B)$

$$\begin{aligned}
 &= (0.2125 \times 174) + (0.7154 \times 68) + (0.0721 \times 45) \\
 &= 36,975 + 48,6472 + 3,2445 \\
 &= 88,8667 = 89
 \end{aligned}$$

Gambar 9. Ilustrasi RGB pada piksel citra sebelum dikonversi



Gambar 10. Ilustrasi RGB pada piksel citra setelah dikonversi

Berikut contoh proses perhitungan nilai rata-rata citra pada titik koordinat (493,85), (494,86), (495,87) dengan persamaan 1.

a. Koordinat (493,85)

$$\begin{aligned}\text{Grayscale} &= (0.2125 \times R) + (0.7154 \times G) + (0.0721 \times B) \\ &= (0.2125 \times 78) + (0.7154 \times 34) + (0.0721 \times 24) \\ &= 16.575 + 24.3236 + 1.7304 \\ &= 42.629 = 42\end{aligned}$$

b. Koordinat (494,86)

$$\begin{aligned}\text{Grayscale} &= (0.2125 \times R) + (0.7154 \times G) + (0.0721 \times B) \\ &= (0.2125 \times 128) + (0.7154 \times 52) + (0.0721 \times 38) \\ &= 27.2 + 37.2008 + 2.7398 \\ &= 62.1406 = 62\end{aligned}$$

7. Pemetaan dan Perhitungan Blob dengan BlobCounter

Proses perhitungan *blob* akan memanfaatkan relasi piksel *8-neighbors*. Menurut pustaka AForge.Net langkah- langkah perhitungan blob adalah sebagai berikut.

- Proses pemetaan objek akan menelusuri tiap piksel pada setiap baris yang ada dan memberikan label pada piksel yang memiliki nilai warna selain hitam (RGB = 0 0 0). Setiap piksel yang memiliki relasi hubungan *8-neighbors* akan diberikan label yang sama.
- Proses pengumpulan informasi *blob*, akan mengumpulkan dan mengolah informasi tiap piksel yang bertetangga berdasarkan letak dan label yang dihasilkan oleh proses pemetaan objek. Letak dan label piksel yang bertetangga tersebut digunakan untuk membentuk suatu *blob* dan informasi pendukungnya seperti luas area, tingkat kepenuhan, titik pusat dan area kotak *blob*.

Gambar 11 adalah hasil dari pemetaan dan perhitungan *blob*.

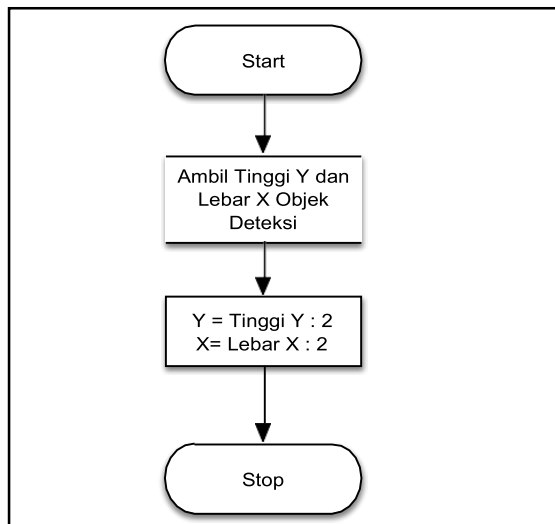


Gambar 11. hasil deteksi blob pada ruang warna RGB

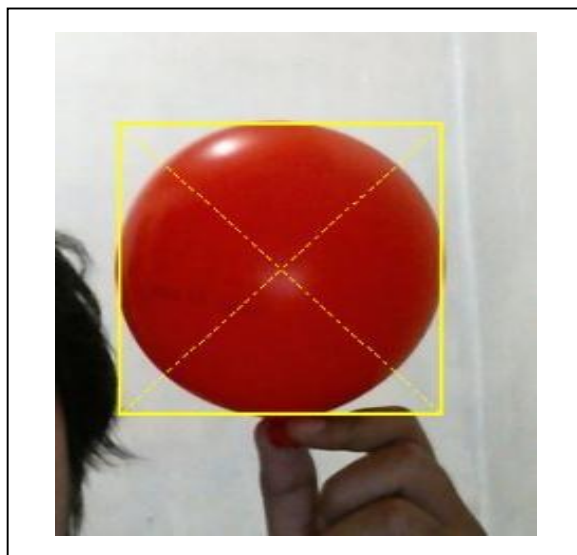
8. Penentuan Titik Pusat (*Center of Gravity*)

Setelah objek warna terdeteksi melalui proses segmentasi warna, grayscale dan blob, maka proses selanjutnya adalah penentuan titik pusat (*center of gravity*) dari objek yang telah dideteksi. Penentuan titik pusat ini nantinya dapat digunakan dalam pembentukan garis *rectangle* dalam proses pengambilan bentuk citra. Proses penentuan titik pusat ini menggunakan persamaan 3 seperti terlihat pada gambar

12. Realisasi dari perhitungan persamaan 3 terlihat pada gambar 13.



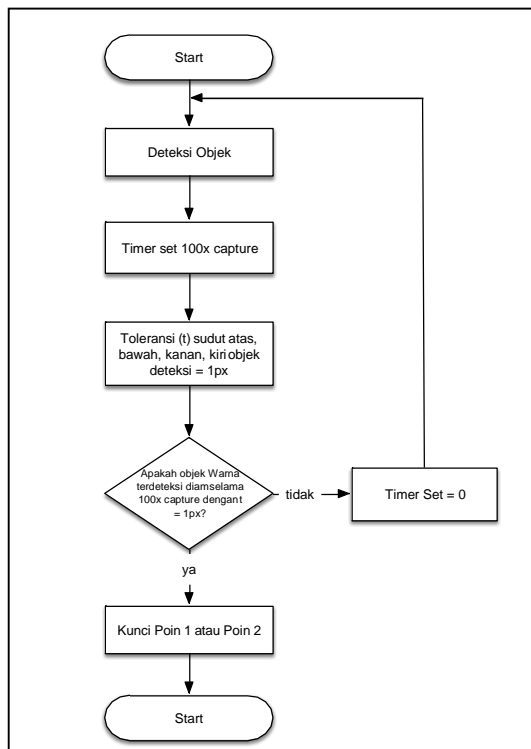
Gambar 12. Flowchart penentuan titik pusat



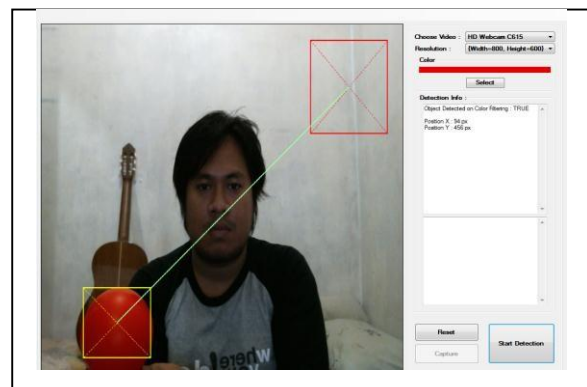
Gambar 13. Titik pusat dari deteksi objek warna

9. Penguncian Poin 1 dan Poin 2

Dalam proses pengambilan bentuk citra *rectangle* dibutuhkan penentuan poin 1 dan poin 2 oleh user. Setelah objek terdeteksi dan telah ditentukan titik pusat, proses selanjutnya adalah penguncian poin 1 dan poin 2. Poin 1 dan poin 2 dapat terkunci jika objek yang terdeteksi diam selama 100x capture dalam toleransi 1px atas, 1px bawah, 1px kanan dan 1px kiri. Gambar 14 adalah *flowchart* proses penguncian poin 1 dan poin 2. Gambar 15 adalah realisasi dari *flowchart* penguncian poin 1 dan poin 2.



Gambar 14. Flowchart Penguncian Poin 1 dan poin 2



Gambar 15. Realisasi penguncian poin 1 dan poin 2

10. Pembentukan *rectangle*

Setelah penguncian poin 1 dan poin 2, maka sistem secara otomatis akan membentuk garis *rectangle*. Proses pembentukan *rectangle* seperti terlihat pada Gambar 16. Realisasi proses pembentukan *rectangle* seperti terlihat pada Gambar 17.



```

private void lock_rectangle(Graphics g, Bitmap mImage, Bitmap realImage)
{
    using (Pen pen = new Pen(Color.FromArgb(160, 255, 160), 2))
    {
        //g.DrawRectangle(pen, objectRect);
        if (endPoint == true && posXA != 0 && posXB != 0)
        {
            int posXAA = 0;
            int posYAA = 0;
            int posXBB = 0;
            int posYBB = 0;

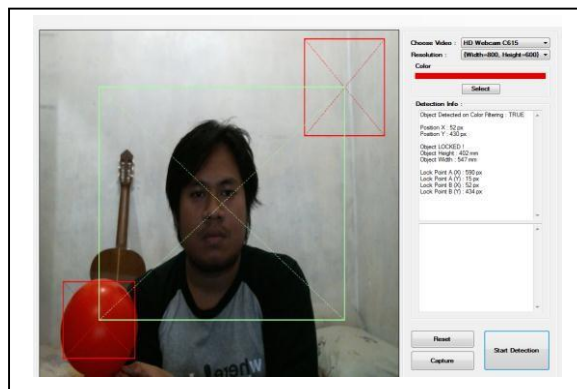
            if (posXB - posXA < 0)
            {
                posXAA = posXB + (widthB/2);
                posXBB = posXA + (widthA/2);
            }
            else
            {
                posXAA = posXA + (widthA/2);
                posXBB = posXB + (widthB/2);
            }

            if (posYB - posYA < 0)
            {
                posYAA = posYB + (heightB/2);
                posYBB = posYA + (heightA/2);
            }
            else
            {
                posYAA = posYA + (heightA/2);
                posYBB = posYB + (heightB/2);
            }

            pointRect = Rectangle.FromLTRB(posXAA, posYAA, posXBB, posYBB);
            g.DrawRectangle(pen, pointRect);
        }
    }
}

```

Gambar 16. Proses Pembentukan
Rectangle



Gambar 17. Realisasi Pembentukan
Rectangle

11. Pengambilan Citra *Rectangle*

Setelah pembentukan garis *rectangle* maka untuk pengambilan bentuk citra *rectangle* dilakukan penekanan pada tombol *capture* yang terdapat pada aplikasi. Hasil proses *capture* seperti terlihat pada Gambar 18. Format gambar dari hasil pengambilan citra berdasarkan pembentukan garis *rectangle* adalah berformat .jpg.

Gambar 18. Hasil Pengambilan gambar

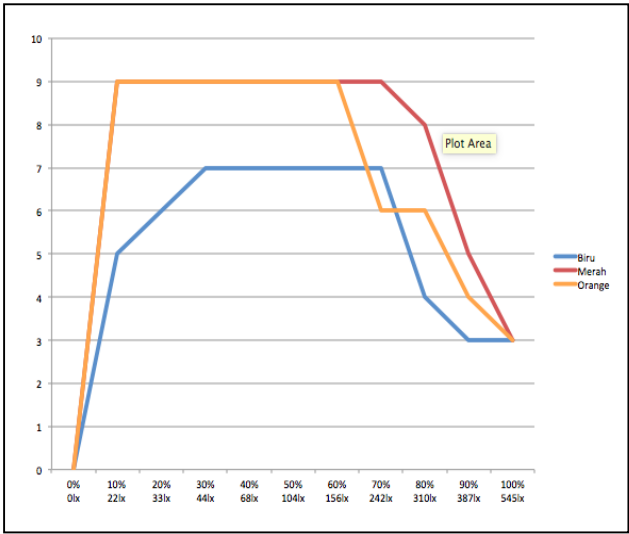
Pengujian Deteksi Terhadap Cahaya dan Jarak

Pengujian pada aplikasi deteksi objek bergerak untuk pengambilan bentuk citra ini meliputi pengujian deteksi objek warna terhadap cahaya, pengujian deteksi objek warna terhadap jarak, dan pengujian blackbox fitur sistem. Hasil pengujian deteksi objek warna terhadap cahaya dan jarak terlihat pada tabel 1.

Tabel 1 tingkat keberhasilan deteksi objek warna terhadap cahaya

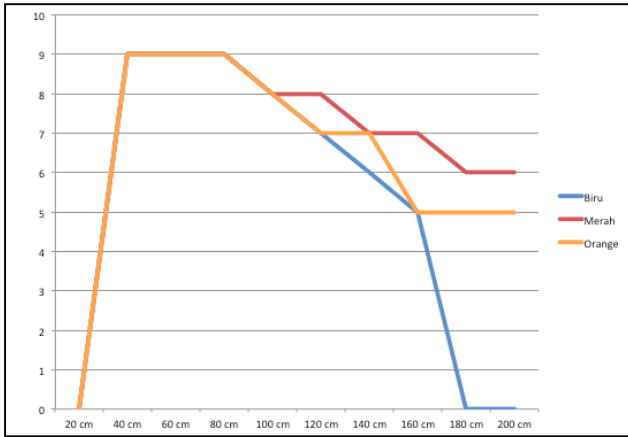
Pengujian Deteksi Objek Warna Merah	Jarak										Nilai Cahaya
	20	40	60	80	100	120	140	160	180	200	
	X	X	X	X	X	X	X	X	X	X	0% = 0 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	10% = 22 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	20% = 33 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	30% = 44 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	40% = 68 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	50% = 104 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	60% = 156 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	70% = 242 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	80% = 310 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	90% = 387 lx
	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	100% = 545 lx

Pada gambar 19 merupakan grafik pengujian deteksi objek warna terhadap cahaya dari 3 objek warna yang diuji yaitu merah, biru dan orange



Gambar 19. Grafik deteksi warna terhadap intensitas cahaya

Pada gambar 20 merupakan grafik pengujian deteksi objek warna terhadap jarak dari 3 objek warna yang diuji yaitu merah, biru dan orange



Gambar 20. Grafik deteksi warna terhadap jarak


Pengujian Blackbox Fitur Sistem

Pengujian fitur sistem ini dilakukan dengan menggunakan metode *blackbox* agar dapat diketahui fitur sistem dapat berfungsi dengan baik seperti terlihat pada tabel 2. Pengujian fitur sistem yang dilakukan

adalah:

1. Pemilihan warna
2. Deteksi objek warna
3. Penguncian poin
4. Pembentukan *Rectangle*
5. Pengambilan Gambar

Tabel 2 Pengujian *Blackbox* fitur sistem

NO	Hasil Pengujian	Keterangan
1		Media pilihan warna muncul/berhasil
2		Deteksi objek warna berhasil
3		Penguncian poin 1 dan poin 2 berhasil
		Pembentukan garis <i>rectangle</i> berhasil
5		Pengambilan gambar bentuk <i>rectangle</i> berhasil



Kesimpulan

Berdasarkan penelitian yang telah dilakukan, didapat kesimpulan bahwa :

1. Segmentasi Citra dan pengambilan titik pusat untuk deteksi objek bergerak dapat diimplementasikan pada aplikasi deteksi objek warna untuk pengambilan citra bentuk *rectangle*.
2. Dari hasil pengujian yang telah dilakukan, dapat ditarik kesimpulan bahwa dari proses segmentasi citra untuk deteksi objek warna diketahui jarak terbaik

adalah antara 40cm – 80cm. Sedangkan untuk deteksi warna terhadap intensitas cahaya, nilai lumiance terbaik adalah antara 22lx

– 242lx.

3. Dari hasil pengujian pada aplikasi deteksi objek warna untuk pengambilan bentuk citra *rectangle*, didapatkan bahwa warna pada objek warna yang dideteksi tidak boleh sama atau terdapat pada latar *background* atau objek yang akan di *capture*.
4. Dari percobaan 3 warna yang dideteksi, warna merah adalah warna yang dapat terdeteksi dengan baik.

Daftar Pustaka

- [1] Rahmadi Kurnia. 2009. "*Penjejak Target Benda Pada Gerakan Liner Berdasarkan Warna*". Padang : Universitas Andalas Padang.
- [2] Wawan Kurniawan. 2011. "*Pengenalan Bahasa Isyarat Dengan Menggunakan Metode Segmentasi Warna Kulit dan Center Of Gravity*". Jambi : Universitas Jambi.
- [3] Shanti Anggriani Tambunan. 2010. "*Pengembangan Kerangka Prototype Detektor Api Pada Citra Digital Dengan Menerapkan Metode Segmentasi SRM (Statistical Region Merging)*". Bandung : Institut Teknologi Nasional Bandung.
- [4] Benedictus Yoga Budi Putranto. 2010. "*Segmentasi Warna Citra Dengan Deteksi HSV Untuk Mendeteksi Objek*". Yogyakarta : Universitas Kristen Duta Wacana.
- [5] Mohamad Ihsan Nurdin. 2013. "*Perbandingan Aplikasi Pengenalan Gender Berdasarkan Citra Wajah Dengan Implementasi Library Shore Terhadap Aplikasi EyeFace Recognition*". Bandung : Institut Teknologi Nasional Bandung
- [6] Framwork aforge.Net. <http://www.aforogenet.com>. Diakses online pada tanggal 1 juli 2014.

JURNAL 3

Klasifikasi Warna Menggunakan Pengolahan Model Warna HSV

R. D. Kusumanto^{1*}, Alan Novi Tompunu, dan Wahyu Setyo Pambudi²

1. Jurusan Teknik Komputer, Politeknik Negeri Sriwijaya, Palembang 30139, Indonesia

2. Jurusan Teknik Elektro, Universitas Internasional Batam, Batam 29442, Indonesia

*E-mail: manto_6611@yahoo.co.id

Abstrak

Pengolahan citra digital (*digital image processing*) adalah sebuah disiplin ilmu yang mempelajari tentang teknik-teknik mengolah citra. Citra yang dimaksud pada penelitian ini adalah gambar statis yang berasal dari sensor *vision* (webcam). Secara matematis, citra merupakan fungsi kontinu dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi $f(x,y)$ yang terdiri dari M kolom dan N baris. Pada pengolahan warna gambar, ada bermacam-macam model salah satunya adalah model *hue, saturation, value* (HSV). Dengan menggunakan model ini, sebuah obyek dengan warna tertentu dapat dideteksi dan mengurangi pengaruh intensitas cahaya dari luar. Pengujian yang dilakukan menggunakan 6 jenis warna, yaitu coklat, kuning, hijau, biru, hitam, dan putih. Berdasarkan pengujian, warna coklat memiliki *error* paling kecil yaitu 10% setiap 10 pengukuran atau 1 kali kesalahan.

1. Pendahuluan

Pengolahan citra digital menggunakan teknologi *computer vision* saat ini banyak digunakan sebagai obyek penelitian. Bagian dari pengolahan citra adalah dengan menggunakan pengolahan berdasarkan warna. Analisis warna dalam pengenalan citra digital ini ada beberapa model diantaranya, model RGB, CMY, HSI, HSV dan *normalized RGB* [1-2]. Salah satu bentuk aplikasi model HSV adalah sebagai pengenalan wajah [3]. Menggunakan model ini sebagai pengenalan wajah memiliki keuntungan yaitu sederhana dalam pemrograman, prosesnya cepat sehingga sangat cocok untuk aplikasi *real time*. Berkembangnya penerapan sensor visual dan disiplin ilmu *image processing* (pengolahan citra) telah menginspirasi pihak yang berwenang dalam peningkatan pendidikan tinggi yang dalam hal ini adalah DIKTI, untuk memasukkan unsur tersebut. Hal ini seperti yang terbukti dengan adanya ajang kompetisi tentang robot *humanoid* pemain bola yang dapat mengenali bola dan gawang yang memiliki warna berbeda [4].

Berdasarkan hal tersebut maka penelitian awal ini akan diarahkan untuk dapat mengenali warna dengan model HSV yang kedepannya warna-warna ini akan

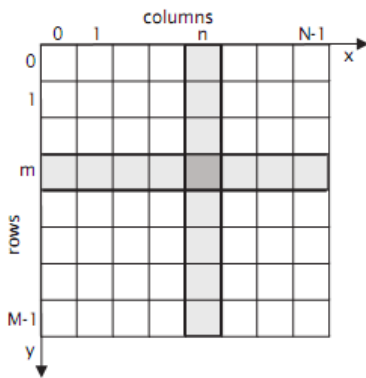
merepresentasikan obyek tertentu. Harapannya bahwa dengan penelitian ini akan mampu membuat dasar konsep pengenalan obyek berdasarkan warna yang akan digunakan untuk aplikasi robot nantinya.

2 Metode Penelitian

Pengolahan citra digital (*digital image processing*) adalah sebuah disiplin ilmu yang mempelajari tentang teknik-teknik mengolah citra. Citra yang dimaksud disini adalah gambar diam (foto) maupun gambar bergerak (yang berasal dari *webcam*). Sedangkan digital disini mempunyai maksud bahwa pengolahan citra/ gambar dilakukan secara digital menggunakan komputer [1].

Secara matematis, citra merupakan fungsi kontinu (*continue*) dengan intensitas cahaya pada bidang dua dimensi. Agar dapat diolah dengan komputer digital, maka suatu citra harus dipresentasikan secara numerik dengan nilai-nilai diskrit. Representasi dari fungsi kontinyu menjadi nilai-nilai diskrit disebut digitalisasi citra.

Sebuah citra digital dapat diwakili oleh sebuah matriks dua dimensi $f(x,y)$ yang terdiri dari M kolom dan N baris, dimana perpotongan antara kolom dan baris disebut piksel (*pixel = picture element*) atau elemen terkecil dari sebuah citra.



Gambar 1. Representasi Citra Digital dalam 2 Dimensi [5]

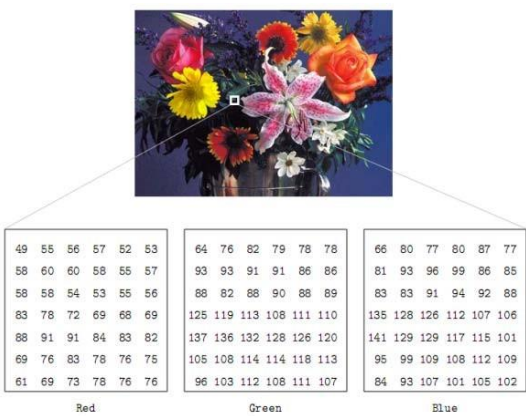
Pengolahan Warna. Pada pengolahan warna gambar, ada bermacam-macam model warna. Model RGB (*red green blue*) merupakan model yang banyak digunakan, salah satunya adalah monitor. Pada model ini untuk merepresentasikan gambar menggunakan 3 buah komponen warna tersebut. Selain model RGB terdapat juga model HSV dimana model ini terdapat 3 komponen yaitu, *hue*, *saturation*, dan *value*. *Hue* adalah suatu ukuran panjang gelombang yang terdapat pada warna dominan yang diterima oleh penglihatan sedangkan *Saturation* adalah ukuran banyaknya cahaya putih yang bercampur pada *hue*.

Jenis Citra Digital. Pada aplikasi pengolahan citra digital pada umumnya, citra digital dapat dibagi menjadi 3, *color image*, *black and white image*, dan *binary image*: a) *color image* atau RGB. Pada *color image* ini masing-masing piksel memiliki warna tertentu, warna tersebut adalah merah (*red*), hijau (*green*) dan biru (*blue*). Jika masing-masing warna memiliki *range* 0- 255, maka totalnya adalah $255^3 = 16.581.375$ (16 K) variasi warna berbeda pada gambar, di mana variasi warna ini cukup untuk gambar apapun. Karena jumlah bit yang diperlukan untuk setiap piksel, gambar tersebut juga disebut gambar bit warna. *Color image* ini terdiri dari tiga matriks yang mewakili nilai-nilai merah, hijau, dan biru untuk setiap pikselnya; b) *black and white*. Citra digital *black and white* (*grayscale*) setiap

pikselnya mempunyai warna gradasi mulai dari putih sampai hitam. Rentang tersebut berarti bahwa setiap piksel dapat diwakili oleh 8 bit, atau 1 *byte*. Rentang warna pada *black and white* sangat cocok digunakan untuk pengolahan *file* gambar. Salah satu bentuk fungsinya digunakan dalam kedokteran (*X-ray*).

Black and white sebenarnya merupakan hasil rata-rata dari *color image*, dengan demikian maka persamaannya dapat dituliskan sebagai berikut:

1. Hasil dan Pembahasan



Dengan $I_{BW}(x,y)$ = nilai piksel *gray* titik (x,y) , $I_{Bin}(x,y)$ = nilai piksel *binary* titik (x,y) , sedangkan T adalah nilai *threshold*.

Pengolahan Citra menggunakan EmguCV. EmguCV adalah cross platform yang terdapat dalam .NET untuk *library* pengolahan citra pada Intel OpenCV. EmguCV ini mengikuti fungsi yang terdapat pada OpenCV yang diambil dari .NET oleh sebab itu *compatible* dengan bahasa pemrograman C#, VB, VC++, IronPython dan sebagainya. Program ini bersifat *opensource* sehingga sangat cocok apabila digunakan untuk penelitian, salah satunya adalah untuk *computer vision*.

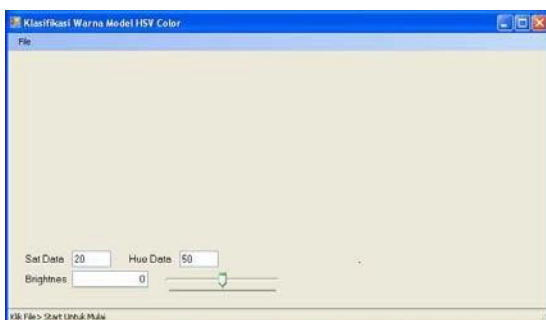
Pada pengujian untuk menentukan klasifikasi warna dengan menggunakan model HSV color ini menggunakan program Visual Studio 2008 yang telah dilengkapi dengan program pendukung EmguCV [3].

Sebelum dilakukan klasifikasi warna ini langkah pertama yang dilakukan adalah merubah *color image* menjadi *HSV image*, yang dalam hal ini menggunakan fungsi yang terdapat pada EmguCV. Kemudian data *Hue* dan *Sat* diambil untuk menentukan area pendeteksian matrix *image*, dalam hal ini menggunakan area 10x10.

Berdasarkan program di atas, di mana untuk i = tinggi *image*, j = panjang *image*, *HueData* = nilai *Hue*, *SatData* = nilai *Saturation*, $sum1$ = jumlah total data *Hue* untuk ukuran 10x10 dan $sum2$ = jumlah total data *Sat* untuk ukuran 10x10. Setelah data *Hue* dan *Sat* 10x10 didapatkan kemudian diambil rata-ratanya.

Berdasarkan nilai yang telah didapatkan tersebut maka dapat ditentukan nilai *threshold*, sehingga warna-warna ini dapat diinterpretasikan

Gambar 5. Program Pengolahan Citra



Gambar 5. Program Pengolahan Citra

```

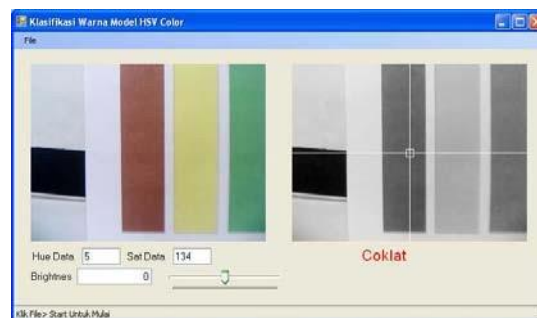
For i = 115 To 124
  For j = 155 To 164
    Dim HueData As Integer = ImgHSV.Data(i, j, 0) Dim
    SatData As Integer = ImgHSV.Data(i, j, 1) sum1 =
    sum1 + HueData
    sum2 = sum2 + SatData
  Next
Next

```

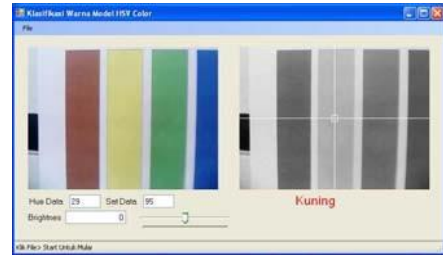


Gambar 6. Proses Pengambilan Data Warna Tabel 1. Hasil Pengambilan Data Warna

Warna	Hue	Sat
Coklat	4-24	126-130
Kuning	30-32	78-83
Hijau	60-66	41-44
Biru	166-175	105-107
Hitam	28-54	65-145
Putih	28-16	4-1



Gambar 7. Program pada saat Mendeteksi Warna Coklat



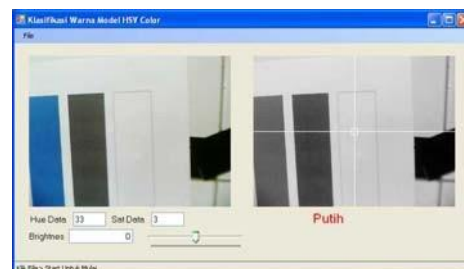
Gambar 8. Program pada saat Mendeteksi Warna Kuning



Gambar 9. Program pada saat Mendeteksi Warna Hijau



Gambar 10. Program pada saat Mendeteksi Warna Biru



Gambar 11. Program pada saat Mendeteksi Warna Putih Tabel 2. Hasil Pengujian

Warna	Pengujian	Error
Coklat	10	1
Kuning	10	3
Hijau	10	2
Biru	10	3
Hitam	10	4
Putih	10	1

Setelah dilakukan pengujian selama 10 kali untuk masing-masing warna, maka didapatkan hasil seperti pada Tabel 2.

3. Simpulan

Berdasarkan pengujian yang telah dilakukan model HSV *color* ini dapat digunakan sebagai untuk menentukan jenis warna secara *real time*. Pada saat pengujian untuk warna coklat, memiliki *error* pengukuran sebesar 10% dari 10 kali pengukuran, sehingga dapat dikatakan bahwa warna coklat mempunyai respon paling baik dibandingkan dengan warna lain pada saat diolah menggunakan model HSV *color*.

Daftar Acuan

- [1] T. Sutoyo, E. Mulyanto, V. Suhartono, O.D. Nurhayati, Wijanarto, Teori Pengolahan Citra Digital, Penerbit Andi, Yogyakarta, 2009, p.256.
- [2] F. Guo, Q. Cao, Proceedings of the 5th World Congress on Intelligent Control and Automation, Hangzhou, P.R. China, 2004.
- [3] A. Iyad, H. Mahmoud, Human Face Detection System Using HSV, Recent Researches in Circuit, Systems, Electronics, Control & Signal Processing, 2009, p.16.
- [4] Anon., Panduan Kontes Robot Cerdas Indonesia (KRCI 2011), Direktorat Penelitian dan Pengabdian kepada Masyarakat, Direktorat Jenderal Pendidikan Tinggi, Kementerian Pendidikan Nasional, Jakarta, 2011.
- [5] J. Bernd, H. Horst, Computer Vision and Applications, Academic Press, San Diego, California, 2000, p.679.
- [6] M. Alasdair, An Introduction to Digital Image Processing with Matlab, Notes for SCM2511 Image Processing 1, School of Computer Science and Mathematics Victoria University of Technology

