PEMBANGUNAN APLIKASI DETEKSI DAN *TRACKING* WARNA *VIRTUAL WRITING* MENGGUNAKAN ALGORITMA *COLOR FILTERING*

M Azhan Mizani[1], Nelly Indriani W[2]

[1,2] Teknik Informatika-Universitas Komputer Indonesia
Jalan Diaptiukur No.112-116 Bandung. 40132
E-mail : m.azhanmizani21@gmail.com[1], indiwidi@gmail.com[2]

ABSTRAK

Pembangunan aplikasi virtual *writing* menggunakan algoritma *color filtering* adalah sebuah aplikasi untuk menuliskan huruf atau kata secara virtual menggunakan objek yang di-*tracking* sebagai *pointer*. Aplikasi ini dibangun untuk memperbaiki kekurangan pada penelitian sebelumnya. Pada penelitian sebelumnya hanya mendeteksi warna merah saja dan mengalami gangguan pada hasil jika ada objek lain yang berwana dan berukuran sama dengan objek (*marker*).

Color filtering merupakan teknik pengolahan citra berdasarkan warna dan *camshift* merupakan metode untuk melacak titik tengah dari objek. Dalam penelitian ini, pembangun aplikasi *virtual writing* memanfaatkan kamera (*webcam*) dan warna objek sebagai *marker*. Aplikasi berjalan dengan memilih area objek yang akan dijadikan sebagai *search window* dan menghitung histogram warna pada *search window* tersebut untuk dijadikan warna acuan dengan metode color filtering, kemudian menghitung titik tengah pada *search window* dengan metode *camshift*. Aplikasi dibangun dengan *library* tambahan yaitu OpenCV. Tujuan dari aplikasi ini bisa digunakan untuk menulis secara virtual tanpa menggunakan *mouse*.

Hasil dari pengujian yang dilakukan dengan objek ukuran 4.5x2cm mendapatkan hasil yang baik pada pencahayaan baik 9-15 watt dari jarak 10 cm - 100 cm. Kesimpulannya, ukuran objek akan mempengaruhi hasil pada jarak tertentu, dan juga jarak mempengaruhi kemampuan deteksi objek berdasarkan ukurannya. Cahaya yang kurang baik akan mempengaruhi pendeteksian objek.

Kata kunci : Deteksi, *Tracking*, Objek, *Color Filtering, CamShift.*

1. PENDAHULUAN

Pengolahan citra digital adalah sebuah proses yang dilakukan pada sebuah gambar dengan bantuan komputer untuk memperbaiki kualitas citra atau gambar menjadi lebih bagus dan mendapatkan nilai warna yang terdapat di dalamnya[1]. Kaitan antara pengolahan citra digital dan *computer vision* adalah pengolahan citra bagian langkah-langkah dari *computer vision* yang disebut dengan *preprocessing*[2].

Computer vision adalah transformasi atau perubahan dari data-data yang dapat berupa gambar diam ataupun *video* kamera menjadi bentuk lain atau satu representasi baru. *Computer vision* dapat didefinisikan dengan pengertian pengolahan citra yang dikaitkan dengan akuisisi citra, pemrosesan, klasifikasi, dan pencakupan keseluruhan, pengambilan keputusan yang diikuti pengidentifikasian citra[2].

Salah satu perkembangan dari penggunaan teknik *computervision* adalah *tracking* objek. *Tracking* objek bertujuan untuk mendeteksi dan mengikuti posisi dari suatu objek bergerak yang diinginkan. *Tracking* objek banyak dibutuhkan oleh berbagai macam aplikasi *visionbased* seperti *human computer interface*, kompresi/komunikasi *video* dan sistem keamanan[4]. *Tracking* objek harus mampu mendeteksi objek yang bergerak, memfilter *noise*, dan gerakan-gerakan lain yang tidak diperlukan.

Penelitian terhadap *object tracking* telah dilakukan oleh Rio[5]. Pada hasil penelitian dapat disimpulkan, aplikasi telah dapat mendeteksi warna dari sebuah objek dan menandai posisi piksel yang dilewati, aplikasi yang yang dikembangkan masih memiliki kekurangan yaitu pada hasil *tracking* masih terputus-putus jika pergerakan objek cepat.

Color filtering adalah suatu teknik pengolahan citra berdasarkan warna yang spesifik. Cara kerja pada *color filtering* adalah dengan membandingkan komponen warna setiap piksel citra dengan warna yang dipilih. *Color filtering* ini digunakan pada tahapan awal untuk mendeteksi objek berbasis warna. Setelah menemukan warna yang ditentukan, peneliatian dapat dilanjutkan ke proses *tracking* objek. kelebihan menggunakan metode *color filtering* yaitu salah satu metode yang paling sederhana melakukan penyaringan warna dalam ruang warna RGB (*Red, Green, Blue*) yang memberikan performa tercepat pada gambar RGB[4].

Metode yang digunakan untuk mencari *centroid* dari objek yang telah difilter berdasarkan warna adalah *CamShift* (*Continuously Adaptive Mean Shift*) *Algorithm*. Algoritma CamShift awalnya dikembangkan untuk aplikasi *tracking* kepala dan wajah berdasarkan histogram warna dari target objek.

CamShift merupakan pengembangan dari algoritma *mean shift* yang mencari probabilitas warna terbesar yang dihasilkan oleh histogram warna[6]. Algoritma ini digunakan, karena dapat mengubah ukuran jendela pencarian (*search window*) jika terjadi rotasi, besar dan kecil objek yang menyebabkan nilai distribusi piksel pada objek berubah[13]. Dan juga menghitung seluruh nilai piksel pada area *capture* untuk menunjukkan nilai probabilitas yang merupakan milik objek[6].

Berdasarkan hal yang telah dijelaskan maka *color filtering* dan metode *CamShift* ini akan digunakan dalam penelitian untuk mengembangkan penelitian sebelumnya. Aplikasi yang akan dibangun pada penelitian ini akan dinamakan *virtual writing*.

Tujuan dilakukannya penelitian ini yaitu untuk mengetahui keakuratan *tracking object* dengan dengan menggunakan metode *CamShift*.

2. ISI PENELITIAN

2.1 Deskripsi Sistem

Pada sistem yang akan dibuat akan melakukan *tracking* warna, dimana sistem ini akan menentukan titik tengah pada objek yang akan di tracking dan akan mengikuti pergerak objek tersebut. Dalam tracking, pencahayaan sangat berpengaruh dan juga warna dari objek yang akan di tracking jug hal yang penting, objek dengan warna hitam dan putih pada sistem ini, tidak dapat di tracking dengan baik.

2.2 Pengolahan Citra

Sebuah citra memiliki kaya informasi, namun sering kali citra yang kita miliki mengalami penurunan mutu (degradasi), misalnya mengandung cacat atau derau (*noise*), warnanya terlalu kontras, kurang tajam, kabur (*blurring*), dan sebagainya. Tentu saja citra semacam ini menjadi lebih sulit diinterpretasi karena informasi yang disampaikan oleh citra tersebut menjadi berkurang.

Agar citra yang mengalami gangguan mudah diinterpretsai (baik oleh manusia maupun mesin), maka citra tersebut perlu dimanipulasi menjadi citra lain yang kualitasnya lebih baik.

Pengolahan citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin (dalam hal ini komputer). Teknik-teknik pengolahan citra mentransformasikan citra menjadi citra lain. Jadi, masukannya adalah citra dan keluarannya juga citra, namun citra keluaran mempunyai kualitas lebih baik daripada citra masukan. Termasuk ke dalam bidang ini juga adalah pemampatan citra (*image compression*).

2.3 *Computer Vision*

Terminologi lain yang berkaitan erat dengan pengolahan citra adalah *computer vision* atau *machine vision*. Pada hakikatnya, *computer vision* mencoba meniru cara kerja sistem visual manusia (*human vision*). *Human vision* sesungghunya sangat kompleks. Manusia melihat objek dengan indera penglihatan (mata), lalu citra objek diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti objek apa yang tampak dalam pandangan matanya. Hasil interpretasi ini mungkin digunakan untuk pengambilan keputusan.

Computervision merupakan proses otomatis yang mengintegrasikan sejumlah besar proses untuk persepsi visual, seperti akuisisi citra, pengolahn citra, klasifikasi, pengenalan (*recognition*), dan membuat keputusan. *Computervision* terdiri dari teknik-teknik untuk mengestimasi ciri-ciri objek dalam citra, pengukuran ciri yang berkaitan dengan geometri objek, dan menginterpretasi informasi geometri tersebut.

Proses-proses dalam *computervision* dibagi dalam 3 (tiga) aktifitas :

1. Memperoleh atau mengakuisisi citra digital.
2. Melakukan teknik komputasi untuk memproses atau memodifikasi data citra(operasi-operasi pengolahan citra).
3. Menganalisis dan menginterpretasi Citra menggunakan hasil pemrosesan untuk tujuan tertentu, misalnya memandu robot, mengontrol peralatan, memantau manufaktur dan lain-lain.

2.4 *Open Computer Vision (Open CV)*

OpenCV (*Open Computer Vision*) adalah sebuah *open source library* yang berisi lebih dari 500 algoritma yang optimal untuk gambar dan analisis video. Sejak pengenalannya di tahun 1999, OpenCV telah diadopsi secara luas sebagai alat pengembangan utama oleh komunitas *researchers* dan *developes* di *computer vision*. OpenCV awalnya dikembangkan di Intel oleh Gary Bradski sebagai seorang inisiatif untuk memajukan penelitian di *vision* dan mempromosikan pengembangan dari *rich vision-based*, *CPU-intensive applications.* Setelah beberapa seri beta diluncurkan, versi 1.0 diluncurkan pada tahun 2006. Peluncuran besar kedua terjadi pada tahun 2009 dengan versi OpenCV 2 yang bertujuan penting untuk mengubah khususnya untuk antarmuka baru C++. OpenCV mengumumkan sebagai non-profit *foundation* yang bergantung pada penggalangan dana publik untuk pengembangan fiturnya.

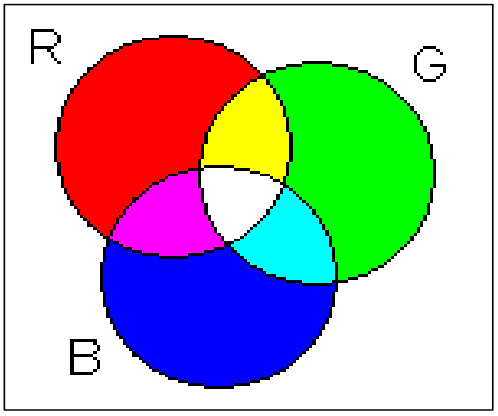
Fitur-fitur yang terdapat pada OpenCV antara lain:

1. Manipulasi data *image* (alokasi, rilis, duplikasi, pengaturan, konversi).
2. *Image* dan I/O video (masukan berbasis file dan kamera, keluaran *image/video file*).
3. Manipulasi matriks dan vektor serta aljabar linear (produk, solusi, *eigenvalues*, SVD).
4. Beragam struktur data dinamis (daftar, baris, grafik).
5. Dasar pengolahan citra (filter, deteksi tepi, deteksi suduk, pengambilan sampel dan interpolasi, konversi warna, operasi morvologi, *histogram*).

6. Analisis struktur (komponene yang berhubungan, pengolahan kontur, transformasi jarak, variasi momen, transformasi Hough, perkiraan plygonal, menyesuaikan garis, Delaunay *triangulation*.
7. Kalibrasi kamera (menemukan dan menelusuri pola kalibrasi, kalibrasi, dasar estimasi matriks, estimasi homografi, korespondensi stereo).
8. Analisis gerakan (*optical flow*, segmentasi gerakan, penelusuran).
9. Pengenalan objek (metode *eigen*, HMM).
10. Dasar Graphical User Interface atau GUI (menampilkan *image/video*, penanganan *mouse* dan *keyboard*, *scroll-bars*).
11. Pelabelan *image* (garis, poligon, gambar teks).
    Modul-modul yang terdapat pada OpenCV antara lain :
12. *cv* – fungsi utama OpenCV.
13. *cvaux* – fungsi penolong OpenCV.
14. *cxcore* –pendukung struktur data dan aljabar linear.
    highgui – fungsi GUI

2.5 Pengolahan Warna RGB

Terdapat warna pokok yang ada yaitu seperti merah,biru dan hijau, dari warna poko dapat doi kombinasikan menjadi beberapa warna lain, berikut adalah gambar RGB, dapa diliha pada gambar 1.



Gambar 1 Contoh Representasi RGB

Pengolahan warna menggunakan model RGB sangat mudah dan sederhana, karena informasi warna dalam komputer sudah dikemas dalam model yang sama [1]. Hal yang perlu dilakukan adalah melakukan pembacaan nilai-nilai R,G, dan B pada suatu piksel. Salah satu cara untuk menghitung warna yaitu dengan cara normalisasi ketiga komponen tersebut. Normalisasi penting dilakukan terutama bila sejumlah citra ditangkap dengan penerangan yang berbeda-beda. Hasil perhitungan tiap komponen warna pokok yang telah dinormalisasi akan menghilangkan pengaruh penerangan sehingga setiap komponen warna dapat dibandingkan satu sama lainnya walaupun berasal dari citra dengan kondisi penerangan yang tidak sama.

2.6 Ruang Warna HSV

Ruang warna HSV dapat diartikan pada Hue, Saturation, danValue. Model HSV menggambarkan warna-warna sama seperti bagaimana mata manusia melihat warna. RGB merupakan gabungan 3 warna dasar, sedangkan, HSV menggambarkan warna menggunakan perbandingan sepertiwarna,intensitas and pencahayaan. Berikut rumus-rumus untuk mengubah warna RGB ke HSV.
Mengubah rentang nilai RGB dari 0..255 menjadi 0..1 :

$$R' = R/255$$
$$G' = G/255$$
$$B' = B/255 \qquad (2.1)$$

Mencari nilai Cmax, Cmin, dan Delta

$$Cmax = \max(R', G', B')$$
$$Cmin = \min(R', G', B') \qquad (2.2)$$

$$\Delta = Cmax - Cmin \qquad (2.3)$$

Untuk mencari nilai H(hue) pada suatu objek dapa dilakukan dean persamaan berikut .

$$h(hue) \begin{cases} 60° \times \left(\frac{G' - B'}{max - min} mod\ 6\right), jika \max = R' \\ 60° \times \left(\frac{B' - R'}{max - min} + 2\right), jika \max = G' \\ 60° \times \left(\frac{R' - G'}{max - min} + 4\right), jika \max = B' \\ 0, \quad jika \max = min \end{cases} \qquad (2.4)$$

2.7 *Color FIltering*

Color filtering adalah suatu teknik pengolahan citra yang yang dipakai untuk memanipulasi suatu citra berdasarkan warna spesifik[7]. Cara kerjanya adalah dengan membandingkan komponen warna setiap pixel citra dengan warna spesifik. Apabila warnanya sesuai dengan warna spesifik komponen warna *pixel* tersebut dibiarkan saja. Namun, bila warnanya tidak sesuai dengan warna spesifik maka komponen warna *pixel* tersebut diubah menjadi warna background, biasanya menjadi warna hitam.
Warna yang digunakan dalam *Color Filtering* dapat direpresentasikan dalam berbagai ruang warna. Ada beberapa ruang warna yang dikenal, antara lain RGB (*Red*, *Green*, *Blue*), HSV (*Hue*, *Saturation*, *Value*), YCbCr, dsb. HSV merupakan ruang warna yang sangat cocok untuk mengidentifikasi warna-warna dasar, dimana warna dasar ini digunakan dalam

penelitian sebagai warna identifikasi robot. Selain itu, HSV menoleransi terhadap perubahan intensitas cahaya. Inilah yang menjadi keunggulan HSV dibandingkan dengan ruang warna lainnya[20].

2.8 *Continuously Adaptive Meanshift (Camshift)*

Algoritma CamShift awalnya dikembangkan untuk aplikasi *tracking* kepala dan wajah. Berdasarkan histogram warna dari target objek[6]. *CamShift* merupakan improvisasi dari algoritma *mean-shift* yang mencari probabilitas warna terbesar yang dihasilkan oleh histogram warna[6]. Algoritma ini digunakan sebagai sebuah bagian penting dari aplikasi-aplikasi lain seperti video pengawasan, interaksi manusia dan komputer, aplikasi *tracking* kepala dan wajah. Dasar dari *workflow CamShift* diilustrasikan pada gambar 1. Algoritma CamShift awalnya dikembangkan untuk aplikasi *tracking* kepala dan wajah. Berdasarkan histogram warna dari target objek[6]. *CamShift* merupakan improvisasi dari algoritma *mean-shift* yang mencari probabilitas warna terbesar yang dihasilkan oleh histogram warna[6]. Algoritma ini digunakan sebagai sebuah bagian penting dari aplikasi-aplikasi lain seperti video pengawasan, interaksi manusia dan komputer, aplikasi *tracking* kepala dan wajah. Dasar dari *workflow CamShift* diilustrasikan pada gambar 1.

Berkut langkah-langkah dari operasi CamShift :

Langkah 1 atur koordinat (x,y) sebagai inisialisasi lokasi dari pencarian *window*.

Langkah 2 menemukan nilai 0,berikut ini rumus pertama dan kedua momen gambar dari pencarian *window* :

Dimana I(x,y) adalah nilai piksel (probabilitas) pada posisi (x,y) di gambar, dan x dan y adalah rentang pencarian *window*.

$$M_{00} = \sum_x \sum_y I(x,y) \quad (2.5)$$

$$M_{10} = \sum_x \sum_y xI(x,y); \quad (2.6)$$

$$M_{01} = \sum_x \sum_y yI(x,y); \quad (2.7)$$

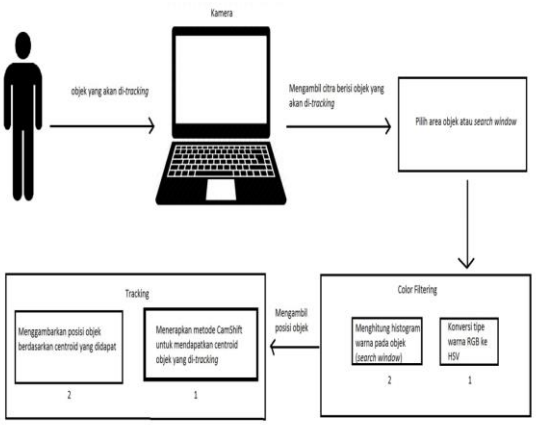
Langkah 3 menghitung lokasi *mean*(*centroid*) di pencarian *window* :

$$X_c = \frac{M_{10}}{M_{00}}; Y_c = \frac{M_{01}}{M_{00}} \quad (2.8)$$

Langkah 4 ulangi langkah 2 dan 3 sampaikonvergensi atau parameterX_c, Y_c tidak berubah atau bernilai nol.

2.9 Analisis Sistem

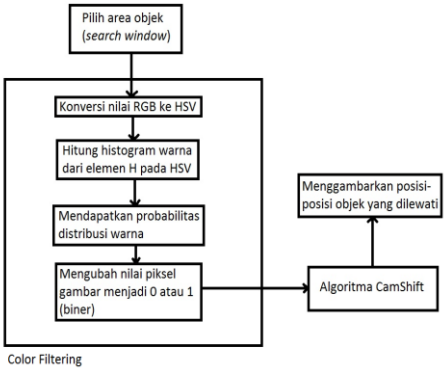
Sistem yang dibangun adalah sistem *tracking* berbasis warna dengan pergerakan marker oleh manusia yang akan ditangkap menggunakan *webcam*. Suatu sistem yang melakukan kendali terhadap warna dengan pergerakan dari manusia dan *webcam* dengan mengimplementasikan metode *color filtering* dan camshift. Sistem ini membutuhkan *input* berupa citra (video) yang diambil secara real time dan citra tersebut akan diproses dengan *color filtering* dan CamShift untuk mendapatkan warna dan centroid objek untuk melakukan proses *tracking*. Berikut gambar dari alur sistem yang akan dibangun pada Gambar 2.



Gambar 2 *Flowchart* Analisis Sistem

2.10 Analisis Metode

Dalam penelitian ini akan menggunakan metode *color filtering* dan CamShift. Algoritma *Color filtering* digunakan untuk menentukan warna yang akan di-*tracking* setelah memilih area objek yang akan dihitung histogram warnyanya. Kemudian *tracking* objek dilakukan dengan menggunakan metode *CamShift*. Berikut garis besar proses yang terjadi di dalam sistem dapat dilihat pada Gambar 3:



Gambar 3 *Flowchart* Algoritma *Tracking* Objek

Terlihat pada gambar 3 merupakan keseluruhan proses yang akan terjadi pada penelitian ini. Berikut merupakan penjelasan *diagram* proses pada sistem yang akan dibangun:

1. Pilih area objek (*search window*) untuk mendapatkan warna objek.
2. Konversi ruang warna RGB ke HSV untuk setiap piksel pada gambar.
3. Membuat histogram warna dari elemen H (*Hue*) pada HSV untuk dijadikan referensi

untuk mencari warna yang sama pada *search window*.

4. Hitung probabilitas warna dari histogram yang telah didapat.
5. Mengubah nilai piksel menjadi 0 atau 1 yang disebut proses binerisasi citra.
6. Deteksi dan mendapatkan nilai tengah dengan menggunakan algoritma CamShift.
7. Nilai tengah yang didapat dijadikan acuan untuk menggambarkan posisi objek yang dilewati.

2.10.1 konversi RGB ke HSV

Setelah sistem memilij area objek yang akan di tracking maka langkah selanjutnya adalah melakukan konversi nilai RGB ke HSV dari setiap piksel yang ada pada citra. Pada langkah ini nilai yang diambil hanyalah nilai dari elemen H(hue). Pada tahap ini data gambar yang digunakan adalah nilai RGB citra 14 x 8 yang berisi objek yang di-*tracking*(buku berwarna hijau) dimulai dari koordinat(15,9) sampai (22,22) dari data simulasi 25 x 25 yang dipilih untuk dihitung histogram warnanya. Langkah-langkah yang dilakukan adalah :

a. Mengubah nilai RGB dari 0..255 menjadi 0..1 dengan persamaan (2.1)
b. Kemudian dari hasil HSV, tahap selanjutnya mencari Cmax dan Cmin dengan menggunakan persamaan (2.2)
c. Kemudian mencari rentan nilai dari Cmax dan Cmin (2.3.)
d. Mencari niali H dengan menggunakan persamaan (2.4)

2.10.2 *Histogram* Warna

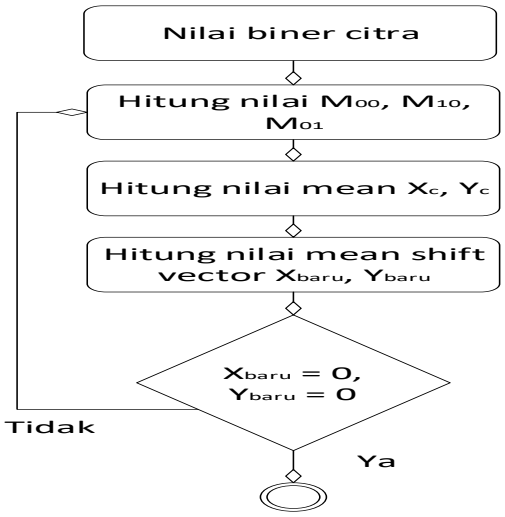
Membuat histogram warna bertujuan untuk mengelompokkan warna dalam rentang nilai tertentu. Histogram warna yang dibuat dalam penelitian ini memiliki 8 bin untuk menyimpan variasi warna dengan rentang nilai *hue* yang ditentukan berdasarkan nilai H terkecil(0) dan terbesar(356) dengan menentukan rentang nilai keduanya kemudian dibagi 8.

2.10.3 Binerisasi Citra

Binerisasi citra adalah proses mengubah nilai dari citra menjadi 0 atau 1. Nilai H yang diubah menjadi 1 adalah nilai yang masuk ke dalam frekuensi terbesar dalam perhitungan yang dilakukan pada perhitungan nilai H.

2.11 *CamShift*

CamShift adalah metode untuk *tracking* pada objek setelah dilakukan konversi warna RGB ke HSV, perhitungan histogram warna, dan binerisasi citra. Cara kerja CamShift yaitu dengan cara mencari poin data terpadat pada search window kemudian menghasilkan nilai tengah dari poin-poin data tersebut. Berikut *flowchart* dan algoritma dari CamShift, dapat dilihat pada gambar 4.



Gambar 4 *Flowchart* algoritma *Camshift*

2.11.1 Menghitung Nilai momen 0 dan 1 (M10, M01, M00)

Menghitung nilai M00 yaitu dengan menjumlahkan semua nilai poin data pada koordinat x,y yang berada pada *search window*. Menentukan nilai M00 dari area perhitungan dengan menggunakan persamaan (2.5), Pada perhitungan dibawah, I(x,y) adalah nilai pada piksel yang bernilai 1. M10 adalah nilai untuk mencari jarak antara tengah lama dan tengah baru pada sumbu x, sedangkan M01 adalah nilai untuk mencari jarak antara tengah awal dan tengah baru pada sumbu y. mencari nilai M10 dan M01 yaitu dengan menggunakan persamaan (2.6).

2.11.2 *Mean* Xc dan Yc

Mean dari Xc dan Yc bertujuan untuk mencari jarak antara koordinat lama dan baru. Untuk mencari Xc dan Yc digunakan persamaan (2.7).

2.11.3 *Mean Shift Vetor*

Mean shift vector bertujuan untuk menentukan nilai koordinat x,y baru dengan cara Xbaru = Xc + X dan Ybaru = Yc + Y. Ulangi tahap ini sampai Xbaru dan Y baru tidak berunbah. Maka ketika sudah tidak beruda itulah titik tenganh yang di dapatkan dalam objek tracking.

2.12 Pengujian Sistem

Metode ini digunakan untuk membandingkan performa dari perangkat lunak. Parameter yang digunakan dalam teknik pengujian ini antara lain:

1. Keadaan cahaya
   Keadaan cahaya dikelompokan dalam pencahayaan yang baik dan pencahayaan kurang baik.
   - Parameter dengan pencahayaan yang baik jika dalam ruangan terdapat cahaya yang dihasilkan dari matahari atau dari lampu dengan output cahaya sebesar antara 9 hingga 15 watt.

- Parameter dengan pencahayaan kurang baik jika dalam ruangan minim cahaya yang dihasilkan matahari atau dari lampu dengan output cahaya kurang dari 9 watt .

2. Kondisi *background*
   Kondisi *background* yang tertangkap oleh *webcam* tidak ada warna yang sama denga marker dan memiliki ukuran objek yang lebih besar dari ukuran marker.
3. Bentuk dan ukuran objek (marker)
   Bentuk objek ditinjau dari ukuran panjang minimal 4.5 cm dan lebar minimal 2 cm dari marker digunakan untuk pengujian.
4. Warna objek
   Warna objek yang dapat di-*tracking* adalah selain warna hitam dan putih.
5. Jarak objek dengan kamera
   Pengujian jarak dilakukan untuk mengetahui seberapa jauh objek akan terdeteksi. Parameter nilai pengujian dihitung dari jumlah terdeteksinya objek dari setiap kali pengujian. Berikut ini penjelasan penilaian pada table 1:

Tabel 1 Penilain Objek Terdeteksi

No	Jumlah Objek Terdeteksi	Penilaian
1	10	Sangat Baik
2	8-9	Baik
3	7	Cukup Baik
4	1-6	Kurang Baik
5	0	Tidak Terdeteksi

2.12.1 Pengujian

Pengujian deteksi objek dilakukan untuk menguji cahaya, jarak, dan ukuran objek yang dapat dideteksi oleh aplikasi. Jarak yang diuji dari 10 cm – 100 cm (kelipatan 10) dalam 10 kali dan menggunakan ukuran objek 1x1 cm, 4.5x2 cm, dan 10x5 cm pada cahaya baik 9-15 watt lampu dan cahaya kurang baik lebih kecil dari 9 watt atau lebih besar dari 15 watt lampu. Berikut hasil dari pengujian yang dilakukan pada jarak dan ukuran objek yang berbeda. Keterangan pada pengujian deteksi objek T = terdeteksi dan TT = tidak terdeteksi.

2.12.1.1 Pengujian pertama

Pengujian pertama dilakukan menggunakan alat sebagai berikut :

- *Webcam* 3MP
- Ukuran objek 1x1cm
- Lampu 9-15 watt (Pencahayaan baik)

Berikut hasil pengujian yang disusun kedalam tabel :

Pengujian deteksi :

Tabel 2 Pengujian Pertama Deteksi Objek

No	**Jarak Objek (cm)**	**Pencahayaan Baik**	**Pencahayaan Kurang Baik**
		Objek	**Objek**
		T	**TT**	**T**	**TT**
1	10	√		√	
2	20	√		√	
3	30	√		√	
4	40	√		√	
5	50	√		√	
6	60	√			√
7	70	√			√
8	80	√			√
9	90	√			√
10	100	√			√
Jumlah		10	0	5	5
Penilaian		**Sangat Baik**		**Kurang Baik**	

Pengujian huruf :

Tabel 3 Pengujian Pertama Penulisan Huruf

Huruf	Jarak	
	Pencahayaan Baik	Pencahayaan Kurang Baik
A-Z	10 cm – 40 cm	10 cm – 20 cm

Pengujian kata :

Tabel 4 Pengujian Pertama Penulisan Huruf

Kata	**Jarak Objek (cm)**					
	Pencahayaan Baik			**Pencahayaan kurang baik**		
	10	**50**	**100**	**10**	**50**	**100**
SAYA	√	-	-	√	-	-
PERGI	√	-	-	√	-	-
MAKAN	√	-	-	√	-	-
Jumlah Kata Terbaca	3	0	0	3	0	0

2.12.1.2 Pengujian Kedua

Pengujian kedua dilakukan menggunakan alat sebagai berikut :

- *Webcam* 3MP
- Ukuran objek 4.5x2cm
- Lampu 9-15 watt (Pencahayaan baik)

Berikut hasil pengujian yang disusun kedalam tabel :

Pengujian deteksi :

Tabel 5 Pengujian Kedua Deteksi Objek

No		**Pencahayaan Baik**	**Pencahayaan Kurang Baik**

	Jarak Objek (cm)	Objek		Objek	
		T	TT	T	TT
1	10	√		√	
2	20	√		√	
3	30	√		√	
4	40	√		√	
5	50	√		√	
6	60	√			√
7	70	√			√
8	80	√			√
9	90	√			√
10	100	√			√
Jumlah		10	0	5	5
Penilaian		**Sangat Baik**		**Kurang Baik**	

Pengujian huruf :

Tabel 6 Pengujian Kedua Penulisan Huruf

Huruf	Jarak	
	Pencahayaan Baik	Pencahayaan Kurang Baik
A-Z	10 cm – 100 cm	10 cm – 50 cm

Pengujian kata :

Tabel 7 Pengujian Kedua Penulisan Kata

Kata	Jarak Objek (cm)					
	Pencahayaan Baik			Pencahayaan kurang baik		
	10	50	100	10	50	100
SAYA	√	√	√	√	√	-
PERGI	√	√	√	√	√	-
MAKAN	√	√	√	√	√	-
Jumlah Kata Terbaca	3	3	3	3	3	0

2.12.1.3 Pengujian Ketiga

Pengujian kedua dilakukan menggunakan alat sebagai berikut :

- *Webcam* 3MP
- Ukuran objek 4.5x2cm
- Lampu 9-15 watt (Pencahayaan baik)

Berikut hasil pengujian yang disusun kedalam tabel :

Pengujian deteksi :

Tabel 8 Pengujian Ketiga Deteksi Objek

No	Jarak Objek (cm)	Pencahayaan Baik		Pencahayaan Kurang Baik	
		Objek		Objek	
		T	TT	T	TT
1	10	√		√	
2	20	√		√	
3	30	√		√	
4	40	√		√	
5	50	√		√	
6	60	√			√
7	70	√			√
8	80	√			√
9	90	√			√
10	100	√			√
Jumlah		10	0	5	5
Penilaian		**Sangat Baik**		**Kurang Baik**	

Pengujian huruf :

Tabel 9 Pengujian Ketiga Penulisan Huruf

Huruf	Jarak	
	Pencahayaan Baik	Pencahayaan Kurang Baik
A-Z	30 cm – 100 cm	30 cm – 50 cm

Pengujian kata :

Tabel 10 PengujianketigaPenulisan Kata

Kata	Jarak Objek (cm)					
	Pencahayaan Baik			Pencahayaan kurang baik		
	10	50	100	10	50	100
SAYA	-	√	√	-	√	-
PERGI	-	√	√	-	√	-
MAKAN	-	√	√	-	√	-
Jumlah Kata Terbaca	0	3	3	0	3	0

2.12.2 Kesimpulan Pengujian

Berdasarkan pengujian yang dilakukan, hasil terbaik didapat pada pengujian kedua. Pengujian kedua dapat mendeteksi objek pada pencahayaan baik di jarak 10 cm – 100 cm, menghasilkan huruf A-Z yang dapat dibaca di jarak 10 cm – 100 cm, dan kata (“SAYA”, “PERGI”, “MAKAN”) yang dihasilkan dapat terbaca pada penulisan di jarak 10 cm – 100 cm.

Kesimpulan yang didapat bahwa ukuran objek mempengaruhi hasil penulisan pada jarak tertentu, objek yang terlalu besar akan menimbulkan hasil yang kurang baik pada jarak yang dekat. Jarak mempengaruhi kemampuan deteksi pada ukuran objek tertentu, semakin jauh jarak kemampuan deteksi akan berkurang pada objek yang kecil. Sedangkan pada cahaya yang kurang baik akan mempengaruhi warna objek yang berdampak pada kemampuan deteksi objek juga.

3. PENUTUP

3.1 Kesimpulan

Berdasarkan dari hasil penelitian, analisis sistem, perancangan sistem , dan pengujian sistem pada perangkat *virtual writing* ini maka dapat diambil kesimpulan bahwa:

1. Penelitian ini telah berhasil mendeteksi dan *tracking* objek secara *realtime* dengan menggunakan metode *color filtering* dan *camshift.*
2. Aplikasi dapat digunakan untuk menulis secara virtual tanpa menggunakan mouse.

3.2 Saran

Penelitian ini masih terdapat kekurangan pada deteksi objek jika terjadi kondisi cahaya yang kurang baik. Diharapkan penelitian selanjutnya agar dapat mengatasi masalah ini dengan metode-metode tertentu.

DAFTAR PUSTAKA

[1] U. Ahmad, Pengolahan Citra Digital dan Pemogramannya, Yogyakarta: Graha Ilmu, 2005.

[2] R. Munir, Pengolahan Citra Digital dengan pendekatan algoritmik, Bandung: Informatika Bandung, 2004.

[3] Fadliansyah, Computer Vision dan Pengolahan Citra, Yogyakarta: ANDI, 2007.

[4] K. Nummiaro, E. Koller-Meier and L. V. Gool, "An Adaptive Color-Based Particle Filter," *Preprint submitted to Elsevier Science*, 2002.

[5] R. Pradhitya, "Pembangunan Aplikasi Deteksi dan Tracking Warna *Virtual Drawing* Menggunakan Algoritma *Color Filtering*," Universitas Komputer Indonesia, Bandung, 2015.

[6] A. D. Mohammed, T. Morris, " An Improved CAMShift Algorithm for Object Detection and Extraction," University of Manchester, Manchester, UK.

[7] A. R. Mulyanto, "Rancang Bangun Aplikasi Pendeteksi Bentuk Dan Warna Benda Pada *Mobile Robot* Berbasis *Webcam*", STIKOM Surabaya.

[8] D. Putra, Pengolahan Citra Digital, Yogyakarta, 2010.

[9] S. J. Prince, Computer Vision: Models, Learning and Inference, Cambridge University Press, 2012.

[10] M. H. Purnomo and A. Muntasa, Konsep Pengolahan Citra Digital dan Ekstrasi Fitur, Yogyakarta: Graha Ilmu, 2010.

[11] A. Basuki, "Aplikasi Pengolahan Citra Deteksi Warna," PENS-ITS, Surabaya, 2009.

[12] G. S. Nahla, S. and D. Pramadihanto, "Tracking Bola Menggunakan Robotino". 2014.

[13] L A Mushawwir, S. I Supriana, "Deteksi dan Tracking Objek untuk Sistem Pengawasan Citra Bergerak" Institut Teknologi Bandung, Bandung.