

IMPLEMENTASI SCALE INVARIANT FEATURE TRANSFORM-HISTOGRAM OF GRADIENTS DAN MULTICLASS SVM UNTUK HANDTRACKING PADA KONTROL MOUSE POINTER

Anggy Edo Prasetya¹, Irfan Maliki, S.T, M.T²

^{1,2} Teknik Informatika-Universitas Komputer Indonesia

Jln. Dipatiukur No. 112-116, Lebakgede, Coblong, Kota Bandung, Jawa Barat 40132

E-mail : anggyedoo@gmail.com¹, irfanmaliki007@gmail.com²

ABSTRAK

Teknik dialog interaktif semakin memungkinkan terjadinya komunikasi antara manusia dan komputer yang memiliki banyak variasi. Teknik interaksi tersebut merupakan keterlibatan fisik secara langsung dengan sistem komputer dan perangkat masukan atau biasa disebut Interaksi Manusia dan Komputer (*Human Computer Interaction*) Salah satu interaksi yang termasuk Interaksi Manusia dan Komputer adalah *handtracking* untuk melakukan kontrol event pointer. Untuk melakukan proses *handtracking* diperlukan beberapa metode di awal (*preprocessing*) seperti RGB to HSV, pencarian warna HSV merah (*colour detection*), opening (erosi dan dilasi), *closing*(dilasi dan erosi), *Different of Gaussian* (DoG) dari SIFT, mencari ROI tangan dan melakukan *Image Moments* dan untuk melakukan proses *handrecognition* diperlukan metode *Histogram of Gradients* untuk ekstraksi ciri dan *Support Vector Machine* untuk klasifikasi beberapa event pointer seperti menggerakkan pointer, klik kiri, klik kanan, *scrolling* dan *drag and drop*. Berdasarkan hasil evaluasi pengujian sistem yang telah dilakukan untuk semua skenario pengujian seperti menggerakkan pointer ke ujung kanan layar *desktop*, klik kiri *icon windows*, klik kanan pada layar *desktop*, *drag and drop* program *word* dan *scrolling* pada dokumen PDF didapatkan rata – rata akurasi sebesar 78%.

Kata kunci : *Handtracking, Handrecognition, Colour detection, SIFT, HOG, SVM, Computer Vision*

1. PRELIMINARY

Interaksi manusia dan komputer biasanya dilakukan dalam pengenalan suara, layar sentuh, pengenalan gesture dan *handtracking*. *Handtracking* merupakan pengenalan pada bentuk dan pergerakan dari tangan manusia dilakukan untuk meraih tujuan tertentu seperti pengenalan bahasa isyarat, interaksi manusia dan robot dan untuk mengontrol mouse pointer langsung dengan tangan manusia melalui kamera webcam [1]. Komputer dalam

pengoperasiannya umumnya menggunakan hardware atau alat bantu tertentu untuk melakukan masukkan perintah pada komputer contohnya mouse, keyboard dan webcam [2]. Penelitian tentang *handtracking* akan berusaha memaksimalkan kinerja dari alat inputan untuk bisa berinteraksi antara manusia dan komputer [3].

Penelitian tentang hand tracking umumnya menggunakan metode *skin colour detection* sebagai metode segmentasi untuk bisa mendeteksi tangan lewat rentang nilai warna kulit manusia pada ruang warna YCbCr seperti penelitian yang menggunakan algoritma *pyramidal lucas kanade* sebagai proses *handtracking* [4]. Penggunaan metode *skin colour detection* mendapatkan hasil yang baik hanya jika menampilkan daerah telapak tangan karena spesifik pada warna kulit manusia, tetapi akan mendapatkan hasil yang kurang baik ketika daerah telapak tangan kebawah atau daerah yang mengandung warna kulit manusia lainnya seperti wajah ditampilkan. Karena metode tersebut akan mendeteksi daerah lain seperti daerah telapak tangan kebawah dan daerah wajah sehingga menyebabkan proses *handtracking* gagal dilakukan.

Sebelumnya sudah ada penelitian tentang *handtracking* untuk interaksi manusia dan robot menggunakan algoritma *adaboost* dengan SIFT(*Scale-invariant feature transform*) dengan memberikan tingkat akurasi sebesar 90% [5]. Penelitian selanjutnya pernah dilakukan menggunakan algoritma *adaptive HOG* (*Histogram of Gradients*)-LBP(*Local Binary Pattern*) untuk mendeteksi telapak tangan (*palm tracking*) dengan memberikan ketepatan akurasi sebesar 95.2% [6]. Mengacu pada penelitian menggunakan algoritma SIFT(*Scale-invariant feature transform*)-HOG(*Histogram of Gradients*) dan Multiclass SVM(*Support Vector Machine*) bisa mendapatkan akurasi sebesar 97% [7].

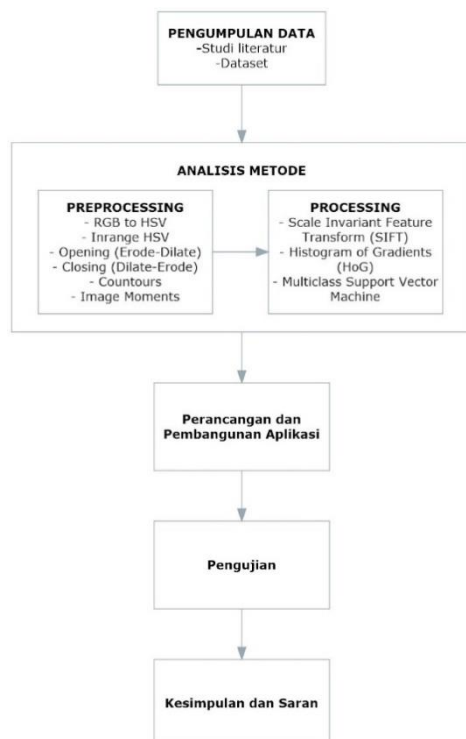
Oleh karena itu, berdasarkan pemaparan masalah yang telah dijelaskan penelitian ini diharapkan bisa mendapatkan hasil hand tracking yang baik dengan menggunakan algoritma SIFT dan HOG sebagai ekstraksi fitur dan Multiclass SVM sebagai klasifikasinya pada kasus yang berbeda yaitu hand tracking pada mouse pointer yang bisa melakukan event mouse pointer dengan tangan.

2. RESEACRH CONTENT

Pada bagian ini membahas mengenai metode penelitian, landasan teori, analisis sistem, dan hasil pengujian.

2.1 Metode Penelitian

Metode penelitian yang dilakukan menggunakan metode penelitian berdasarkan studi kasus. Studi kasus, atau penelitian kasus (*Case study*), adalah penelitian status subjek berdasarkan penelitian yang berkenaan dengan khas dari keseluruhan personalitas [8]. Adapun langkah – langkah penelitian seperti yang terlihat pada Gambar 1



Gambar 1 Langkah – langkah penelitian

2.2 Computer Vision

Pada hakikatnya nya, *Computer Vision* mencoba meniru cara kerja sistem visual manusia (*Human Vision*). *Human Vision* sesungguhnya sangat kompleks. Manusia melihat objek melalui indra penglihatan (mata), lalu citra objek diteruskan ke otak untuk diinterpretasi sehingga manusia mengerti objek apa yang terlihat dalam pandangan matanya. Hasil interpretasi ini memungkinkan untuk pengambilan keputusan [9].

2.3 Citra HSV

HSV merupakan singkatan dari *Hue*, *Saturation* dan *Value*. Keuntungannya adalah banyak warna-warna yang sama dengan yang diambil oleh indra manusia. Sedangkan warna yang dibuat model lain seperti RGB merupakan hasil campuran dari warna-warna primer. Ciri dari HSV tersebut adalah sebagai berikut :

1. *Hue*, merupakan warna sebenarnya seperti merah, violet dan kuning. Digunakan untuk menentukan kemerahan (*Redness*), kehijauan (*Greeness*) dan lain sebagainya.
2. *Saturation*, atau disebut dengan *Chroma* adalah kemurnian atau kekuatan warna.
3. *Value*, kecerahan dari warna. Nilainya berkisar antara 0-100%. Apabila nilainya 0 maka warnanya akan menjadi hitam, semakin besar nilai maka akan semakin cerah dan muncul variasi baru dari warna tersebut.

Berikut ini merupakan perhitungan nilai HSV dari pengubahan citra RGB menjadi citra HSV [10] :

$$r = \frac{R}{(R+G+B)}, g = \frac{G}{(R+G+B)}, b = \frac{B}{(R+G+B)} \quad (1)$$

$$V = \max(r, g, b) \quad (2)$$

$$S = \begin{cases} 0, & \text{jika } V = 0 \\ 1 - \frac{\min(r, g, b)}{V}, & \text{jika } V > 0 \end{cases} \quad (3)$$

$$H = \begin{cases} 0, & \text{jika } S = 0 \\ \frac{60 * (g - b)}{(S * V)}, & \text{jika } V = r \\ 60 * \left[2 + \frac{(b - r)}{(S * V)} \right], & \text{jika } V = g \\ 60 * \left[4 + \frac{(r - g)}{(S * V)} \right], & \text{jika } V = b \end{cases} \quad (4)$$

$$H = H + 360, \text{ jika } H < 0 \quad (5)$$

R,G,B adalah nilai dari citra RGB yang akan dikonversikan ke citra HSV.

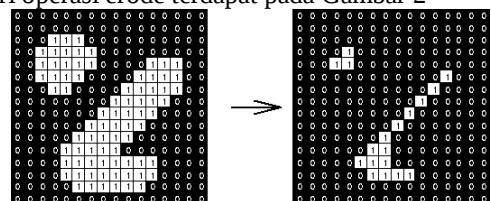
2.4 Erode

Operasi erode (Erosi) merupakan proses memperkecil struktur citra. Operasi erosi dapat dirumuskan sebagai berikut [11]:

$$A \ominus B = \{ p \in z^2 \mid (a + b) \in |, b \in B \} \quad (6)$$

Dimana A merupakan $f(x,y)$ dari citra asli dan B adalah elemen penstruktur. Elemen penstruktur yang umumnya digunakan dalam operasi erosi adalah bentuk kotak. Bentuk elemen penstruktur lainnya berupa *ellipse*, garis, piringan dan lainnya.

Hasil erosi merupakan operasi nalar AND dari setiap koordinat A dan B. Berikut merupakan hasil dari operasi erode terdapat pada Gambar 2



Gambar 2 Hasil Operasi Erode/Erosi

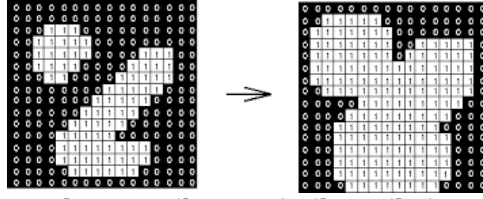
2.5 Dilate

Operasi dilate (Dilasi) merupakan proses pelebaran terhadap piksel bernilai 1. Operasi dilasi dapat dirumuskan sebagai berikut [11]:

$$A \oplus B = \{ z \mid z = a + b, a \in A, b \in B \} \quad (7)$$

Dimana A merupakan $f(x,y)$ dari citra asli dan B adalah elemen penstruktur. Elemen penstruktur yang biasa digunakan dalam operasi dilasi juga umumnya berbentuk kotak.

Hasil dilasi berupa penjumlahan seluruh pasangan koordinat dari A dan B. Berikut merupakan hasil dari operasi dilate terdapat pada Gambar 3



Gambar 3 Hasil Operasi Dilate/Dilasi

2.6 Histogram of oriented Gradients

Histogram of oriented Gradients merupakan cara umum untuk memperoleh deskriptor untuk deteksi objek tertentu. Sebagai contoh, untuk mendeteksi keberadaan manusia (*Human detection*) seperti penelitian yang pernah dilakukan oleh Dalal dan Trigs [12]. Proses dari algoritma ini dapat dijelaskan sebagai berikut [1]:

1. *Preprocessing*, melakukan intensitas normalisasi atau dengan kata lain mengubah citra gambar menjadi *greyscale* (Citra berskala keabuan).
2. Menghitung *edge map*. Mengestimasi arah x dan y pada gambar lalu menghitung gradien *magnitudes* dan gradien *angle* untuk setiap pixel gambar. Berikut ini merupakan rumus perhitungan untuk gradien *magnitudes* dan gradien *angle*:

$$|G| = \sqrt{I_x^2 + I_y^2} \quad (8)$$

Dimana $|G|$ adalah bilangan absolut dari gradien *magnitudes* dan I adalah citra gambar yang sudah *greyscale* dari hasil tahapan sebelumnya. I_x merupakan matrik terhadap sumbu-x dan I_y merupakan matrik terhadap sumbu-y. I_x dan I_y dapat dihitung dengan rumus perhitungan sebagai berikut:

$$I_x = I * D_x \text{ dan } I_y = I * D_y \quad (9)$$

D_x adalah *mask* $[-1 \ 0 \ 1]$, sedangkan D_y adalah *mask* $\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$ masing – masing dihitung dengan cara konvolusi (tanda *). Kemudian gradien *angle* ke dalam koordinat sumbu dengan sudut diantara 0 sampai 180 dapat dihitung dengan rumus perhitungan sebagai berikut:

$$\theta = \arctan\left(\frac{I_x}{I_y}\right) \quad (10)$$

3. *Spatial binning*. Tahapan selanjutnya adalah melakukan perhitungan *histogram* dari gradien *angle* ke tiap-tiap *cell*. Setiap pixel dalam sebuah *cell* mempunyai nilai *histogram* nya sendiri – sendiri berdasarkan nilai yang dihasilkan dalam perhitungan gradien yang kemudian dilakukan normalisasi pada setiap blok. *Cell* memiliki ukuran 8x8 pixel pada sebuah citra. Sedangkan blok memiliki ukuran 2x2 *cell*.
4. *Normalize voting values for generating a descriptor*. Nilai normalisasi fitur blok

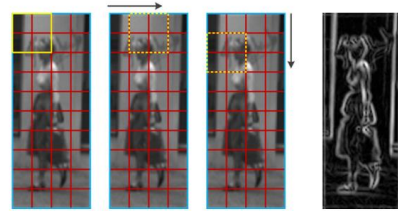
selanjutnya didapat dengan rumus perhitungan sebagai berikut [13]:

$$norm = \frac{v(n)}{\sqrt{(\sum_{k=1}^{block \cdot l} v(k)^2) + 1}} \quad (11)$$

Nilai v merupakan nilai gradien *magnitudes* sedangkan n adalah jumlah *bins* dan *block* (2x2 *cell*) merupakan jumlah *cell* sedangkan l merupakan jumlah blok yang tidak *overlap*. Fitur blok dinormalisasi untuk mengurangi efek perubahan kecerahan obyek pada satu blok.

5. *Augment all block vectors consecutively*. Setelah fitur blok dinormalisasi, nilai normalisasi setiap blok akan digabungkan menjadi satu vektor (vektor 1 dimensi) hasil satu vektor inilah bisa disebut sebagai fitur vektor *Histogram of oriented Gradients*.

Berikut ini merupakan ilustrasi dari cara kerja *Histogram of oriented Gradients* terdapat pada Gambar 4



Gambar 4 Histogram of oriented Gradients

2.7 Multiclass Support Vector Machine

Ada dua cara untuk mengimplementasikan *multiclass SVM* yaitu dengan melakukan penggabungan beberapa SVM biner dan penggabungan semua data yang terdiri dari beberapa kelas ke dalam bentuk permasalahan optimasi. Berikut metode yang umum digunakan untuk mengimplementasikan *multiclass SVM* dengan pendekatan sebagai berikut [14]:

1. Metode *one-against-all*

Dengan menggunakan metode ini, akan dibangun k buah model SVM biner (k adalah jumlah kelas). Contohnya, terdapat permasalahan klasifikasi dengan 4 buah kelas. Untuk pelatihan digunakan 4 buah SVM biner seperti pada Tabel 1 dan penggunaannya dalam mengklasifikasi kelas pada data baru dapat dilihat pada persamaan sebagai berikut [15]:

$$\text{Kelas } x = \arg \max_{i=1..k} ((w^{(i)})^T \cdot \phi(x) + b^{(i)}) \quad (12)$$

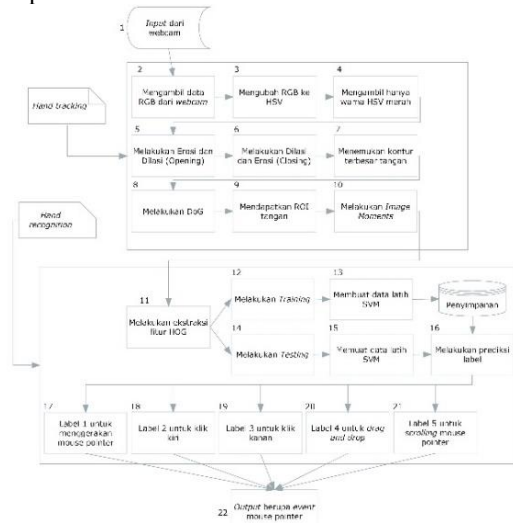
Dengan menentukan *hyperplane* terbesar pada nilai x maka akan mengklasifikasikan kelas tersebut.

Tabel 1 4 SVM biner metode *One-against-all*

$y_i = 1$	$y_i = -1$	Hipotesis
Kelas 1	Bukan kelas 1	$f^1(x) = (w^1)x + b^1$
Kelas 2	Bukan kelas 2	$f^2(x) = (w^2)x + b^2$
Kelas 3	Bukan kelas 3	$f^3(x) = (w^3)x + b^3$
Kelas 4	Bukan kelas 4	$f^4(x) = (w^4)x + b^4$

2.9 Analisis Sistem

Sistem yang dibangun terdiri dari 2 buah tahap. Tahap pertama yaitu tahap *Hand tracking* yaitu pendeteksian letak posisi ROI (*Region of Interest*) tangan lalu selanjutnya akan dilakukan tahap *Hand recognition* yang dimana pada tahapan ini akan dilakukan *training* dan *testing*. Tahapan *training* akan dilakukan dengan ekstraksi fitur HOG dan membuat data latih SVM. Tahapan *testing* akan memuat data latih SVM dan melakukan prediksi label. Gambaran deskripsi umum sistem dapat dilihat pada Gambar 5



Gambar 5 Gambaran deskripsi umum sistem

Penjelasan dari tahapan – tahapan diatas adalah sebagai berikut :

1. *Input* dari webcam

Merupakan tahapan awal untuk mendapatkan citra yang berasal dari webcam.

2. Mengambil data RGB dari webcam

Proses dalam tahap ini adalah mengubah nilai RGB pada setiap pixel citra menjadi nilai citra HSV. Misalnya diambil contoh satu *frame* citra RGB dari tahap *Hand tracking* dengan citra RGB berukuran 320 x 240 seperti pada Gambar 6



Gambar 6 Contoh citra RGB

3. Mengubah RGB ke HSV

Merupakan proses pengubahan data matriks RGB menjadi HSV. Dilakukan untuk kebutuhan pengambilan nilai warna HSV merah. Sebagai contoh ambil nilai RGB pada pixel (1,1), maka proses perhitungan RGB ke HSV menggunakan persamaan 1 sampai 5 sebagai berikut :

$$r = \frac{R}{(R + G + B)}, g = \frac{G}{(R + G + B)}, b = \frac{B}{(R + G + B)}$$

$$r = \frac{0}{(0 + 255 + 0)}, g = \frac{255}{(0 + 255 + 0)}, b = \frac{0}{(0 + 255 + 0)}$$

$$r = 0 \quad g = 1 \quad b = 0$$

$$V = \max(r, g, b)$$

$$V = 1$$

$$S = \begin{cases} 0, & \text{jika } V = 0 \\ 1 - \frac{\min(r, g, b)}{V}, & \text{jika } V > 0 \end{cases}$$

$$S = 1 - \frac{0}{1}$$

$$S = 1$$

$$H = \begin{cases} 0, & \text{jika } S = 0 \\ \frac{60 * (g - b)}{(S * V)}, & \text{jika } V = r \\ 60 * \left[2 + \frac{(b - r)}{(S * V)} \right], & \text{jika } V = g \\ 60 * \left[4 + \frac{(r - g)}{(S * V)} \right], & \text{jika } V = b \end{cases}$$

$$H = 60 * \left[2 + \frac{(0 - 0)}{(1 * 1)} \right]$$

$$H = 120$$

$$H = H + 360, \quad \text{jika } H < 0$$

$$H = 120$$

Jadi didapatkan nilai HSV dari pixel (1,1) yaitu H = 120, S = 1 dan V = 1. Contoh hasil dari konversi citra RGB ke citra HSV bisa dilihat pada Gambar 7 sebagai berikut :



Gambar 7 Contoh Citra HSV

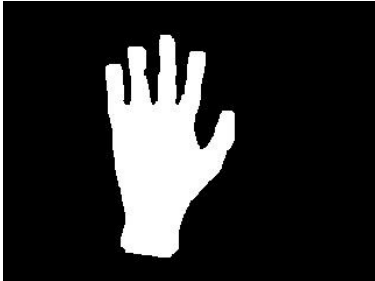
4. Mengambil hanya warna HSV merah

Merupakan proses pengambilan hanya pada warna merah HSV. Dilakukan untuk kebutuhan segmentasi tangan. Pada proses ini akan dilakukan pengambilan warna HSV hanya pada HSV warna merah dan mengabaikan warna lain untuk segmentasi tangan. Proses pengambilan warna merah HSV ini didasarkan pada skala rentang nilai HSV merah. Berikut ini merupakan skala rentang nilai warna HSV merah :

- Nilai *Hue* berada dikisaran nilai dari 0 sampai 10 atau 170 sampai 200.
- Nilai *Saturation* dikisaran nilai 70 sampai 255.
- Nilai *Value* dikisaran nilai 50 sampai 255.

5. Melakukan Erosi dan Dilasi (*Opening*)

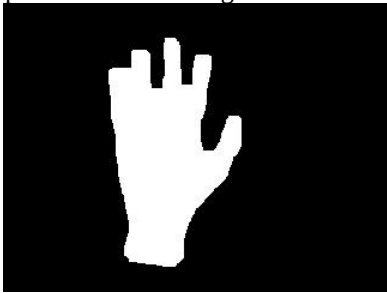
Merupakan proses penebalan data matriks yang sudah disegmentasi untuk kebutuhan memperlebar daerah ROI tangan supaya proses klasifikasi lebih mendapatkan akurasi yang baik. Contoh dari hasil operasi *opening* terdapat pada Gambar 8 sebagai berikut :



Gambar 8 Contoh hasil operasi *opening*

6. Melakukan Dilasi dan Erosi (*Closing*)

Merupakan proses penepisan data matriks yang sudah disegmentasi untuk kebutuhan daerah ROI tangan supaya proses klasifikasi lebih mendapatkan akurasi yang baik. Contoh dari hasil operasi *closing* terdapat pada Gambar 9 sebagai berikut :



Gambar 9 Contoh hasil operasi *closing*

7. Menemukan kontur terbesar tangan

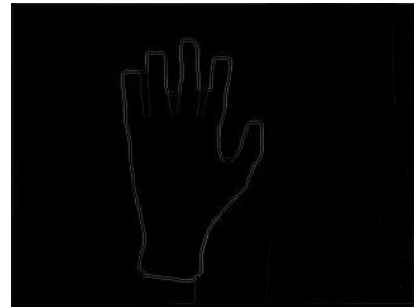
Merupakan proses pencarian kontur terbesar data matriks. Dilakukan untuk segmentasi tangan agar mendapatkan hasil yang baik. Contoh dari hasil mencari kontur terbesar tangan terdapat pada Gambar 10 sebagai berikut :



Gambar 10 Kontur terbesar tangan

8. Melakukan DoG

Merupakan proses deteksi tepi menggunakan DoG untuk segmentasi tangan agar mendapatkan hasil yang baik. Contoh hasil dari matriks citra DoG terdapat pada Gambar 11 sebagai berikut :



Gambar 11 Hasil matriks citra DoG

9. Mendapatkan ROI tangan

Merupakan proses mendapatkan data matriks hanya pada bagian ROI tangan saja. Dilakukan untuk kebutuhan ekstraksi fitur HOG, dibuat data latih dan klasifikasi SVM. Contoh dari hasil mendapatkan ROI tangan terdapat pada Gambar 12 sebagai berikut :



Gambar 12 Hasil mendapatkan ROI tangan

10. Melakukan *Image Moments*

Merupakan proses untuk mendapatkan titik tengah dari ROI tangan. Dibutuhkan untuk kebutuhan pergerakan titik x dan y dari mouse pointer. Contoh dari hasil metode *Image moments* ini dapat dilihat pada Gambar 13 sebagai berikut :



Gambar 13 Hasil dari metode *Image moments*

11. Melakukan ekstraksi fitur HOG

Merupakan proses untuk mendapatkan ekstraksi ciri tiap gambar yang akan dijadikan data latih. Sebagai contoh untuk tahapan ini akan diambil sebuah sampel data latih citra RGB matriks berukuran 200 x 200 yang dapat dilihat pada Gambar 14 sebagai berikut :



Gambar 14 Sampel data latih ukuran 200 x 200

Berikut ini merupakan proses metode HOG :

a. Ubah citra RGB menjadi citra *Grayscale*

Pada proses metode HOG diperlukan sebuah citra berskala keabuan (*Grayscale*) Sebagai contoh akan diambil nilai RGB matriks koordinat (1,1) selanjutnya akan dilakukan perhitungan untuk mendapatkan nilai *Grayscale* sebagai berikut :

$$Grayscale = (0.299 * 198) + (0.587 * 219) + (0.114 * 216)$$

$$Grayscale = (59.202) + (128.553) + (24.624)$$

$$Grayscale = 212 \text{ (dibulatkan)}$$

Contoh hasil dari konversi citra RGB data latih ke citra *Grayscale* bisa dilihat pada Gambar 15 sebagai berikut :



Gambar 15 Contoh Citra *Grayscale*

b. Menghitung *edge map*

Pada proses ini akan dilakukan perhitungan gradien *magnitudes* dan *angle* untuk setiap pixel gambar yang sudah berskala keabuan (*Grayscale*) sebelum itu akan dilakukan konvolusi dengan *mask D_x* yaitu

$$[-1 \ 0 \ 1] \text{ dan } D_y \text{ yaitu } \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} \text{ untuk mencari nilai } I_x$$

dan I_y dengan matriks *Grayscale* berukuran 4 x 4 sebagai sampel untuk dicari nilai gradien *magnitudes* dan *angle* seperti persamaan 8 sebagai berikut :

$$I_x = I * D_x \text{ dan } I_y = I * D_y$$

$$I_x = \begin{bmatrix} 212 & 212 & 213 & 214 \\ 207 & 208 & 209 & 210 \\ 203 & 204 & 205 & 206 \\ 203 & 204 & 204 & 204 \end{bmatrix} * [-1 \ 0 \ 1]$$

$$I_y = \begin{bmatrix} 212 & 212 & 213 & 214 \\ 207 & 208 & 209 & 210 \\ 203 & 204 & 205 & 206 \\ 203 & 204 & 204 & 204 \end{bmatrix} * \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$$

Sebagai contoh perhitungan konvolusi hanya pada koordinat (2,2) untuk I_x dan I_y maka hasilnya adalah :

$$I_x = | (207 * -1) + (208 * 0) + (209 * 1) |$$

$$I_x = 2$$

$$I_y = | (212 * -1) + (208 * 0) + (204 * 1) |$$

$$I_y = 8$$

Selanjutnya akan dilakukan perhitungan gradien *magnitudes* dan *angle* seperti persamaan 9 dan 10 sebagai berikut :

$$|G| = \sqrt{I_x^2 + I_y^2}$$

$$\theta = \arctan\left(\frac{I_x}{I_y}\right)$$

$$|G| = \sqrt{2^2 + 8^2}$$

$$|G| = \sqrt{4 + 64}$$

$$|G| = 8 \text{ (dibulatkan)}$$

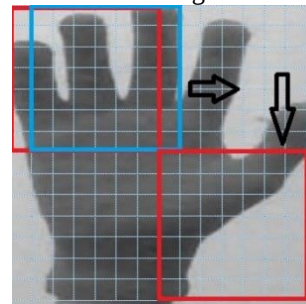
$$\theta = \arctan\left(\frac{2}{8}\right)$$

$$\theta = \arctan(0.25) \sim 14^\circ$$

Maka didapatkan nilai gradien *magnitudes* dan *angle* yaitu 8 dan 14° untuk koordinat (2,2).

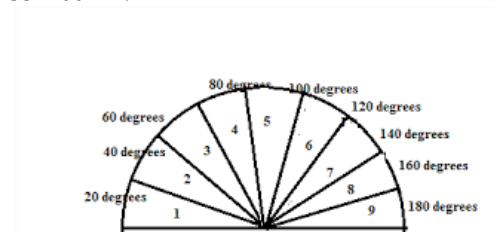
c. Melakukan *Spatial binning*

Pada proses ini akan dilakukan perhitungan *histogram* dari gradien *angle* yang sudah didapatkan diproses sebelumnya ke tiap – tiap *cell*. Setiap pixel dalam sebuah *cell* akan mempunyai nilai *histogram* nya sendiri – sendiri berdasarkan nilai yang dihasilkan dalam perhitungan gradien yang kemudian dilakukan normalisasi pada setiap blok. *Cell* untuk penelitian ini memiliki ukuran 7 x 7 yang akan terus bergerak ke arah kanan dan kebawah sampai *cell* berada pada ujung gambar seperti yang terlihat pada Gambar 16 sebagai berikut :



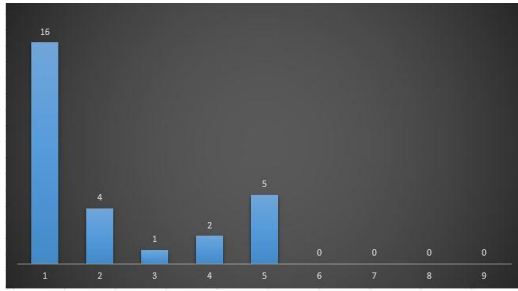
Gambar 16 Cell bergerak ke ujung gambar

Berikut ini merupakan contoh 9 *bins* dengan orientasi 0 - 180° seperti yang terlihat pada Gambar 17 berikut ini :



Gambar 17 Gambar 9 bins orientasi 0-180°

Hasil dari pemberian semua *voting* dalam bentuk sebuah *histogram* dari citra data latih terdapat pada Gambar 18 sebagai berikut :



Gambar 18 Histogram citra data latih

d. *Normalize voting values for generating a descriptor*

Langkah selanjutnya akan dilakukan perhitungan normalisasi nilai fitur blok pada setiap nilai *gradient magnitude* pada proses sebelumnya di *spatial binning* maka rumus perhitungannya seperti persamaan 11 sebagai berikut:

$$\begin{aligned}
 norm &= \frac{v(n)}{\sqrt{(\sum_{k=1}^{block \times l} v(k)^2) + 1}} \\
 &= \frac{28 * (9)}{\sqrt{(\sum_{k=1}^{7*7*16} 28 * (1)^2) + 1}} \\
 &= \frac{252}{\sqrt{(\sum_{k=1}^{784} 28) + 1}} \\
 &= \frac{252}{\sqrt{(4506251681)}} \\
 &= 0.003754
 \end{aligned}$$

Normalisasi nilai fitur blok dilakukan bertujuan untuk mengurangi efek perubahan kecerahan obyek pada setiap blok.

12. Melakukan *training*

Merupakan proses melatih data latih dari hasil ekstraksi fitur HOG menggunakan SVM. Berikut ini merupakan hasil fitur vektor HOG dari citra data latih yang telah digabungkan dapat dilihat pada Tabel 2 sebagai berikut :

Tabel 2 Hasil fitur vektor HOG citra data latih

0: 0.00375 4	1: 0.00271 6	2: 0.00305 1	3: 0.00217 1	4: 0.00083 4
5: 0.00185 5	6: 0.00163 9	7: 0.00146 1	8: 0.00066 4	9: 0.00458 9
10: 0.00150 6	11: 0.00373 6	12: 0.00337 4	...	7055: 0.00611 1

13. Membuat data latih SVM

Merupakan proses menyimpan hasil data latih SVM yang nantinya akan dimuat ketika tahap *testing* dimulai.

14. Melakukan *testing*

Merupakan proses menguji citra *real-time* dan akan dilakukan pencocokan dengan data latih SVM.

15. Memuat data latih SVM

Merupakan proses memuat data latih SVM yang sudah pernah dibuat sebelumnya.

16. Melakukan prediksi label

Merupakan proses prediksi SVM untuk memprediksi angka yang keluar dari masukkan citra *real-time* saat *testing*. sebagai contoh data uji yang sudah menemukan nilai *support vector*nya yaitu (0.043258588,-4) akan dimasukkan ke persamaan 12 sebagai berikut :

$$\begin{aligned}
 &Kelas x \\
 &= \arg \max_{k=1..5} \left(\begin{bmatrix} 336.4060992 \\ 571.7978874 \end{bmatrix}^T \cdot \begin{bmatrix} 0.043258588 \\ -4 \end{bmatrix} \right. \\
 &+ 83.96716895, \begin{bmatrix} 7.264950745 \\ 13.52758601 \end{bmatrix}^T \cdot \begin{bmatrix} 0.043258588 \\ -4 \end{bmatrix} \\
 &+ 2.040385033, \begin{bmatrix} 11.67556696 \\ 19.36271229 \end{bmatrix}^T \cdot \begin{bmatrix} 0.043258588 \\ -4 \end{bmatrix} \\
 &+ 2.988421626, \begin{bmatrix} 54.91771231 \\ 96.38651963 \end{bmatrix}^T \cdot \begin{bmatrix} 0.043258588 \\ -4 \end{bmatrix} \\
 &+ 14.44890498, \begin{bmatrix} 28.24582215 \\ 48.26470044 \end{bmatrix}^T \cdot \begin{bmatrix} 0.043258588 \\ -4 \end{bmatrix} \\
 &\left. + 7.226943864 \right)
 \end{aligned}$$

$$\begin{aligned}
 &Kelas x \\
 &= \arg \max_{k=1..5} \left(\begin{matrix} (-2188.671928, -51.7556875, \\ -73.957359 \\ -368.7215109, -184.6099835) \end{matrix} \right)
 \end{aligned}$$

$$Kelas x = -51.7556875$$

Seperti yang bisa dilihat pada persamaan diatas nilai *hyperplane* yang paling besar adalah -51.7556875 yang dimana indeks nilai *hyperplane* tersebut adalah milik kelas/label 2 yaitu untuk *event* klik kiri mouse pointer.

17. Label 1 untuk menggerakkan mouse pointer

Merupakan proses *event* mouse pointer jika angka prediksi yang keluar adalah 1.

18. Label 2 untuk klik kiri

Merupakan proses *event* mouse pointer jika angka prediksi yang keluar adalah 2.

19. Label 3 untuk klik kanan

Merupakan proses *event* mouse pointer jika angka prediksi yang keluar adalah 3.

20. Label 4 untuk *drag and drop*

Merupakan proses *event* mouse pointer jika angka prediksi yang keluar adalah 4.

21. Label 5 untuk *scrolling* mouse pointer

Merupakan proses *event* mouse pointer jika angka prediksi yang keluar adalah 5.

22. *Output* berupa *event* mouse pointer

Seperti yang sudah dijelaskan sebelumnya, keluaran dari sistem ini adalah *event mouse pointer* yang berupa pergerakan mouse pointer, klik kiri, klik kanan, *drag and drop* dan *scrolling mouse pointer*.

2.10 Evaluasi Pengujian Sistem

Pada pengujian peromansi dapat disimpulkan bahwa metode SIFT-HOG+SVM memiliki perbedaan akurasi disetiap skenario pengujiannya. Berdasarkan hasil pengujian keakuratan dari semua skenario pengujian yang telah dilakukan hasilnya dapat dilihat pada Tabel 3 sebagai berikut :

Tabel 3 Hasil akurasi semua skenario pengujian

Skenario Pengujian	Metode SIFT-HOG+SVM
Data latih MOVE	80%
Data latih CLICK1	90%
Data latih CLICK2	70%
Data latih DRAG	100%
Data latih SCROLL	50%
Rata – rata akurasi	78%

Dari data tersebut dapat diambil kesimpulan bahwa didapatkan hasil *handtracking* yang baik dengan persentase rata-rata akurasi untuk semua skenario pengujian yaitu sebesar 78%.

3. PENUTUP

3.1 Kesimpulan

Berdasarkan hasil penelitian mengenai implementasi algoritma SIFT-HOG dan SVM dalam mengontrol pointer dapat disimpulkan bahwa :

1. Hasil *handtracking* yang baik dengan persentase rata – rata akurasi untuk semua skenario pengujian yaitu sebesar 78%.

3.2 Saran

Disamping kelebihan sistem pasti akan ada kekurangan, oleh karena itu pengembangan kedepannya tentang penelitian ini merupakan saran – saran yang dapat diberikan sebagai berikut :

1. Menggunakan metode ICP (*Iterative Closest Point*) untuk melakukan proses *handtracking*.
2. Menggunakan metode CNN (*Convolutional Neural Network*) untuk klasifikasi *gesture* tangan untuk bisa mencapai akurasi yang tinggi.
3. Membuat *uniqu*e atau berbeda tiap – tiap dataset dari *gesture* tangan supaya hasil akurasi tidak mengalami penurunan.

DAFTAR PUSTAKA

- [1] R. Klette, Concise Computer Vision, New York: Springer, 2014.
- [2] V. H. Amiraj Dhawan, “Implementation of Hand Detection based Techniques for Human Computer Interaction,” International Journal of

Computer Applications, vol. 72, 2013.

- [3] D. M. Iqbal, “Computer Vision Penginderaan Visual untuk berbagai,” Foshema Research Group, Depok, 2016.
- [4] H. Kharisma, “Implementasi Handtracking Pada Kontrol Mouse Pointer Menggunakan Algoritma Pyramidal Lucas-Kanade,” Skripsi, 2011.
- [5] C.-c. wang dan K.-. C.-. Wang, “Hand Posture Recognition Using Adaboost with SIFT for Human Robot Interaction,” Graduate Institute of Networking and Multimedia.
- [6] J. Gao dan Q. Cao, “Adaptive HOG-LBP Based Learning for Palm Tracking,” International Conference on Computer and Information Application, 2012.
- [7] F. J. Ansari, “Hand Gesture Recognition using fusion of SIFT and HoG with SVM as a Classifier,” International Journal of Engineering Technology Science and Research, vol. 4, no. 9, 2017.
- [8] M. Nazir, Metode Penelitian, 2009.
- [9] R. Munir, Pengolahan citra digital dengan pendekatan Algoritmik, 2004.
- [10] A. Fauzan, “Ruang Warna Hue Saturation Value (HSV) serta Proses Konversinya,” 3 Januari 2015. [Online]. Available: <http://www.charisfauzan.net/2015/01/ruang-warna-hue-saturation-value-hsv.html>. [Diakses 22 April 2018].
- [11] A. S. Abdul Kadir, Teori dan Aplikasi Pengolahan Citra, 2013.
- [12] B. T. Navneet Dalal, “Histograms of Oriented Gradients for Human Detection,” 2005. [Online]. Available: <https://lear.inrialpes.fr/people/triggs/pubs/Dalal-cvpr05.pdf> [Diakses 16 April 2018].
- [13] M. I. T. Taichiro Tokumori, “Histograms of Oriented Gradients(HOG),” Universitas Ryukyus, Jepang.
- [14] K. Sembiring, “Penerapan Teknik Support Vector Machine untuk Pendeteksian Intrusi pada Jaringan,” Jurnal ITB, 2007.
- [15] P. A. Octaviani, “Penerapan Metode Klasifikasi Support Vector Machine (SVM) pada Data Akreditasi Sekolah Dasar (SD) di Kabupaten Magelang,” Universitas Diponegoro, Semarang, 2014.