ANALISIS KINERJA *HAND-TRACKING-BY-DETECTION* UNTUK TEKNOLOGI HOLOGRAM INTERAKTIF MENGGUNAKAN MODEL *HIDDEN MARKOV*

PERFORMANCE ANALYSIS OF HAND-TRACKING-BY-DETECTION FOR INTERACTIVE HOLOGRAM TECHNOLOGY USING HIDDEN MARKOV MODEL

Devita Rahma Apriliani, Suryo Adhi Wibowo[2], Rissa Rahmania[3]

[1,2,3]Prodi S1 Teknik Telekomunikasi, Fakultas Teknik Elektro, Universitas Telkom, Bandung
[1]devitara@student.telkomuniversity.ac.id, [2]suryoadhiwibowo@telkomuniversity.ac.id,
[3]saniarahmani@telkomuniversity.ac.id

ABSTRAK

Hologram merupakan proyeksi 3-Dimensi dari suatu objek yang diproyeksikan pada permukaan 2-Dimensi. *Image hologram* dapat diaplikasikan dalam berbagai hal seperti untuk pembelajaran, hiburan, dan lain-lain. *Human Computer Interaction* (HCI) adalah studi yang mempelajari bagaimana manusia berinteraksi dengan komputer dan sejauh mana komputer dapat dikembangkan atau tidak dikembangkan untuk berinteraksi dengan manusia. *Hand tracking-by-detection* dapat diimplementasikan dalam interaksi antara manusia dengan hologram.

Pada tugas akhir ini dirancang sistem untuk melakukan *hand tracking-by-detection* dengan menggunakan *Hidden Markov Model*. Deteksi tangan pada sistem ini dilakukan menggunakan *haar-like features*. *State sequence* yang akan dijadikan masukan untuk HMM didapat dari proses deteksi. Pada tugas akhir ini digunakan 2 *gesture* tangan yaitu *open* dan *close*.

Pengujian dilakukan dengan menggunakan *dataset* dengan data positif sebanyak 1600 dan 1800. Analisis dilakukan berdasarkan parameter presisi, IoU, dan akurasi. Dari hasil pengujian didapatkan nilai presisi 5,3015, IoU sebesar 0,5221, dan akurasi sebesar 56,67% untuk keadaan *close* dan nilai presisi 14,9993, IoU sebesar 0,5288, dan akurasi sebesar 68,72% untuk keaadaan *open* sedangkan akurasi sistem kelesuruhan tertinggi sebesar 96,33%.

Kata Kunci: Hologram, *Human Computer Interaction, Tracking-by-Detection, Hidden Markov Model*, *haar-like feature.*

ABSTRACT

Hologram is a 3-dimensional projection of an object projected on 2-dimensional surface. Holographic image can be applied in various fields such as education, entertainment, etc. Human Computer Interaction is a study about how human interact with computer and how far computer can be developed or not to interact with human. Hand tracking-by-detection can be implemented in interaction between human and hologram.

In this final project, a system is designed to do hand tracking-by-detection using Hidden Markov Model. Hand detection in this system is done using haar-like features. State sequence that would be used as input in HMM is obtained in detection process. This final project using 2 hand gesture, open and close.

The test on the system was carried out using datasets with 1600 and 1800 positive data and 800 and 900 negative data. Analysis are done based on precision, IoU, and accuracy parameters. From the test result, achieved precision 5,3015, IoU 0,5221, and accuracy 56,67% for closed hand, and precision 14,9993, IoU 0,5288, and accuracy 68,72% for open hand with the highest accuracy obtained for whole system is 96,33%.

Keywords: Hologram, *Human Computer Interaction, Tracking-by-Detection, Hidden Markov Model*, *haar-like feature.*

1 Pendahuluan

Seiring dengan perkembangan zaman, teknologi pun berkembang dengan pesat. Dengan berkembangnya teknologi, kebutuhan manusia akan teknologi semakin meningkat. Berbagai teknologi dirancang untuk mempermudah pekerjaan dan kebutuhan manusia. Seiring berkembangnya teknologi, muncul beberapa inovasi dan gagasan baru salah satunya adalah *image hologram*. *Image hologram* dapat diaplikasikan dalam berbagai hal seperti untuk pembelajaran, hiburan, penelitian, dan lain-lain. Untuk mempermudah dan memberikan kenyamanan dalam penggunaan teknologi, berbagai macam metode *Human Computer Interaction* (HCI) telah dikembangkan. HCI adalah studi yang mempelajari bagaimana manusia berinteraksi dengan komputer dan sejauh mana komputer dapat dikembangkan atau tidak dikembangkan untuk berinteraksi dengan manusia [1]. Salah satu metode HCI yang banyak dikembangkan adalah menggunakan *hand tracking-by-detection*. Untuk dapat menggunakan *hand tracking-by-detection* dalam HCI, akan diperlukan algoritma *tracking-by-detection*. Dalam pengembangannya, masih ada beberapa permasalahan dalam penggunaan *tracking-by-detection*.

Pada penelitian sebelumnya *tracking-by-detection* telah dilakukan dengan berbagai metode. M. K. Ahuja dan A. Singh [2] melakukan segmentasi tangan berdasarkan warna kulit sehingga pada metode tersebut tidak dapat mendeteksi tangan apabila warna tangan bukan warna kulit (menggunakan sarung tangan). Metode tersebut juga hanya bisa mendeteksi gerakan statis. S. Song, D. Yan dkk [3], juga menggunakan segmentasi tangan berdasarkan warna kulit. *Gesture detection* dengan metode tersebut sudah dapat mendeteksi gerakan dinamis akan tetapi tingkat keberhasilan penggunaan metode pada latar belakang yang kompleks masih perlu di tingkatkan. Metode tersebut bekerja lebih baik pada saat menggunakan latar dengan warna putih karena memiliki perbedaan warna yang signifikan dengan tangan. Sedangkan S. Hussain, R. Saxena, dkk [4], menggunakan metode *pretrained Convolutional Neural Network* (CNN) yang dilatih dengan *dataset* yang cukup besar. Pada penelitian tersebut, penulis berhasil mendapatkan akurasi 93,09% pada gerakan statis maupun dinamis akan tetapi *gesture* yang digunakan adalah *gesture* menggunakan satu tangan. Pada penelitian [5], telah diciptakan sebuah *table top hologram* dengan menggunakan HCI akan tetapi kualitas hologram 3-dimensi yang dihasilkan masih kurang baik karena penggunaan kabut sebagai media proyeksinya.

Pada tugas akhir ini dilakukan implementasi *hand tracking-by-detection* pada hologram menggunakan metode *Hidden Markov Model* (HMM). Pada implementasi *hand tracking-by-detection* ini menggunakan *haar-like features* sebagai ekstraksi fitur. Pada tugas akhir ini dilakukan pengujian terhadap jumlah *dataset* dan parameter *merge threshold.* Hasil dari pengujian yang dilakukan digunakan untuk melakukan analisis kinerja sistem berdasarkan parameter presisi, IoU, dan akurasi.

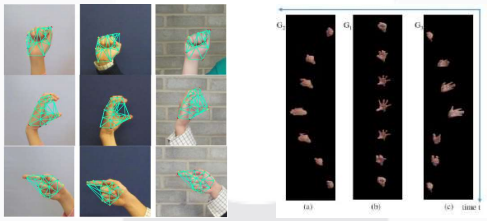
2 Tinjauan Pusataka

2.1 Hologram

Kata Holografi yang disusun dari kata-kata Bahasa Yunani "*holos*" yang memiliki arti keseluruhan dan "*gram*" yang memiliki arti catatan, sehingga Holografi dapat diartikan sebagai seluruh catatan. Holografi merupakan metode yang digunakan untuk merekam pola cahaya, dan pola ini direproduksi sebagai citra tiga dimensi yang disebut hologram [6]. Hologram dapat diaplikasikan dalam berbagai hal seperti untuk pembelajaran, hiburan, penelitian, dan lain-lain. Hologram dapat dibuat menggunakan berbagai macam media seperti prisma kaca, kipas angin, dan kabut.

2.2 *Hand Tracking-by-Detection*

Hand Tracking-by-Detection adalah suatu algoritma yang digunakan untuk mendeteksi gerak tangan yang dilakukan. Dalam penggunaannya *hand trackimg-by-detection* terbagi menjadi dua, yaitu untuk mendeteksi gerak dinamis dan statis seperti pada gambar 2.1 di mana gambar (a) merupakan contoh deteksi *gesture* statis [7] dan gambar (b) merupakan deteksi tangan dinamis [8]. Deteksi gerak statis digunakan untuk mendeteksi keadaan tangan seperti tangan terbuka atau mengepal. Deteksi gerak dinamis digunakan untuk mendeteksi pergerakan tangan.



(a) (b)

Gambar 2.1 Contoh deteksi *gesture* (a) statis, (b) dinamis

2.3 *Haar-like Features*

Haar-like features merupakan salah satu metode ekstraksi fitur. Nilai fitur pada *haar-like features* didapat dari selisih antara jumlah nilai-nilai piksel pada bagian terang dan gelap [9]. *Haar-like features* terdiri dari tiga jenis seperti dapat dilihat pada gambar 2.2. Fitur-fitur inilah yang nantinya akan disusun dan digunakan untuk mendeteksi objek.

(a) (b) (c)

Gambar 2.2 *Haar Like Features* (a) *edge*, (b) *lines*, (c) *diagonal*

Masukan yang digunakan pada ekstraksi menggunakan *haar-like features* merupakan *integral image* atau yang bisa disebut sebagai *summed area table*. Nilai piksel dari *integral image* didapatkan dari jumlah nilai piksel di atas dan sebelah kirinya. Nilai piksel *integral image* dapat dihitung menggunakan persamaan 2.1 berikut,

$$ii_{(x,y)} = \sum_{i=1}^{x}\sum_{j=1}^{y} C_{(i,j)}, \quad (2.1)$$

di mana $ii_{(x,y)}$ merupakan nilai piksel dari *integral image* pada koordinat x dan y dan $C_{(i,j)}$ merupakan nilai piksel citra asli pada koordinat x dan y. Contoh hasil dari perhitungan *integral image* dapat dilihat pada gambar 2.3.

0.4	0.1	0.6	0.7
0.1	0.3	0.7	0.9
0.3	0.2	0.6	0.7
0.1	0.2	0.8	0.8

(a)

0.4	0.5	1.1	1.8
0.5	0.9	2.2	3.8
0.8	1.4	3.3	5.6
0.9	1.7	4.4	7.5

(b)

Gambar 2.3 Citra (a) asli, (b) *integral image*

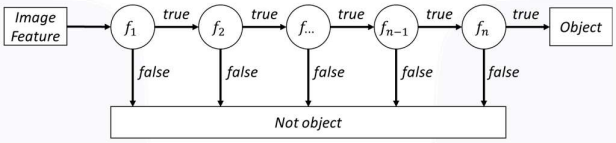
Dari *integral image* yang didapat, akan dicari nilai dari *haar-like features*. Nilai *haar-like features* didapat dari selisih jumlah nilai piksel pada bagian terang dengan jumlah nilai piksel pada bagian gelap. Nilai *haar-like features* dapat dihitung dengan persamaan 2.2.

$$\Delta = \frac{\sum ii_{dark}}{n_{dark}} - \frac{\sum ii_{light}}{n_{light}}, \quad (2.2)$$

di mana Δ merupakan nilai *haar-like features*, $\sum ii_{dark}$ merupakan jumlah nilai piksel *integral image* pada daerah gelap, $\sum ii_{ligt}$ merupakan jumlah nilai piksel *integral image* pada daerah terang, n_{dark} merupakan banyaknya piksel pada daerah gelap, dan n_{light} merupakan banyaknya piksel pada daerah terang.

2.4 *Cascade Classifier*

Cascade classifier digunakan untuk membedakan antara objek dan selain objek pada citra. *Cascade classifier* terdiri dari beberapa tahap di mana pada setiap tahap akan dilakukan penentuan ada atau tidak nya objek pada bagian dari citra [10]. Pada gambar 2.4 ditunjukkan contoh dari diagram *cascade classifier*.



Gambar 2.4 Contoh diagram *cascade classifier*

2.5 *Hidden Markov Model*

Metode HMM ini merupakan model statistik dari suatu proses markov dengan *state* tersembunyi. *State* pada HMM tidak dapat diamati secara langsung. Setiap *state* memiliki distribusi peluang *output* yang mungkin muncul sebagai suatu set proses stokastik yang akan membentuk suatu deretan observasi [11]. HMM dapat dianggap sebagai jaringan *Bayesian* dinamis yang sederhana (*simplest dynamic Bayesian network*).

Hidden Markov Model terdiri dari beberapa elemen yaitu:

1. N, yaitu jumlah keadaan (*state*). Dengan ruang *state* $S = \{s_1, s_2, \dots, s_N\}$ dan *state* pada waktu t dinyatakan dengan Q.
2. M, yaitu jumlah pengamatan (observasi) tiap *state*, dengan ruang observasi $V = \{v_1, v_2, \dots, v_N\}$.
3. $A = [a_{ij}]$, yaitu matriks peluang transisi.
4. $B = [b_{jm}]$, yaitu matriks peluang bersyarat observasi v_m jika proses berada pada *state* j, di mana:

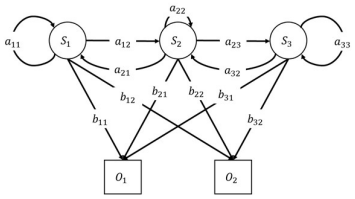
   $b_{jm} = b_j(O_t) = P(O_t = v_m | Q_t = s_j),\ 1 \le j \le N\ \ dan\ 1 \le m \le M.$
5. π_i yaitu distribusi *state* awal.

Sehingga Hidden Markov Model (HMM) dapat dituliskan dalam notasi $\lambda = (A, B, \pi)$. Jika diberikan N, M, A, B, dan π, HMM dapat digunakan sebagai pembangkit barisan observasi: $O = O_1, O_2, \dots, O_T$.

Untuk menyimpulkan hasil keluaran (*output*) berdasarkan hasil observasi pada HMM, digunakan ukuran untuk peluang yang disebut kemungkinan yang nilainya didapatkan dari persamaan 2.3.

$$L(Q_1 = s_{ij}, \dots, Q_T = s_{i_T} | O_1 = v_{m_1}, \dots, O_T = v_{m_T}) = \prod_{t=1}^{T} P(O_t = v_{m_t} | Q_t = s_{i_t}).P(Q_1 = s_{i_t}).\prod_{t=2}^{T} P(Q_t = s_{i_t} | Q_{t-1} = s_{i_{t-1}}). \quad (2.3)$$

Pada gambar 2.5 Menunjukkan contoh dari HMM dengan implementasi elemen-elemen pada HMM. Pada gambar 2.5 Terdapat S, a, b, dan O di mana s merupakan *state*, a merupakan probabilitas transisi, b probabilitas output, dan O merupakan observasi. Jumlah nilai dari $a11$ dan $a12$ bernilai satu, begitu juga dengan nilai $a21$, $a22$, $a23$, dan nilai $a32$, dan $a33$.



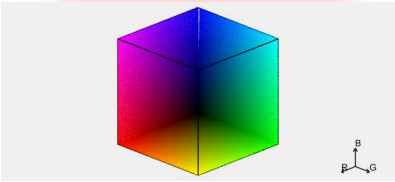
Gambar 2.5 Contoh *Hidden Markov Model*

2.6 Ruang Warna

Ruang warna merupakan suatu spesifikasi sistem koordinat dan suatu sub ruang dalam sistem tersebut dengan setiap warna dinyatakan dengan suatu titik di dalamnya. Ruang warna dibentuk dengan tujuan untuk memfasilitasi spesifikasi warna dalam bentuk standar [11]. Ruang warna yang paling sering digunakan pada perangkat komputer saat ini adalah ruang warna *Red*, *Green*, *Blue* (RGB).

2.6.1 *Red, Green, Blue (RGB)*

Ruang warna RGB merupakan ruang warna yang terdiri dari tiga komponen dasar yaitu merah (R), hijau (G), dan biru (B). Setiap piksel pada citra RGB terdiri dari ketiga komponen tersebut [12]. Model untuk ruang warna RGB biasa disajikan dalam bentuk kubu tiga dimensi dengan warna merah, hijau, dan biru pada setiap pojok sumbunya seperti pada gambar 2.6.



Gambar 2.6 Model ruang warna RGB.

2.6.2 *Grayscale*

Graycsale atau skala keabuan dapat didefinisikan sebagai tingkat keabu-abuan dari suatu citra digital. Nilai – nilai pada matriks menunjukkan kecerahan piksel. Citra *Grayscale* hanya terdiri dari warna abu-abu dengan berbagai tingkat kecerahan [12]. Tingkat kecerahan piksel dinyatakan dengan intensitas dari 0 hingga 255, 0 adalah hitam dan 255 adalah putih. Menentukan nilai dari citra *grayscale* dapat dilakukan menggunakan persamaan 2.2 berikut,

$$grayscale = (0{,}2989 \times R) - (0{,}5870 \times G) - (0{,}1141 \times B), \quad (2.2)$$

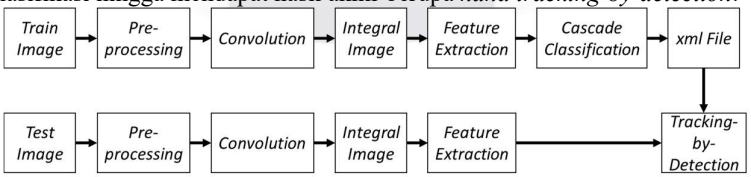
di mana R merupakan nilai dari *red* pada citra RGB, G merupakan nilai dari *green* pada citra RGB, dan B merupakan nilai dari *blue* pada citra RGB.

3 MODEL SISTEM DAN PERANCANGAN

3.1 Desain Sistem

Sistem yang digunakan pada penelitian ini terdiri dari beberapa tahap. Pada tahap pertama yaitu akuisisi citra, merupakan proses pengambilan citra yang akan digunakan untuk proses *hand tracking-by-detection*. Jenis klasifikasi yang akan digunakan pada penelitian ini adalah *Hidden Markov Model* (HMM) dengan ekstraksi fitur menggunakan *haar-like features*.

Pada gambar 3.1 dijelaskan proses dari *hand tracking-by-detection* yang akan dilakukan. Langkah pertama dalam proses *hand tracking-by-detection* adalah akuisisi citra atau pengambilan citra. Setelah dilakukan akuisisi citra, citra yang diperoleh akan melalui tahap *pre-processing* di mana citra akan diproses terlebih dahulu sebelum memasuki tahap selanjutnya. Selanjutnya, setelah melalui *pre-processing*, akan dilakukan ekstraksi fitur dan dilanjutkan dengan klasifikasi hingga mendapat hasil akhir berupa *hand tracking-by-detection*.



Gambar 3.1 Diagram blok hand tracking-by-detection

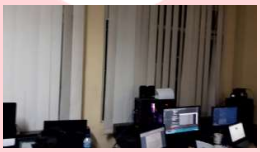
3.2 Data Latih

Data latih yang digunakan pada sistem terbagi menjadi dua, yaitu data latih positif dan negatif. Data latih positif ini terbagi menjadi 2 sesuai dengan *gesture* yang dilakukan. *Gesture* ke-1 adalah *open* yaitu kondisi tangan

terbuka dengan bagian telapak tangan menghadap ke depan. *Gesture* ke-2 adalah *close* yaitu kondisi tangan mengepal dengan bagian telapak tangan menghadap ke depan. Data latih negatif merupakan citra latar tanpa adanya tangan. Contoh data latih dapat dilihat pada gambar 3.2 dan 3.3. Data latih dihasilkan dari video 1280×720 dengan *frame rate* sebesar 30 fps. Data latih yang digunakan terbagi menjadi 2, untuk lebih jelasnya dapat dilihat pada Tabel 3.1.

(a) (b)

Gambar 3.2 Data latih positif (a) *open*, (b) *close*



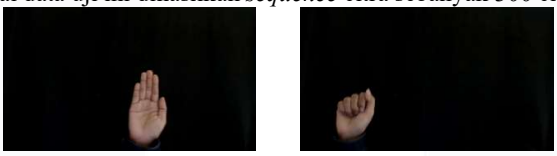
Gambar 3.3 Data latih negatif

Tabel 3.1 Rincian data latih

Kondisi Tangan	Jumlah Data Positif	Jumlah Data Negatif
Open	1600	800
Close	1600	
Open	1800	900
Close	1800	

3.2.1 *Data Uji*

Data uji yang digunakan pada sistem merupakan video tangan dalam keadaan *open* dan *close* dengan bagian telapak tangan menghadap ke depan dengan latar berwarna hitam dan latar tidak polos seperti pada gambar 3.4 dan 3.5. Data uji dihasilkan dari video 1280×720 dengan *frame rate* sebesar 30 fps berdurasi 10 detik. Dari video yang digunakan sebagai data uji ini dihasilkan *sequence* citra sebanyak 300 citra.



(a) (b)

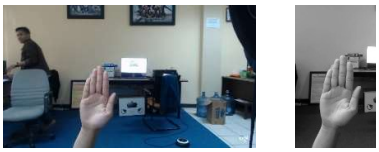
Gambar 3.4 Data uji (a) *open,* (b) *close* latar hitam

(a) (b)

Gambar 3.5 Data uji (a) *open*, (b) *close*

3.2.2 *Pre-processing*

Pada proses *pre-processing* pada citra, dilakukan proses konversi dari RGB menjadi *grayscale*. Pada data latih selain konversi dari RGB menjadi *grayscale*, dilakukan juga proses *cropping* pada data positif. Pada data positif dengan tangan kondisi *open* di *crop* menjadi berukuran 360×640 sedangkan pada kondisi *close* berukuran 90×90. Proses ini dilakukan pada data latih maupun data uji. Dari proses *pre-processing* ini dihasilkan citra seperti pada gambar 3.6 dan 3.7.



(a) (b)

Gambar 3.6 Data latih (a) sebelum, (b) sesudah *pre-processing*

(a) (b)

Gambar 3.7 Data uji (a) sebelum, (b) sesudah *pre-processing*

3.2.3 *Convolution*

Proses konvolusi dilakukan untuk menentukaan keberadaan objek pada citra. konvolusi dilakukan pada citra asli dan filter. Filter yang digunakan adalah *haar-like features*. Contoh dari proses konvolusi dengan nilai *stride* 1 dapat dilihat pada gambar 3.8.

0.2	0.3	0.8	1.5
0.6	1.2	2.5	4.1
1.3	2.3	4.2	6.1
1.5	2.8	5.5	8

(a)

1	1
-1	-1

(b)

0.3	0.6	1.2	0.7
0.9	1.3	1.7	0.9
1.1	1	0.9	0.3
0.5	1.1	1.4	0.6

(c)

Gambar 3.8 Matriks (a) citra asli, (b) filter, (c) hasil konvolusi

3.2.4 *Ekstraksi Fitur*

Ekstraksi fitur bertujuan untuk mendapatkan nilai-nilai fitur dari citra yang diproses. Nilai fitur pada proses ini didapat dari nilai piksel pada daerah citra yang terdapat objek. Untuk mencari nilai tersebut, citra akan diubah terlebih dahulu menjadi *integral image* seperti pada gambar 3.9.

0.2	0.1	0.5	0.7
0.4	0.5	0.8	0.9
0.7	0.4	0.6	0.3
0.2	0.3	0.8	0.6

(a)

0.2	0.3	0.8	1.5
0.6	1.2	2.5	4.1
1.3	2.3	4.2	6.1
1.5	2.8	5.5	8

(b)

Gambar 3.9 Citra (a) asli, (b) *integral image*

Dari *integral image* pada gambar 3.9, akan dilakukan perhitungan jumlah nilai piksel pada daerah yang dipilih. Pada gambar 3.10 ditunjukkan daerah pada citra yang akan dihitung dengan ditandai menggunakan warna abu. Perhitungan jumlah nilai piksel daerah tersebut dilakukan menggunakan persamaan 3.1

$$s = a + d - (b + c) \tag{3.1}$$

0.2 (a)	0.3	0.8 (b)	1.5
0.6	1.2	2.5	4.1
1.3 (c)	2.3	4.2 (d)	6.1
1.5	2.8	5.5	8

Gambar 3.10 Perhitungan jumlah nilai piksel area citra

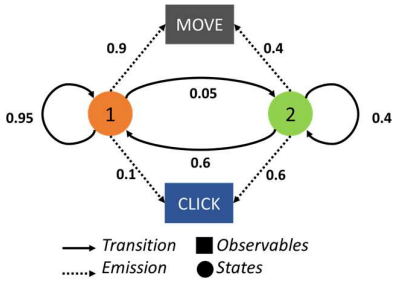
di mana s merupakan jumlah nilai piksel pada daerah citra dan a,b,c,d, adalah nilai pada *integral image* sebagaimana ditunjukkan pada gambar 3.10. Pada gambar 3.10 maka nilai s yang didapatkan adalah 2,3. Nilai tersebut sesuai dengan nilai piksel pada daerah citra asli yaitu $2{,}3 = 0{,}5 + 0{,}8 + 0{,}4 + 0{,}6$.

3.2.5 *Cascade Classification*

Pada proses *cascade classification* ini akan dilakukan pemilahan pada fitur-fitur yang sudah didapat pada tahap sebelumnya. Fitur yang tidak terklasifikasi sebagai objek akan diabaikan. Fitur yang akan digunakan di akhir adalah fitur yang terklasifikasi sebagai objek.

3.2.6 *Tracking-by-Detection*

Proses *tracking-by-detection* dilakukan menggunakan koordinat yang didapatkan dari proses deteksi. *Hidden Markov Model* digunakan untuk menentukan aksi yang dilakukan (*click* atau *move*). Aksi yang dilakukan ditentukan dari *state sequence* yang didapat pada saat deteksi dilakukan. *State* yang terdapat pada sistem terbagi menjadi dua yaitu, tangan dalam kondisi *open* yang dilambangkan dengan 1 dan tangan dalam kondisi *close* yang dilambangkan dengan 2. Adapun hasil observasi dari sistem adalah *move* dan *click*. Diagram *state* pada sistem dapat dilihat pada gambar 3.11. Untuk nilai probabilitas awal transisi dan emisi dari inisialisasi awal dapat dilihat pada Gambar 3.12.



Gambar 3.7 *State diagram* awal HMM sistem

0,95	0,05
0,6	0,4

(a)

0,9	0,1
0,4	0,6

(b)

Gambar 3.12 Matriks probabilitas awal (a) transisi, (b) emisi

Setelah melalui proses latih, nilai dari matrik probabilitas awal transisi dan emisi berubah. Input pada proses latih HMM ini adalah *state sequence* dengan jumlah *state* sebanyak 300. Dari proses latih ini didapatkan nilai akhir dari matriks transisi dan emisi yang dapat dilihat pada gambar 3.13. Untuk *state diagram* akhir dari sistem dapat dilihat pada gambar 3.14.

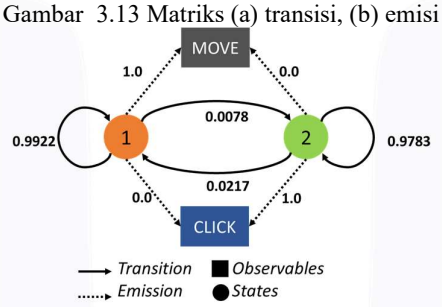
0.9922	0.0078
0.0217	0.9783

(a)

1.0	0
0	1.0

(b)

Gambar 3.13 Matriks (a) transisi, (b) emisi



Gambar 3.14 *State diagram* akhir sistem

3.3 Performansi Sistem

3.3.1 *Precision*

Precision atau presisi merupakan parameter yang nilainya ditentukan dari jarak antara *centroid ground truth* dengan *centroid bounding box actual*. Nilai dari parameter ini dapat dicari menggunakan persamaan 3.1 berikut,

$$P = \sqrt[2]{\left(C_{x_{gt}} - C_{x_{ac}}\right)^2 + \left(C_{y_{gt}} - C_{y_{ac}}\right)^2} \quad (3.1)$$

di mana P merupakan parameter presisi, $C_{x_{gt}}$ merupakan *centroid ground truth* pada kordinat x, $C_{x_{ba}}$ merupakan *centroid bounding box actual* pada kordinat x, $C_{y_{gt}}$ merupakan *centroid ground truth* pada kordinat y, dan $C_{y_{ba}}$ merupakan *centroid bounding box actual* pada kordinat y.

3.3.2 *Intersect Over Union (IoU)*

Intersect Over Union (IoU) merupakan parameter pengukuran yang digunakan untuk mengukur akurasi detektor objek suatu sistem. Perhitungan IoU dilakukan menggunakan persamaan 3.2 berikut,

$$IoU = \frac{B_{gt} \cap B_{ac}}{B_{gt} \cup B_{ac}}, \quad (3.2)$$

di mana IoU merupakan nilai dari *Intersect Over Union*, B_{gt} merupakan area dari *bounding box ground truth*, dan B_{ac} merupakan area dari *bounding box actual*.

3.3.3 *Akurasi*

Akurasi merupakan parameter keberhasilan dari suatu sistem dalam mendeteksi objek. Akurasi dapat dihitung menggunakan persamaan 3.3.

$$Akurasi = \frac{\sum data\ benar}{\sum seluruh\ data} \times 100\% \quad (3.3)$$

dimana data benar merupakan data yang memiliki nilai IoU lebih dari 0,5.

4 Hasil dan Analisis

4.1 Skenario Pengujian Sistem

Pada penelitian tugas akhir ini pengujian sistem dilakukan menggunakan tiga video yang menampilkan tangan terbuka dengan tangan dalam kondisi *open, close* dan kondisi tangan *open* maupun *close* menuju depan dengan latar berwarna hitam. Pada pengujian ini terdapat empat skenario analisis pengujian yang bertujuan untuk mencari konfigurasi terbaik. Berikut ini penjelasan mengenai masing-masing skenario:

- Skenario 1: Pengujian terhadap parameter presisi

  Skenario ini bertujuan untuk mencari jarak antara *centroid ground truth* dengan *centroid bounding box actual*. Parameter presisi digunakan untuk mengetahui ketepatan deteksi pada sistem. Pengujian dilakukan terhadap data uji berupa video kondisi *open,* dan *close.* Nilai presisi dianggap semakin baik apabila mendekati nilai 0.
- Skenario 2: Pengujian terhadap parameter *Intersection Over Union* (IoU)

  Skenario ini bertujuan untuk mengetahui keberhasilan sistem dalam mendeteksi objek. IoU merupakan perpotongan antara area *ground truth bounding box* dengan area *bounding box actual*. Pengujian dilakukan terhadap data uji berupa video kondisi *open,* dan *close.* Nilai IoU dianggap semakin baik apabila mendekati 1.
- Skenario 3: Pengujian terhadap parameter akurasi deteksi

  Parameter akurasi menentukan seberapa baik sistem dalam mendeteksi objek. Pada skenario ini, sistem akan diuji terhadap data uji berupa video kondisi *open,* dan *close.*
- Skenario 4: Pengujian terhadap parameter akurasi prediksi

  Parameter akurasi menentukan seberapa baik sistem dalam memprediksi objek dalam sebuah *sequence* atau urutan. Pada skenario ini, sistem akan diuji menggunakan data uji berupa video dengan tangan dalam kondisi *open* dan *close*.

4.2 Data Hasil Pengujian Sistem

Pada penelitian tugas akhir ini dilakukan 6 percobaan dengan konfigurasi *Merge Threshold* 75, 100, dan 125 untuk kondisi *close* dan 225, 250, 275 untuk kondisi *open*. Pada seluruh percobaan menggunakan *scale factor* 1,5. Untuk *minimum size* pada kondisi *open* maupun *close* adalah 200×250 dan 200×200. Sedangkan untuk *maximum size* adalah 500×400 untuk *open* dan 300×300 untuk *close*. Diuji dengan inisialisasi awal *emission probability* dan *transition probability* pada persamaan 4.1 dan 4.2.

$$Trans_{prob} = \begin{bmatrix} 0.95 & 0.05 \\ 0.60 & 0.40 \end{bmatrix}, \quad (4.1)$$

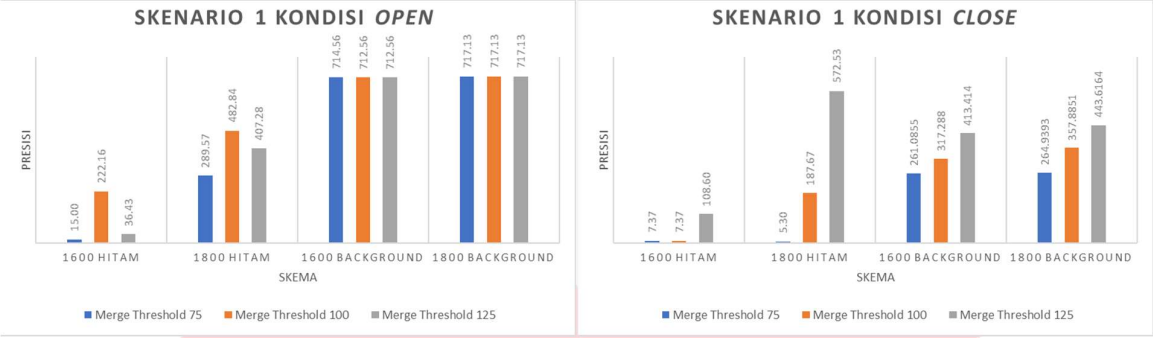
$$Emis_{prob} = \begin{bmatrix} 0.9 & 0.1 \\ 0.4 & 0.6 \end{bmatrix}, \quad (4.2)$$

di mana $Trans_{prob}$ merupakan matriks transisi dari *Hidden Markov Model* dan $Emis_{prob}$ merupakan matriks emisi dari *Hidden Markov Model*. Nilai dari $Trans_{prob}$ dan $Emis_{prob}$ akan berubah pada saat proses.

4.2.1 *Skenario 1: Pengujian terhadap parameter presisi*

Pada parameter presisi nilai mendekati 0 dianggap semakin baik. Hasil pengujian dari skenario 1 saat kondisi *open* dapat dilihat pada gambar 4.1. Dari gambar 4.1 dapat dilihat bahwa data latih dengan data positif sebanyak 1600 dengan *merge threshold* sebesar 225 mendapatkan hasil terbaik untuk tangan dalam kondisi *open*. Hasil pengujian dari skenario 1 saat kondisi *close* dapat dilihat pada gambar 4.2. Dari hasil tersebut dapat dilihat bahwa pengujian menggunakan data latih dengan jumlah data positif 1800 dan *merge threshold* sebesar 75 mendapat hasil terbaik untuk kondisi tangan *close*. Dari gambar 4.1 dan 4.2 terlihat bahwa semakin besar nilai dari *merge threshold* yang digunakan maka nilai presisi yang dihasilkan akan semakin besar. Dari gambar 4.1 dapat dilihat bahwa pada penggunaan data uji dengan background tidak hitam,nilai presisi sangat besar yang menandakan bahwa jarak *centroid bounding box* hasil prediksi sangat jauh dari *centroid bounding box* yang sesungguhnya.

Maka dari itu, nilai terbaik pada parameter presisi adalah nilai yang mendekati 0 karena hal tersebut menandakan jarak *centroid* antara *bounding box* hasil prediksi tidak begitu jauh dari *bounding box* sesungguhnya.

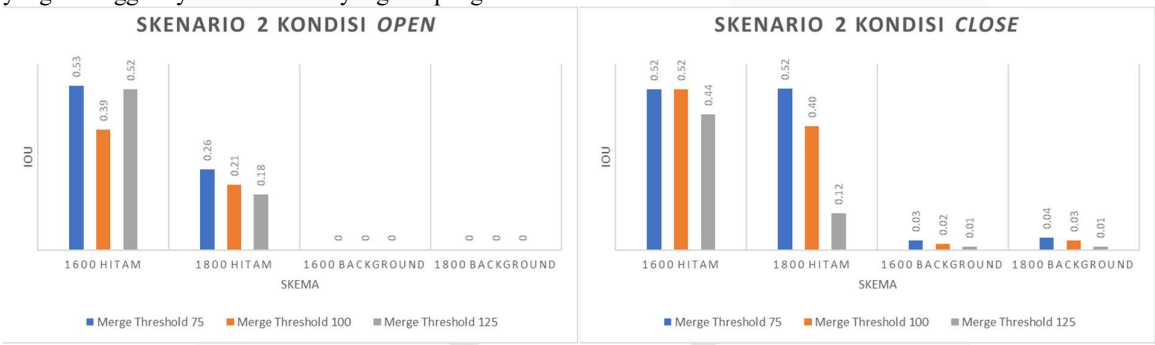


(a) (b)

Gambar 4.1 Hasil skenario 1 pada kondisi (a) *open*, (b) *close*

4.2.2 *Skenario 2: Pengujian terhadap parameter IoU*

Pada parameter IoU nilai mendekati 1 dianggap semakin baik. Data akan dianggap benar apabila nilai IoU lebih besar dari 0,5. Dari hasil pada gambar 4.3 data latih dengan data positif sebanyak 1600 dengan *merge threshold* sebesar 225 mendapatkan hasil terbaik untuk tangan dalam kondisi *open*. Dari hasil pada gambar 4.4 dapat dilihat bahwa pengujian menggunakan data latih dengan jumlah data positif 1800 dan *merge threshold* sebesar 75 mendapat hasil terbaik untuk kondisi tangan *close*. Pada IoU, semakin nilai IoU mendekati 1 maka semakin daerah bounding box hasil prediksi yang tumpang tindih dengan daerah bounding box sesungguhnya sehingga hasil deteksi semakin akurat. Dari gambar 4.3 dan 4.2 dapat dilihat nilai dari IoU pada setiap pengujian. pada pengujian yang mendapatkan nilai presisi besar pada skenario sebelumnya mendapatkan nilai IoU yang rendah dikarenakan semakin besar jarak antara *centroid bounding box* hasil prediksi dengan *centroid bounding box* yang sesungguhnya maka daerah yang tumpang tindih akan semakin kecil.

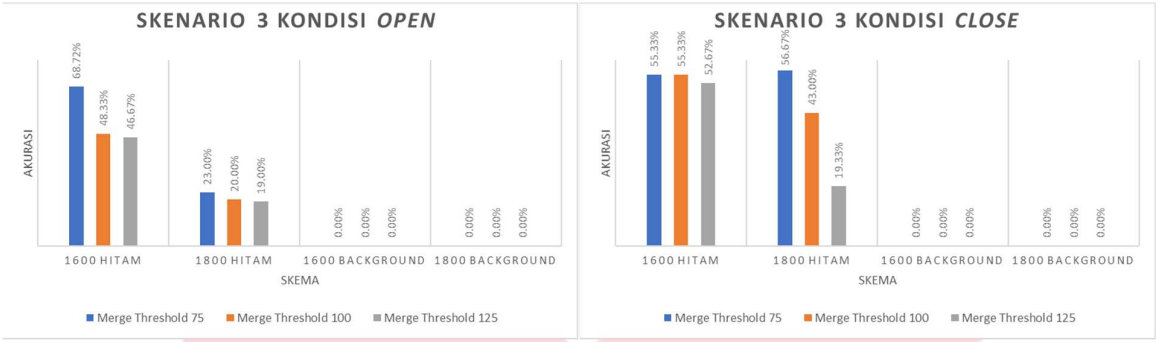


(a) (b)

Gambar 4.2 Hasil skenario 2 pada kondisi (a) *open*, (b) *close*

4.2.3 *Skenario 3: Pengujian terhadap parameter akurasi deteksi*

Pada parameter akurasi merupakan perbandingan antara data benar dan dengan keseluruhan data. Semakin mendekati 100% maka dianggap semakin baik. Pada skenario 3 ini, nilai akurasi dipengaruhi oleh nilai dari IoU. Data hasil pengujian akan dianggap benar apabila nilai IoU $\geq$ 0,5. Semakin banyak data dengan nilai IoU $\geq$ 0,5 maka akan semakin besar nilai akurasi yang didapat. Dari hasil pada gambar 4.5 dapat dilihat bahwa pengujian menggunakan data latih dengan data positif sebanyak 1600 dengan *merge threshold* sebesar 225 mendapatkan hasil terbaik untuk tangan dalam kondisi *open*. Pada gambar 4.6 dapat dilihat bahwa pengujian menggunakan data latih dengan jumlah data positif 1800 dan *merge threshold* sebesar 75 mendapat hasil terbaik untuk kondisi tangan *close*.

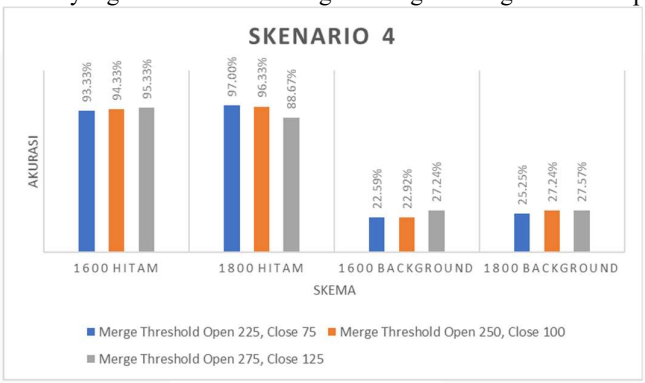


(a) (b)

Gambar 4.3 Hasil skenario 3 pada kondisi (a) *open*, (b) *close*

4.2.4 *Skenario 4: Pengujian terhadap parameter akurasi prediksi*

Hasil dari pengujian skenario 4 dapat dilihat pada gambar 4.7. Pada skenario ini nilai akurasi hanya ditentukan oleh hasil dari deteksi. Semua data dengan hasil deteksi yang sesuai akan dianggap benar walaupun nilai IoU yang didapat kurang dari 0,5. Pada gambar 4.7 dapat dilihat bahwa pengujian dengan data positif sebanyak 1600 mendapat akurasi terbaik pada pengaturan *Merge Threshold Open* 275 *Close* 125 dan pada data positif sebanyak 1800 nilai akurasi tertinggi pada pengaturan *Merge Threshold Open* 250 *Close* 100 dan *Merge Threshold Open* 275 *Close* 125. Nilai akurasi tertinggi yang didapat pada scenario 4 ini adalah 96,2085%. Dari gambar 4.7 dapat dilihat bahwa kondisi *background* dapat mempengaruhi hasil deteksi. Pengujian dengan *background* polos berwarna hitam mendapat akurasi yang lebih baik dibandingkan dengan background tidak polos.



Gambar 4.4 Hasil skenario 4

Analisis Hasil Pengujian

Dari empat skenario pengujian yang sudah dilakukan, didapatkan hasil analisis dari setiap parameter kinerja. Nilai presisi terbaik pada kondisi tangan dalam keadaan *open* yang didapat adalah 14,9993 saat menggunakan data latih dengan 1600 data positif dengan *merge threshold* 225. Nilai IoU terbaik pada kondisi tangan dalam keadaan *open* yang didapat adalah 0,5288 saat menggunakan data latih dengan 1600 data positif dengan *merge threshold* 225. Nilai akurasi deteksi terbaik pada kondisi tangan dalam keadaan *open* yang didapat adalah 68,72% saat menggunakan data latih dengan 1600 data positif dengan *merge threshold* 225. Nilai presisi terbaik pada kondisi tangan dalam keadaan *close* yang didapat adalah 5,3015 saat menggunakan data latih dengan 1800 data positif dengan *merge threshold* 75. Nilai IoU terbaik pada kondisi tangan dalam keadaan *close* yang didapat adalah 0,5221 saat menggunakan data latih dengan 1800 data positif dengan *merge threshold* 75. Nilai akurasi deteksi terbaik pada kondisi tangan dalam keadaan *close* yang didapat adalah 56,75% saat menggunakan data latih dengan 1800 data positif dengan *merge threshold* 75. Pada skenario 4 hasil akurasi terbaik adalah 96,2085%.

Dari hasil analisis yang didapat, dapat dilihat bahwa nilai yang didapat pada skenario 1,2, dan 4 tidak stabil. Nilai *merge threshold* yang kecil tidak menjamin kinerja yang lebih baik, begitu pula sebaliknya. Dikarenakan nilai dari parameter kinerja yang tidak stabil, maka untuk menemukan konfigurasi *merge threshold* terbaik dibutuhkan *trial and error*. Dari seluruh scenario yang sudah dilakukan juga dapat dilihat bahwa kondisi background memberikan pengaruh yang besar. Pengujian dengan *background* polos berwarna hitam mendapat akurasi yang lebih baik dibandingkan dengan background tidak polos.

5 Kesimpulan

Dari seluruh pengujian yang telah dilakukan terhadap sistem dan melakukan analisa terhadap hasil yang didapat, dapat diambil kesimpulan sebagai berikut:

1. HMM berhasil diimplementasikan untuk *interactive hologram* dengan menggunakan *haar-like features* sebagai ekstraksi ciri.
2. Dari hasil pengujian pada skenario 1, 2, 3 dapat disimpulkan bahwa data latih dengan data positif sebanyak 1800 dengan *merge threshold* 75 memiliki kinerja terbaik pada kondisi tangan *close* dengan nilai presisi 5,3015, IoU sebesar 0,5221, dan akurasi sebesar 56,67%.
3. Dalam kondisi tangan *open* data latih dengan data positif sebanyak 1600 dengan *merge threshold* 225 memiliki kinerja terbaik dengan nilai presisi 14,9993, IoU sebesar 0,5288, dan akurasi sebesar 68,72%.
4. Dari skenario 4 didapat akurasi sistem terbaik sebesar 96.33% pada saat menggunakan data latih 1800 dengan *merge threshold open* 250, *merge threshold close* 100.

6 Daftar Pustaka

[1] The Interaction Design Foundation, “Human Computer Interaction - brief intro,” dalam *The Encyclopedia of Human-Computer Interaction, 2nd Ed.*, The Interaction Design Foundation, 2013.

[2] M. K. Ahuja dan A. Singh, “Static Vision Based Hand Gesture Recognition,” dalam *IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE)*, 2015.

[3] S. Song, D. Yan dan Y. Xie, “Design of control system based on hand gesture,” dalam *IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018.

[4] S. Hussain, R. Saxena, X. Han, J. A. Khan dan H. Shin, “Hand Gesture Recognition Using Deep Learning,” dalam *International SoC Design Conference (ISOCC)*, 2017.

[5] D. M. Plasencia, E. Joyce and S. Subramanian, "MisTable: reach-through personal screens for tabletops," in *CHI '14 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, Toronto, 2014.

[6] S. G. Paredes and N. R. Vázquez, "My Teacher is a Hologram: Measuring innovative STEM learning experiences," in *2019 IEEE Integrated STEM Education Conference (ISEC)*, Princeton, NJ, USA, USA, 2019.

[7] Y. T. Li and J. P. Wachs, "Recognizing hand gestures using the weighted elastic graph matching (WEGM) method," *Image and Vision Computing,* vol. 31, no. 9, pp. 649-657, 2013.

[8] H. G. Doan, H. Vu and T. H. Tran, "Recognition of hand gestures from cyclic hand movements using spatial-temporal features," in *the Sixth International Symposium on Information and Communication Technology*, Hue City, Viet Nam, 2015.

[9] D. S. Chen and Z. K. Liu, "GENERALIZED HAAR-LIKE FEATURES FOR FAST FACE DETECTION," in *2007 International Conference on Machine Learning and Cybernetics*, Hong Kong, China, 2007.

[10] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision,* vol. 57, no. 2, pp. 137-154, 2004.

[11] S. A. Wibowo, "Analisis Kinerja IS-127 Enhanced Variable Rate Code Rate Determination Algorithm Menggunakan Metode Klasifikasi Hidden Markov Model," in *Telkom University*, Bandung, 2012.

[12] A. Kadir dan A. Susanto, “Pengolahan Citra Berwarna,” dalam *Teori dan Aplikasi Pengolahan Citra*, Yogyakarta, Penerbit ANDI, 2013, pp. 285-334.

[13] U. Buyukhasin, "Vision-Based Sensor Technologies – Webcam: A Multifunction Sensor," in *Comprehensive Materials Processing*, Istanbul, Elsevier, 2014, pp. 375-392.