PERANCANGAN SISTEM PENGENALAN WAJAH DAN MASKER MENGGUNAKAN *RASPBERRY PI*

DESIGN OF FACE AND MASK RECOGNITION SYSTEM USING RASPBERRY PI

Rahmat Syamsudaris[1], Muhammad Ary Murti[2], Willy Anugrah Cahyadi[3]

[1,2,3] Universitas Telkom, Bandung
rahmatsyam@student.telkomuniversity.ac.id[1], **arymurti@telkomuniversity.ac.id[2]**,
waczze@telkomuniversity.ac.id[3]

Abstrak

Pada penelitian tugas akhir ini akan dibuat sistem yang memanfaatkan teknologi *Artificial Intelligence* (AI) yang berbasis *Raspberry Pi* 4. Dengan memanfaatkan bahasa pemrograman *Python* dan beberapa *library* yang ada seperti *OpenCV* dan *face_recognition*, sistem ini akan mengenali wajah seseorang dan dia menggunakan masker atau tidak yang sebelumnya *dataset* tersebut telah disimpan di *database*. Kemudian, hasil dari proses tersebut akan di tampilkan pada sebuah layar monitor. Terdapat 5 *dataset* wajah yang akan dilatih oleh program yang telah dibuat dan 3 dari 5 *dataset* yang telah dilatih akan diuji. Pengujian sistem yang dilakukan adalah pengujian berdasarkan jarak yang berubah-ubah dengan posisi wajah lurus menghadap kamera, yaitu ketika memakai masker dan tidak memakai masker. Kemudian dilakukan juga pengujian berdasarkan posisi wajah dengan jarak wajah ke kamera tidak berubah, yaitu ketika memakai masker dan tidak memakai masker. Setelah melakukan beberapa skenario pengujian di atas, dengan kondisi sedang tidak menggunakan masker didapatkan tingkat akurasi terbaik sebesar 100% dan ketika pengujian dengan kondisi sedang menggunakan masker tingkat akurasi terbaik yang didapatkan juga sebesar 100%.

Kata kunci : ***Face Recognition, Python, Raspberry Pi* 4**

Abstract

In this final project, a system that utilizestechnology Artificial Intelligence (AI) implementation is proposed on Raspberry Pi 4. Taking advantage of Python programming language and some existing libraries, such as OpenCV and face_recognition. The proposed system recognizes a person's face and whether he uses a mask or not based on the trained model from previously stored dataset in the database. The results can then be displayed on a monitor screen. There are 5 faces datasets that are trained for the created program and 3 of them are validated. System testing was carried on varying distances with a straight face position facing the camera, namely when the person was wearing a mask and when he was not wearing it. Further, a test was also carried out based on the position of the face with a fixed distance from the face to the camera, for both case when he was wearing a mask and not wearing it. After all aforementioned tests, the proposed system achieved, the best accuracy rate of 100% for both cases, i.e., when the user is wearing a mask and when he is not wearing it.

Keywords: ***Face Recognition, Python, Raspberry Pi* 4**

1. Pendahuluan

Sejak beberapa bulan yang lalu di mana penyakit *Coronavirus Disease*-2019 (COVID-19) melanda Indonesia yang membuat Pemerintah Indonesia menerapkan Pembatasan Sosial Berskala Besar (PSBB). Kemudian dibulan Juni kemarin Pemerintah Indonesia akhirnya mencoba menerapkan *New Normal* yaitu tatanan baru untuk beradaptasi dengan COVID-19. Beberapa aturan yang harus dipatuhi di antaranya yaitu memakai masker ketika keluar rumah dan menjaga jarak aman. Pemerintah Indonesia pun melakukan penelitian untuk penanganan COVID-19 melalui Badan Riset dan Inovasi (BRIN) Indonesia yang salah satunya dengan memanfaatkan *Artificial Intelligence* (AI). Salah satu teknologi dari AI yaitu pengenalan wajah (*face recognition*) dan ini telah dimanfaatkan dalam banyak hal, seperti pada proses otentifikasi ataupun identifikasi.

Terdapat beberapa penelitian yang serupa yaitu untuk mencoba mengenali wajah dengan metode yang berbeda seperti pada penelitian yang dilakukan Novita Anik Iswanti dengan judul Implementasi Algoritma *Viola-Jones* Untuk Deteksi Wajah Tampak Depan dapat mendeteksi wajah dengan baik di mana tingkat akurasi sebesar 80,55% [1]. Penelitian yang serupa juga di lakukan oleh Wagh Priyanka, dkk., dengan judul *Attendance System Based on Face Recognition using Eigen face and PCA Algorithms* yang menghasilkan tingkat kesuksesan sebesar 93,7% [2].

Maka dari itu terpikirkanlah untuk membuat sebuah sistem yang dapat mengenali wajah seseorang dan tetap mematuhi protokol kesehatan yaitu dengan memakai masker. Serta yang diharapkan dari penelitian ini dapat menghasilkan sebuah sistem *face recognition* yang akurat dalam membedakan wajah setiap orang yang telah terdapat di dalam *dataset*.

2. Dasar Teori

2.1 *Raspberry Pi*

Raspberry Pi merupakan papan elektronik yang hampir mirip dengan kartu kredit tapi memiliki kemampuan serta fungsi yang mirip dengan sebuah komputer. Jika *Raspberry Pi* ini dihubungkan dengan monitor, *keyboard*, *mouse*, dan jaringan komputer maka dapat digunakan untuk melakukan pekerjaan yang mirip dengan komputer seperti melayani pemakaian internet dan bahkan dapat menjadikannya sebagai *web server*[3]. Karena *Raspberry Pi* biasanya disebut sebagai mini komputer maka dari itulah alat tersebut dapat melakukan pekerjaan yang mirip dengan komputer.

2.2 *Face Recognition*

Setiap wajah manusia memiliki keanekaragaman wajah yang berbeda-beda, orang kembar pun masih memiliki perbedaan wajah. Oleh karena itu wajah bisa di ibaratkan suatu ciri khusus untuk mengenali dan mengingat wajah manusia satu sama lain[4]. *Face recognition* merupakan prinsip dasar pengenalan wajah untuk mengutip informasi unik dari wajah, yang kemudian di *encode* dan dibandingkan dengan hasil dengan hasil *decode* yang sebelumnya dilakukan. Prinsip kerja dari sistem *face recognition* adalah kamera atau *webcam* akan mengambil sebuah gambar, kemudian mendeteksi ada atau tidaknya wajah pada gambar tersebut. Ketika kamera atau *webcam* berhasil mendeteksi sebuah wajah pada gambar tersebut lalu sistem akan membandingkan gambar wajah tersebut dengan gambar wajah yang telah terdaftar atau sudah ada sebelumnya pada sistem. Cocok atau tidaknya hasil dari perbandingan tersebut akan digunakan sesuai implementasi dari sistem.

Pada proses deteksi wajah digunakan metode *Histogram of Oriented Gradients* (HOG), HOG merupakan salah satu metode ekstraksi ciri yang digunakan dalam *image processing* untuk mendeteksi suatu objek. HOG berasal dari sebuah asumsi yang menyatakan bahwa suatu objek dapat direpresentasikan dengan baik berdasarkan bentuk. Untuk memperoleh informasi pembeda maka gambar akan dibagi menjadi *cell* dan setiap *cell* akan dihitung sebagai HOG. Setiap piksel dalam *cell* berkontribusi pada saat dilakukan voting bobot untuk membangun sebuah histogram yang berorientasi pada nilai-nilai gradien yang dihitung[5].

2.3 *OpenCV*

OpenCV (*Open Source Computer Vision*) ini adalah sebuah *library* pemrograman yang fungsi utamanya ditujukan pada *computer vision* (visi komputer) secara *realtime* yang sangat berguna untuk *image processing* atau pengolahan citra. *OpenCV* dilengkapi oleh beberapa fitur yang salah satunya dapat mencakup area yang luas seperti, sistem pengenalan wajah, pengenalan isyarat, identifikasi segmentasi, pengenalan objek, serta pelacak gerak, 2D/3D, estimasi egomotion. *OpenCV* biasanya menggunakan bahasa pemrograman yang fleksibel yaitu *Python*[6]. Pada kasus ini, *OpenCV* digunakan untuk sistem pengenalan wajah.

2.4 *Deep Metric Learning*

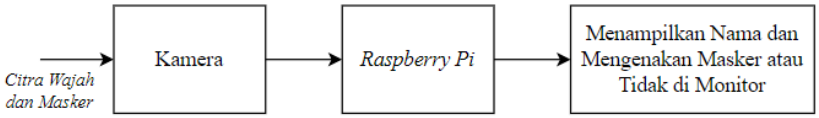
Pada tugas akhir ini juga dilakukan penginstalan *dlib* [7] dan modul *face_recognition*. Davis King merupakan *author* dari *dlib* dan Adam Geitgey merupakan *author* dari modul *face_recognition*. Ide untuk mengurangi *raw* data yang kompleks dari gambar menjadi *computer generated numbers* mulai menjadi *trend* di *machine learning*. Pendekatan yang digunakan pada modul *face recognition* ditemukan pada tahun 2015 oleh para peneliti dari *Google* yang dinamakan dengan *FaceNet* [8]. Adapun *network architecture* yang digunakan pada modul *face recognition* prinsipnya berdasarkan *ResNet-34* dari *paper Deep Residual Learning for Image Recognition* [9] namun dengan jumlah *layer* yang lebih sedikit dan jumlah *filter* yang dikurangi hingga separuhnya. Kemudian dinamakan dengan *Deep Metric Learning*. Hal ini juga terinspirasi dari *OpenFace* [10]. Beberapa ide dari proyek David juga diperoleh dari *Deep Face Recognition* dari *Visual Geometry Group* [11].

Pada *dlib facial recognition network* inilah dikeluarkan *output* 128-*d vector* untuk kuantifikasi wajah. *Training* ini dilakukan dengan menggunakan *triplet training step*. Misalkan terdapat tiga buah *images* wajah. Dua *images* (*image* #1 dan *image* #2) merupakan orang yang sama. Sedangkan *image* yang lainnya merupakan wajah orang asing. *Neural Network* akan menguantifikasi wajah dan melakukan konstruksi 128-*d vectors* atau 128 *measurements* untuk masing-masing *image* wajah. Langkah selanjutnya adalah dilakukan *tweak* terhadap *weights* dari *Neural network* sehingga *128-d vectors* dari *image* #1 dan *image* #2 nilainya akan mendekati satu sama lain, tetapi nilai pada *image* #3 menjauh dari *128-d vectors*. Jadi, Tugas Akhir ini menggunakan metode *deep metric learning* dengan *triplet training step*. Biasanya *training* sebuah *network* bertujuan untuk menerima masukan dalam bentuk sebuah *image* lalu keluarannya berupa klasifikasi atau *label* dari *image* tersebut. Namun *deep metric learning* berbeda sama sekali. *Deep metric learning* mencoba mengeluarkan *output* berupa *real-value feature*

vector sebanyak 128 bilangan *real* (mencakup bilangan positif, negatif, dan pecahan). *Output* nya bukan berupa *label* atau bahkan koordinat/*bounding box* dari objek pada *image*.

3. Perancangan Sistem

3.1. Diagram Blok Sistem

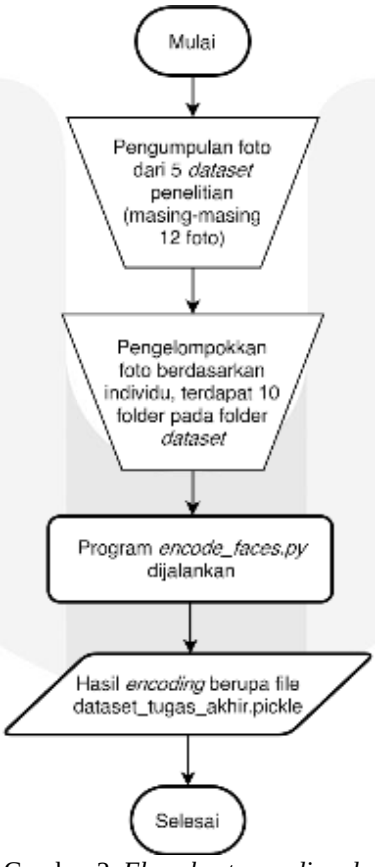


Gambar 1. Diagram Blok Sistem.

Gambar 1 merupakan diagram blok sistem untuk yang menggambarkan cara kerja sistem secara umum yang menggunakan metode *image processing* atau pengolahan citra. Pertama kamera akan melakukan proses pengolahan citra untuk pengambilan wajah yang terdeteksi, setelah itu akan mencoba mengidentifikasi atau mengenali wajah tersebut dan mengirimkan hasil data ke *Raspberry Pi*. *Raspberry Pi* akan memproses datanya untuk melakukan pengecekan apakah wajah yang ada di depan kamera sudah ada dalam *database*, jika wajah dikenali maka akan ditampilkan nama orang yang terdeteksi serta menggunakan masker atau tidak pada sebuah monitor.

3.2 Desain Perangkat Lunak

Pada tugas akhir ini terbagi menjadi dua bagian. Bagian pertama sebagai bagian proses persiapan merupakan proses *encode* gambar-gambar wajah dan masker yang terdapat pada *dataset*. Proses cukup membutuhkan sumber daya komputasi yang mumpuni sehingga gambar-gambar yang dikumpulkan cukup terbatas. Proses bagian pertama dapat digambarkan dalam *flowchart* seperti pada Gambar 2 berikut ini.

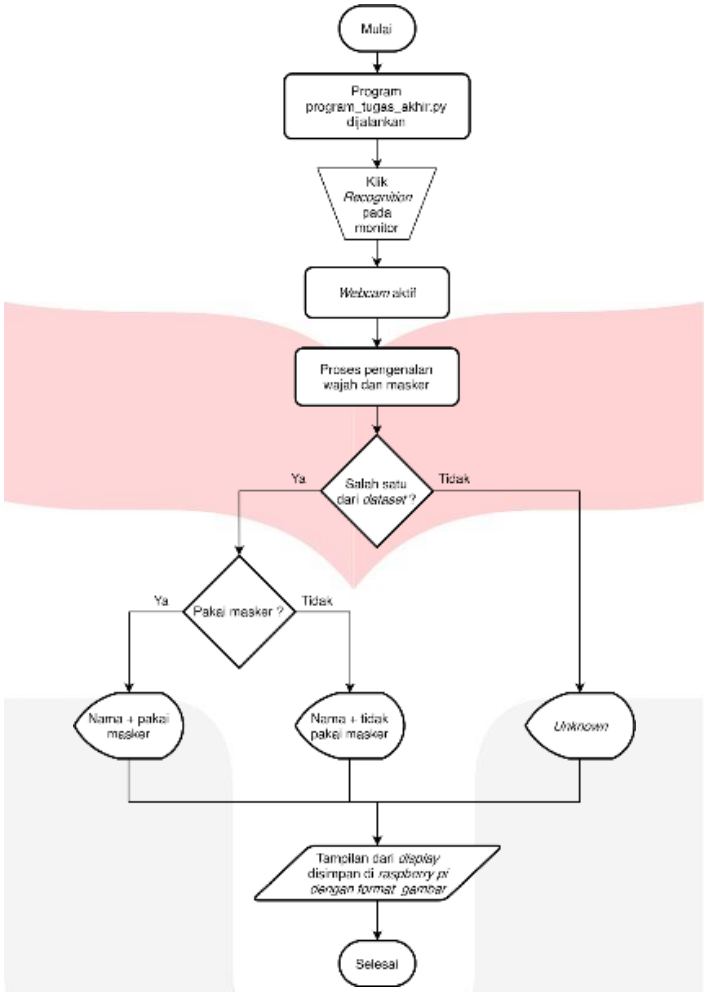


Gambar 2. *Flowchart encoding dataset.*

Pada Gambar 2 proses dimulai dengan mengumpulkan gambar-gambar dari 5 orang yang dijadikan *dataset*. Setiap *dataset* memiliki 12 gambar ketika menggunakan masker atau tidak menggunakan masker. Kemudian dikelompokkan berdasarkan individu, sehingga diperoleh 10 folder dan diberi nama sesuai nama individu serta tambahan menggunakan masker atau tidak menggunakan masker. 10 folder tersebut ditempatkan dalam satu folder yang diberi nama *dataset*, selanjutnya program *encode_faces.py* dijalankan. Program tersebut akan mendeteksi wajah pada gambar dalam folder *dataset*, proses mendeteksi wajah menggunakan metode *Histogram*

of Gradient (HOG). Setelah itu, hasilnya akan disimpan dalam suatu file yang diberi nama *dataset_tugas_akhir.pickle*.

Bagian kedua yaitu proses pengenalan wajah dan masker secara *realtime*, adapun sistem ini memiliki *flowchart* seperti pada Gambar 3 berikut ini.



Gambar 3. *Flowchart* sistem pengenalan wajah dan masker secara *realtime.*

Pada Gambar 3 proses dimulai ketika menjalankan program *program_tugas_akhir.py* dan akan memuat file *dataset_tugas_akhir.pickle*, file ini dihasilkan dari proses pada Gambar III-5. Setelah itu mengeklik tombol *Recognition* pada monitor dan kamera/*webcam* pun akan aktif. Lalu ketika terdapat wajah di depan kamera, *Raspberry Pi* akan melakukan proses pengenalan wajah dan apakah menggunakan masker atau tidak menggunakan masker dengan bantuan *library face_recognition*, *dlib*, dan *OpenCV*. Setelah proses tersebut hasilnya akan ditampilkan pada monitor, terdapat tiga jenis hasil dari proses tersebut, pertama dikenali dan menggunakan masker, kedua dikenali dan tidak menggunakan masker, kemudian yang ketiga tidak dikenali atau *unknown*. Selanjutnya hasil dari proses pengenalan wajah dan masker tersebut akan disimpan pada folder *hasil* yang telah dibuat sebelumnya dalam *Raspberry Pi.*

4. Pembahasan

4.1 Skenario Pengujian Sistem

Untuk sub bab ini akan dijelaskan tentang ketentuan skenario pengujian sistem guna mengetahui kemampuan dari sistem pengenalan wajah dan masker yang telah dirancang. Pada sistem ini menggunakan *library face_recognition*, *dlib*, dan *OpenCV* dengan menggunakan bahasa pemrograman *python*. Adapun jumlah *dataset* wajah yang dilatih sebanyak 5 *dataset* wajah yaitu Adun, Asri, Hanif, Kholis, dan Rahmat. Sedangkan *dataset* yang diuji hanya 3 *dataset* yaitu Adun, Hanif, dan Rahmat. Jenis-jenis pengujian yang dilakukan adalah pengujian berdasarkan jarak yang berubah-ubah dengan posisi wajah lurus menghadap kamera, yaitu ketika memakai masker dan tidak memakai masker. Kemudian dilakukan pengujian berdasarkan posisi wajah dengan jarak wajah ke kamera tidak berubah, yaitu ketika memakai masker dan tidak memakai masker. Setelah mendapatkan hasil dari pengujian di atas, penulis dapat menghitung tingkat akurasi dari sistem dengan menggunakan persamaan matematis sebagai berikut:

$$Akurasi = \frac{Total\ Dataset\ yang\ Berhasil\ Dikenali}{Total\ Pengujian\ yang\ Dilakukan} \times 100\% \quad (1)$$

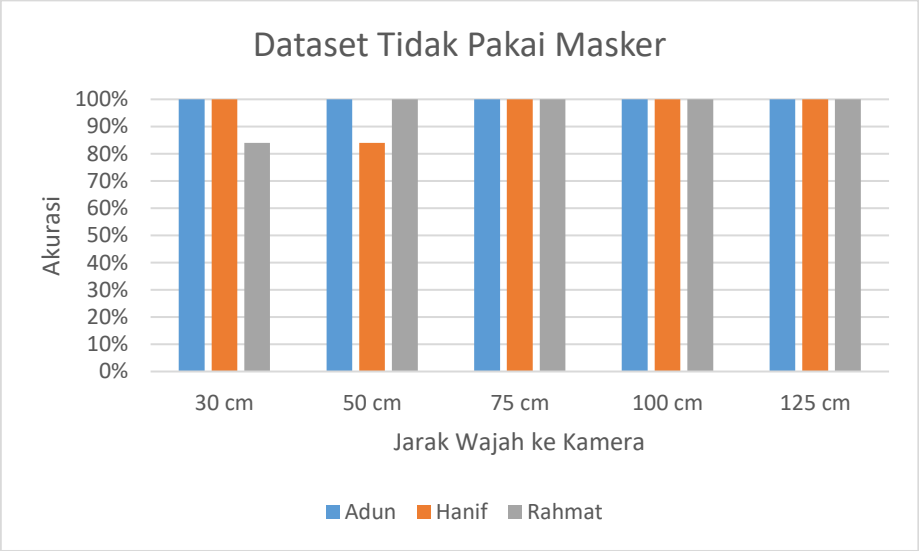
4.2 Pengujian Berdasarkan Jarak dengan Posisi Wajah Lurus Menghadap ke Kamera Ketika Tidak Memakai Masker

Untuk setiap *dataset* wajah dilakukan percobaan pengenalan wajah ketika tidak memakai masker dengan jarak 30 cm, 50 cm, 75 cm, 100 cm, dan 125 cm dengan masing-masing jarak dilakukan percobaan sebanyak 25 kali percobaan. Hasil dari percobaan tersebut dilihat pada Tabel 1

Tabel 1. Hasil Pengujian Setiap *Dataset* Ketika Tidak Pakai Masker

Jarak Dataset Tidak Pakai Masker	Adun		Hanif		Rahmat	
	Berhasil	Gagal	Berhasil	Gagal	Berhasil	Gagal
30 cm	25	0	25	0	21	4
50 cm	25	0	21	4	25	0
75 cm	25	0	25	0	25	0
100 cm	25	0	25	0	25	0
125 cm	25	0	25	0	25	0

Berdasarkan Tabel 1 dapat diketahui persentase tingkat akurasi sistem untuk setiap *dataset* dengan menggunakan persamaan (1) dapat dilihat pada Gambar 4



Gambar 4. Tingkat Akurasi Setiap Dataset Ketika Tidak Pakai Masker

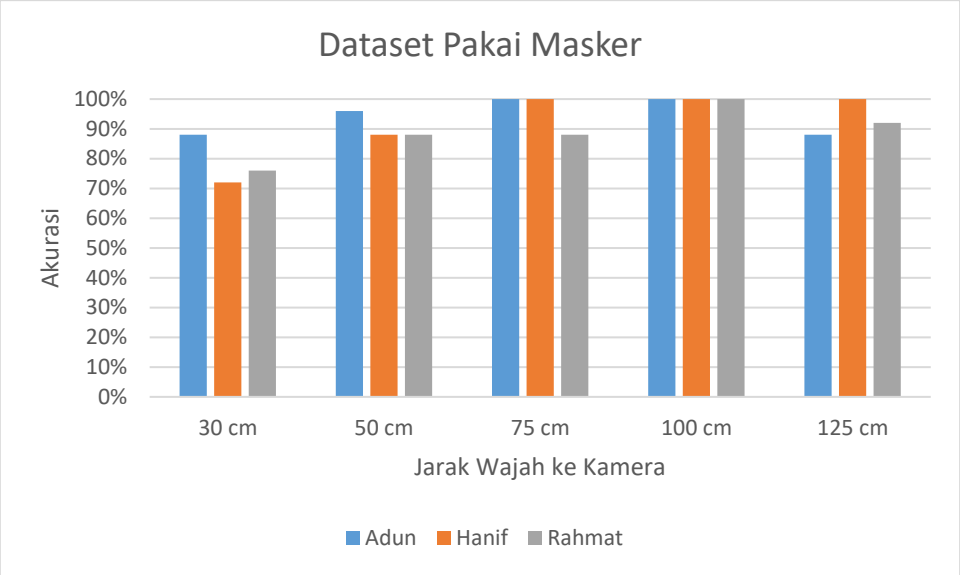
4.3 Pengujian Berdasarkan Jarak dengan Posisi Wajah Lurus Menghadap ke Kamera Ketika Memakai Masker

Untuk setiap *dataset* wajah dilakukan percobaan pengenalan wajah ketika memakai masker dengan jarak 30 cm, 50 cm, 75 cm, 100 cm, dan 125 cm dengan masing-masing jarak dilakukan percobaan sebanyak 25 kali percobaan. Hasil dari percobaan tersebut dilihat pada Tabel 2

Tabel 2. Hasil Pengujian Setiap *Dataset* Ketika Pakai Masker

Jarak Dataset Pakai Masker	Adun		Hanif		Rahmat	
	Berhasil	Gagal	Berhasil	Gagal	Berhasil	Gagal
30 cm	22	3	18	7	19	6
50 cm	24	1	22	3	22	3
75 cm	25	0	25	0	22	3
100 cm	25	0	25	0	25	0
125 cm	22	3	25	0	23	2

Berdasarkan Tabel 2 dapat diketahui persentase tingkat akurasi sistem untuk setiap *dataset* dengan menggunakan persamaan (1) dapat dilihat pada Gambar 5



Gambar 5. Tingkat Akurasi Setiap Dataset Ketika Pakai Masker

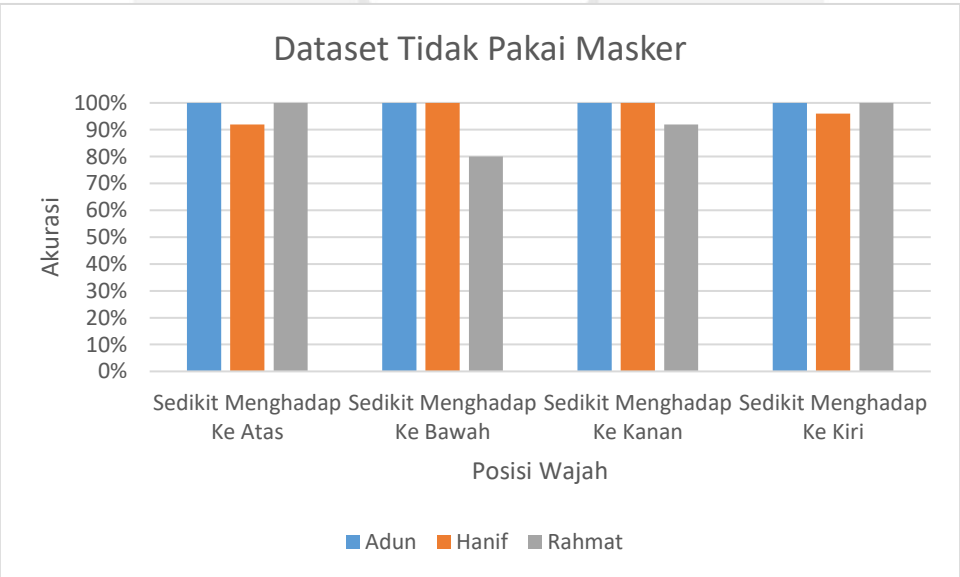
4.4 Pengujian Berdasarkan Posisi Wajah dengan Jarak Wajah ke Kamera Tidak Berubah Ketika Tidak Memakai Masker

Untuk setiap *dataset* wajah dilakukan percobaan pengenalan wajah ketika tidak memakai masker dengan posisi wajah sedikit menghadap ke atas, ke bawah, ke kanan, dan ke kiri tetapi jarak wajah ke kamera tidak berubah, percobaan dilakukan sebanyak 25 kali percobaan. Hasil dari percobaan tersebut dilihat pada Tabel 3

Tabel 3. Hasil Pengujian Setiap *Dataset* Ketika Tidak Pakai Masker

Posisi Dataset Tidak Pakai Masker	Adun		Hanif		Rahmat	
	Berhasil	Gagal	Berhasil	Gagal	Berhasil	Gagal
Sedikit Menghadap Ke Atas	25	0	23	2	25	0
Sedikit Menghadap Ke Bawah	25	0	25	0	20	5
Sedikit Menghadap Ke Kanan	25	0	25	0	23	2
Sedikit Menghadap Ke Kiri	25	0	24	1	25	0

Berdasarkan Tabel 3 dapat diketahui persentase tingkat akurasi sistem untuk setiap *dataset* dengan menggunakan persamaan (1) dapat dilihat pada Gambar 6



Gambar 6. Tingkat Akurasi Setiap Dataset Ketika Tidak Pakai Masker

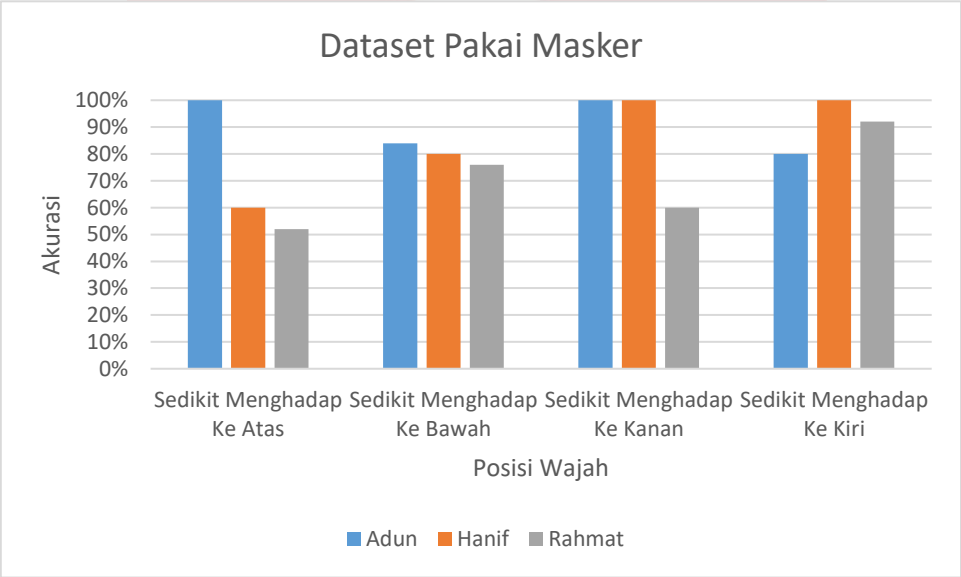
4.5 Pengujian Berdasarkan Posisi Wajah dengan Jarak Wajah ke Kamera Tidak Berubah Ketika Memakai Masker

Untuk setiap *dataset* wajah dilakukan percobaan pengenalan wajah ketika memakai masker dengan posisi wajah sedikit menghadap ke atas, ke bawah, ke kanan, dan ke kiri tetapi jarak wajah ke kamera tidak berubah, percobaan dilakukan sebanyak 25 kali percobaan. Hasil dari percobaan tersebut dilihat pada Tabel 4

Tabel 4. Hasil Pengujian Setiap *Dataset* Ketika Tidak Pakai Masker

Posisi Dataset Pakai Masker	Adun		Hanif		Rahmat	
	Berhasil	Gagal	Berhasil	Gagal	Berhasil	Gagal
Sedikit Menghadap Ke Atas	25	0	15	10	13	12
Sedikit Menghadap Ke Bawah	21	4	20	5	19	6
Sedikit Menghadap Ke Kanan	25	0	25	0	15	10
Sedikit Menghadap Ke Kiri	20	5	25	0	23	2

Berdasarkan Tabel 4 dapat diketahui persentase tingkat akurasi sistem untuk setiap *dataset* dengan menggunakan persamaan (1) dapat dilihat pada Gambar 7



Gambar 7. Tingkat Akurasi Setiap Dataset Ketika Pakai Masker

5. Kesimpulan

Dinilai dari hasil pengujian dan analisis yang dilakukan terhadap sistem pengenalan wajah dan masker yang telah dibuat dengan metode *deep metric learning* yang memanfaatkan *library dlib* dan *face_recognition*, dapat disimpulkan bahwa sistem tersebut dapat mengenali 3 citra wajah yang berbeda, baik itu ketika sedang tidak menggunakan masker ataupun sedang menggunakan masker. Dapat dirangkum juga akurasi dengan beberapa skenario pengujian, sebagai berikut:

1. Pengujian berdasarkan jarak wajah:
   a. Tidak memakai masker:
      - Akurasi tertinggi: 100 %
      - Akurasi terendah: 84 %
   b. Memakai masker:
      - Akurasi tertinggi: 100 %
      - Akurasi terendah: 72 %
2. Pengujian berdasarkan posisi wajah:
   a. Tidak memakai masker:
      - Akurasi tertinggi: 100 %
      - Akurasi terendah: 80 %
   b. Memakai masker:
      - Akurasi tertinggi: 100 %
      - Akurasi terendah: 52 %

Hasil tersebut menunjukkan bahwa *deep metric learning* merupakan salah satu metode yang efisien dalam melakukan pengenalan citra wajah. Mengamati hasil pengujian yang ada pada bagian 4, maka terlihat peningkatan akurasi dalam mengenali wajah dibandingkan penelitian serupa yang menggunakan Algoritma *Viola-Jones*[8] dan *Eigen face and PCA Algorithms*[9].

Referensi:

[1] Novita Anik Iswanti. 2019. "Implementasi Algoritma *Viola-Jones* Untuk Deteksi Wajah Tampak Depan". Yogyakarta. Universitas Teknologi Yogyakarta.
[2] P. Wagh and dkk, "*Attendance System based on Face Recognition using Eigen face and PCA Algorithms*," IEEE, pp. 303 - 308, 2015.
[3] A.Kadir, *Dasar Raspberry Pi*, Edisi 1. 2017.
[4] D. Suprianto, R. N. Hasanahm, and P. B. Santosa, "Sistem Pengenalan Wajah Secara *Real-Time* dengan *Adaboost, Eigenface PCA & MySQL*," J. EECCIS, vol. 7, no. 2, pp. 179–184, 2013.
[5] Hua-chunYang, Xu An Wang, *Cascade face detection based on Histogram of Oriented Gradients and Support Vector Machine, School of Life Science and technology Xidian University, Department of Information Engineering, Engineering University of Armed Police Xi'an*, China, 2015.
[6] R. R. Palekar, S. U. Parab, D. P. Parikh, and V. N. Kamble, "*Real time license plate detection using openCV and tesseract*," Proc. 2017 IEEE *Int. Conf. Commun. Signal Process*. ICCSP 2017, vol. 2018-Janua, pp. 2111–2115, 2018.
[7] Davis E. King, (2009). "*Dlib-ml: A Machine Learning Toolkit*". *Journal of Machine Learning Research*, vol 10,pp.1755-1758
[8] F. Schroff, D. Kalenichenko and J. Philbin. (2015). "*FaceNet: A unified embedding for face recognition and clustering*". IEEE *Conference on Computer Vision and Pattern Recognition* (CVPR), Boston, MA, pp. 815-823.
[9] He, K., Zhang, X., Ren, S., & Sun, J. (2016). "*Deep Residual Learning for Image Recognition". IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 770-778.
[10] B. Amos, B. Ludwiczuk, M. Satyanarayanan. (2016). "*Openface: A general-purpose face recognition library with mobile applications". CMU-CS-16-118, CMU School of Computer Science, Tech. Rep*.,
[11] Parkhi, O. M. and Vedaldi, A. and Zisserman, A. (2015). "*Deep Face Recognition". British Machine Vision Conference*.