IMPLEMENTASI FACE RECOGNITION SECARA REAL-TIME DENGAN METODE HAAR CASCADE CLASSIFIER MENGGUNAKAN OPENCV-PYTHON

[1]Windy Dwiparaswati
[2]Sephikhar Varid Hilmawan

[1]Universitas Gunadarma, windy_dwi@staff.gunadarma.ac.id
[2]Universitas Gunadarma, sephikhar@student.gunadarma.ac.id

ABSTRAK

Pengolahan citra memiliki keterhubungan dengan Computer Vision, hanya Computer Vision dapat dikaitkan dengan akuisisi citra, pemrosesan, klasifikasi, pengakuan dan pencakupan keseluruhan, dan pengambilan keputusan yang diikuti dengan pengidentifikasian citra. Penelitian ini menggunakan OpenCV bersama Python yang dimanfaatkan untuk mengolah image atau video (tumpukan frame/image) sesuai dengan tujuan masing-masing yang melibatkan kamera untuk menangkap gambar lalu diolah di komputer. Metode yang digunakan adalah metode Haar Cascade classifier, Haar-like feature memproses gambar dalam kotak-kotak, dimana dalam satu kotak terdapat beberapa pixel. Per kotak itu pun kemudian di-proses dan didapatkan perbedaan nilai (threshold) yang menandakan daerah gelap dan terang, yang nantinya dijadikan dasar dalam image processing. Penelitian ini dianalisa dengan menggunakan 7 faktor diantaranya adalah faktor pencahayaan terang dan gelap, faktor jarak wajah jauh dan dekat, dan faktor posisi wajah hadap atas, depan, dan bawah. Hasil penelitian ini adalah Sistem berhasil mengimplementasikan Face Recognition untuk mendeteksi wajah seseorang yang dikenal secara live video realtime dengan metode Haar Cascade Classifier menggunakan opencv berbasis python. Sistem dapat mengenali wajah yang dikenal sebagai pengguna awal pada faktor jarak wajah dekat dan jauh, pada faktor posisi wajah hadap depan dan hadap bawah. Pada faktor pencahayaan terlalu terang dan gelap, dan posisi wajah hadap atas, program menghasilkan output tidak dikenal sebagai pengguna awal karena memiliki nilai akurasi dibawah 60%.

Kata kunci: Computer Vision, Face Recognition, Metode Haar Cascade Classifier.

PENDAHULUAN

Computer Vision merupakan salah satu ilmu yang mempelajari teknologi Face recognition. Face recognition itu sendiri adalah salah satu bagian dari Face Processing yang mampu mengidentifikasi atau memverfikasi wajah seseorang melalui sebuah gambar atau video digital (Saphiro, 2001). Adanya sebuah gambar atau video yang menunjukkan wajah seseorang dengan tingkat keakuratan pada suatu sistem. Sistem yang memiliki nilai akuratnya tinggi diharapkan akan lebih mudah digunakan dan dipercaya oleh pengguna.

Pengenalan wajah adalah adalah proses mengidentifikasi atau memverifikasi sebuah citra wajah yang tidak diketahui dengan algoritma komputasi, dan membandingkannya dengan data wajah yang ada (R. Kumar and S. Singh, Face, 2013). Metode Haar-like feature memproses gambar dalam kotak-kotak, dimana dalam satu kotak terdapat beberapa pixel. Per kotak itu pun kemudian di-proses dan didapatkan perbedaan nilai (threshold) yang menandakan daerah gelap dan terang. Nilai – nilai inilah yang

nantinya dijadikan dasar dalam image processing. Lalu untuk gambar bergerak(video), perhitungan dan penjumlahan pixel terjadi secara terus – menerus dan membutuhkan waktu yang lama. Oleh karena itu, penjumlahan diganti dengan integral sehingga didapatkan hasil lebih cepat. Hasil deteksi dari Haar-Like kurang akurat jika hanya menggunakan satu fungsi saja sehingga biasanya digunakan beberapa fungsi sekaligus (massal). Semakin banyak fungsi yang digunakan maka hasilnya akan semakin akurat. Pemrosesan Haar-Like feature yang banyak tersebut diorganisir atau diatur di dalam *classifier cascade*.

Teknologi Face Recognition ini dapat bekerja dengan mendeteksi wajah seseorang, hal ini berguna untuk mencegah perilaku menyimpang atau terjadinya tindak kejahatan. Prinsip dari Face recognition itu sendiri adalah objek wajah yang tertangkap kamera yang diikuti pergerakannya ke arah kanan dan kiri. Implementasi ini akan menggunakan webcam, dan menggunakan metode Haar Cascade untuk mengenali wajah. Metode ini menghitung perbedaan jumlah setiap piksel pada daerah persegi panjang yang berdekatan pada lokasi tertentu dalam frame deteksi.

METODE PENELITIAN

Tahap pertama yang dilakukan dalam penelitian ini adalah membuat diagram alur pemrograman. Pemrograman dimulai dengan adanya inisialisasi, kemudian dilanjutkan dengan proses mengaktifkan webcamera pada laptop pengguna, dan memasukkan ID = 1, yang berarti 1 adalah ID pengguna dari wajah yang akan dikenali. Proses selanjutnya adalah mendeteksi wajah, jika tidak ada wajah yang terdeteksi maka frame akan terus mencari wajah yang akan dideteksi, jika terdeteksi ada wajah makan program akan mengambil citra wajah sebanyak 200 kali sebagai data sample wajah untuk membentuk dataset. Kemudian kembali diproses oleh program untuk mentrainer dataset, dengan tujuan untuk melatih data model pengenalan wajah. Pengenalan wajah oleh webcam akan mengeluarkan output teks berupa nama wajah yang dikenal dan nilai akurasi dari program tersebut, jika wajah tidak dikenal makan akan mengeluarkan output berupa “Tidak dikenal” dengan diikuti nilai akurasi yang rendah. Berikut diagram alur program dapat dilihat pada Gambar 1.

Logika Program Face Detection (Pengenalan Wajah)

Gambar 2 adalah logika program pada file capture.py yang merupakan program untuk mendeteksi apakah ada wajah yang dapat dideteksi dari frame yang telah ada. Jika wajah telah terdeteksi pada frame, maka akan diambil gambarnya pada Data Capture sebanyak 200 kali sample dan disimpan dalam folder Data Wajah.

Logika Program Training

Pada Gambar 3 menjelaska program selanjutnya yaitu mentraining data model pada file trainer.py. Tujuan dari mentraining data ini adalah melatih data model untuk mengenali wajah dengan mengambil data pada folder Data Wajah yang telah tersimpan setelah melakukan pendeteksian wajah.

Logika Program Face Recognition (Pendeteksian Wajah)

Setelah melakukan training pada program, maka program akan menghasilkan informasi apakah wajah dikenal atau tidak dikenal, dengan nilai akurasi yang sesuai dengan hasil pendeteksian wajah. Berikut ini adalah potongan kode program dari pendeteksian wajah dapat dilihat pada Gambar 4.

Pada bagian kode program berikut adalah mengimport library yang ada pada openCV untuk dipanggil ke dalam program, seperti import cv2, import numpy as np, import datetime, import time, dan import RPi.GPIO as GPIO.

Selanjutnya pada bagian Gambar 4b adalah mendefinisikan author dan unauthor dengan memberikan pesan "Terdeteksi Wajah" sesuai dengan wajah author dan "Terdeteksi Wajah Tidak Dikenal" pada wajah unauthor.

Pada Gambar 4c menjelaskan algoritma dari Face detection menggunakan metode Haar Cascade Classifier. Kode program face_cascade =cv2. cascade Classifier adalah mendapatkan data wajah dengan metode tersebut, kemudian dimasukkan kedalam array.

HASIL DAN PEMBAHASAN

Analisa Hasil dari penulisan ini dibuat dengan memberikan kondisi 7 faktor yang dapat mempengaruhi hasil dari pendeteksian wajah. Faktor tersebut adalah Pencahayaan yang lebih terang, pencahayaan yang lebih gelap, posisi wajah dari jarak dekat dan jauh, dan posisi wajah hadap atas, depan, dan belakang. Dari faktor-faktor ini menghasilkan nilai akurasi yang berbeda sesuai dengan tingkat keberhasilan dalam mengenali wajah.

Hasil Pencahayan

Uji coba pengenalan wajah terhadap cahaya pengambilan citra ini akan dianalisis pengaruhnya terhadap akurasi pengenalan. Ada dua macam kondisi yang diambil dalam sampel yaitu terang dan gelap. Pada pengujian ini kategori terang pengambilan citra dengan banyak cahaya dan kategori gelap adalah pengambilan citra sedikit cahaya. Berikut hasil pencahyaan terang dapat dilihat pada Gambar 5a dan hasil pencahayaan gelap dapat dilihat pada gambar 5b.

Berikut ini adalah tabel hasil analisa faktor pencahayaan terang dan gelap, dengan melakukan percobaan sebanyaak 5 kali dan dengan wajah orang yang telah tersimpan data wajahnya.

Hasil pengujian pada pencahayaan terang dan gelap dengan 5 sample yang sama tidak dapat mengenali wajah pengguna awal, karena nilai akurasi dibawah 60%. Rata-rata nilai akurasi pada faktor pencahayaan terang adalah sebesar 25%, dan rata-rata nilai akurasi pada faktor pencahayaan gelap adalah 34%.

Hasil Jarak Wajah

Uji coba pada pengenalan wajah yang diambil dengan jarak dekat dan jauh. Dimana dalam penelitian range untuk jarak dekat = 20-30cm, dan untuk jarak jauh = 31-60cm dari kamera. Berikut ini adalah hasil jarak wajah secara dekat yang dapat dilihat pada gambar 7 dan jarak wajah secara jauh pada gambar 8.

Berikut ini adalah tabel hasil analisa faktor jarak wajah dekat dan jauh, dengan melakukan percobaan sebanyaak 5 kali dan dengan wajah orang yang telah tersimpan data wajahnya.

Hasil pengujian pada jarak wajah dekat dan jauh dengan 5 sample yang sama dapat mengenali wajah pengguna awal, dengan nilai akurasi lebih besar sama dengan 60%. Rata-rata nilai akurasi pada faktor jarak wajah dekat adalah sebesar 68%, dan rata-rata nilai akurasi pada faktor jarak wajah jauh adalah 61%.

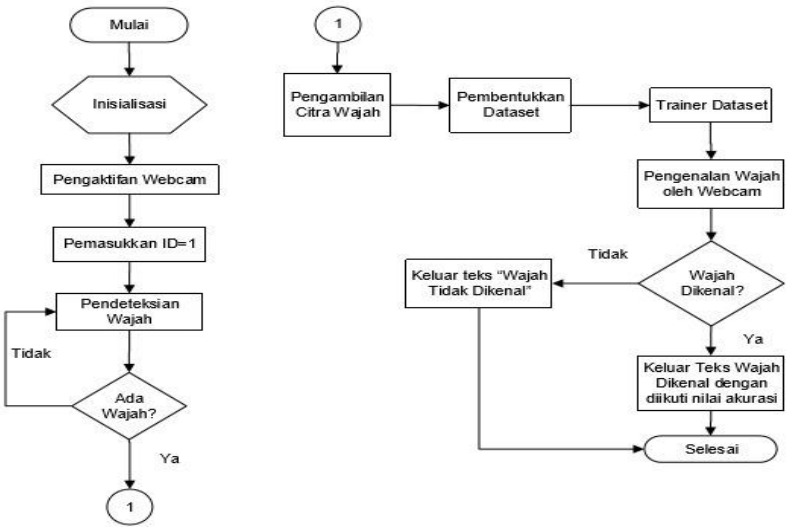
Hasil Posisi Wajah

Uji coba pada pengenalan wajah yang dengan melakukan 3 posisi berbeda, posisi wajah hadap atas. Posisi wajah hadap depan, dan posisi wajah hadap bawah. Berikut ini adalah hasil posisi wajah hadap atas yang dapat dilihat pada gambar 9, posisi wajah

hadap depan yang dapat dilihat pada gambar 10, dan posisi wajah hadap bawah yang dapat dilihat pada gambar 11.

Berikut ini adalah tabel hasil analisa faktor posisi wajah hadap atas, depan dan bawah, dengan melakukan percobaan sebanyaak 5 kali dan dengan wajah orang yang telah tersimpan data wajahnya.

Hasil pengujian pada posisi wajah hadap depan dan bawah dengan 5 sample yang sama dapat mengenali wajah pengguna awal, dengan nilai akurasi lebih besar sama dengan 60%. Dan pada posisi wajah hadap atas tidak dikenali sebagai pengguna awal karena nilai akurasi dibawah 60%. Rata-rata nilai akurasi pada faktor posisi wajah hadap atas adalah sebesar 52%, rata-rata nilai akurasi pada posisi wajah hadap depan adalah 69% dan rata-rata nilai akurasi pada posisi wajah hadap bawah adalah 64%.



Gambar 1 Diagram Alur Pemrograman

```
import cv2

videoCam = cv2.VideoCapture(0)
face = cv2.CascadeClassifier('face-detect.xml')
id=input("Masukkan ID : ")
sampleNum = 0

while True:
    cond, frame = videoCam.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    muka = face.detectMultiScale(gray, 1.3, 5)
    for (x,y,w,h) in muka:
        sampleNum = sampleNum+1
        cv2.imwrite("dataWajah/User."+str(id)+"."+str(sampleNum)+".jpg", gray[y:y+h, x:x+w])
        cv2.rectangle(frame, (x,y), (x+w, y+h), (0, 255, 0), 5)
        cv2.waitKey(100)

    cv2.imshow('Data Capture', frame)
    cv2.waitKey(1)
    if(sampleNum>200):
        break

videoCam.release()
cv2.destroyAllWindows()
```

Gambar 2 Program Face Detection

```
import os
import cv2
import numpy as np
from PIL import Image

recognizer=cv2.face.LBPHFaceRecognizer_create();
path='dataWajah'

def getImageWithID(path):
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
    faces=[]
    IDs=[]
    for imagePath in imagePaths:
        faceImg=Image.open(imagePath).convert('L');
        faceNp=np.array(faceImg,'uint8')
        ID=int(os.path.split(imagePath)[-1].split('.')[1])
        faces.append(faceNp)
        IDs.append(ID)
        cv2.imshow("training",faceNp)
        cv2.waitKey(10)
    return np.array(IDs), faces
Ids, faces=getImageWithID(path)
recognizer.train(faces,Ids)
recognizer.save('recognizer/dataTraining.yml')
cv2.destroyAllWindows()
```

Gambar 3 Program Training

```
import cv2
import numpy as np
import datetime
import time
import RPi.GPIO as GPIO
from PIL import *
from time import sleep
from twilio.rest import Client

servoPIN = 17
buzzer = 23
GPIO.setwarnings(False)
GPIO.setmode(GPIO.BCM)
GPIO.setup(servoPIN, GPIO.OUT)
GPIO.setup(buzzer, GPIO.OUT)
p = GPIO.PWM(servoPIN,50)
p.start(2)
def author():
    GPIO.output(buzzer, GPIO.LOW)
    sleep(0.5)
    p.ChangeDutyCycle(2)
    sleep(0.5)
```

Gambar 4a Program rec.py (Bagian 1)

```
def unauthor():
    p.ChangeDutyCycle(8)
    sleep(0.1)
    for i in range(1,15):
        GPIO.output(buzzer, GPIO.HIGH)
        sleep(0.3)
        GPIO.output(buzzer, GPIO.LOW)
        sleep(0.3)
def detect_sendwa(id):
# Account Sid and Auth Token yang diambil dari twilio.com/console
    account_sid = 'AC6f2c0d7787f393f3f44441135c5dcf6e'
    auth_token = 'f9225041ee79f943b78100efedbf04ed'
    client = Client(account_sid, auth_token)
    message = client.messages.create(body='Terdeteksi Wajah '+str(id),from_='whatsapp:+14155238886',to='whatsapp:+6289606987951')
    print(message.sid)
def undetect_sendwa():
# Account Sid and Auth Token yang diambil dari twilio.com/console
    account_sid = 'AC6f2c0d7787f393f3f44441135c5dcf6e'
    auth_token = 'f9225041ee79f943b78100efedbf04ed'
    client = Client(account_sid, auth_token)
    message = client.messages.create(body='Terdeteksi Wajah Tidak Dikenal',from_='whatsapp:+14155238886',to='whatsapp:+6289606987951')
    print(message.sid)
```

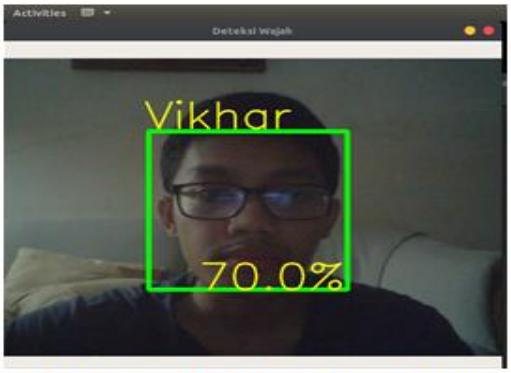
Gambar 4b Program rec.py (Bagian 2)

```
def mod(x):
    if(x%10==0):
        detect_sendwa()
        author()

def mod2(x):
    if(x%10==0):
        undetect_sendwa()
        unauthor()

videoCam = cv2.VideoCapture(0)
face_cascade = cv2.CascadeClassifier('face-detect.xml')

rec=cv2.face.LBPHFaceRecognizer_create();
rec.read("./recognizer/dataTraining.yml")
id=0
count = 0
font=cv2.FONT_HERSHEY_SIMPLEX
while True:
    cond, frame = videoCam.read()

    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    muka = face_cascade.detectMultiScale(gray, 1.3, 5)
    for (x,y,w,h) in muka:
        cv2.rectangle(frame, (x,y), (x+w, y+h), (0, 255, 0), 5)
        id,conf=rec.predict(gray[y:y+h, x:x+w])
        if(id==1):
            id = "Vikhar"
        if(conf<50):
            confid ="  {0}%".format(round(100-conf))
            cv2.putText(frame, str(id), (x-5,y-5),font,2, (0,255,255),2, cv2.LINE_AA)
            cv2.putText(frame,str(confid), (x,y+h), font, 2, (0,255,255), 2, cv2.LINE_AA)

            if(confid>="  {0}%".format(round(65))):
                count=count+1
                print(count)
                mod(count)
    else:
        confid ="  {0}%".format(round(100-conf))
        cv2.putText(frame, "Tidak dikenal" , (x-5,y-5),font,2, (0,255,255),2, cv2.LINE_AA)
        cv2.putText(frame,str(confid), (x,y+h), font, 2, (0,255,255), 2, cv2.LINE_AA)
        if(confid<="  {0}%".format(round(50))):
            count=count+1
            print(count)
            mod2(count)

    cv2.imshow('Camera Detection', frame)

    if(cv2.waitKey(1)==ord('q')):
        break

videoCam.release()
cv2.destroyAllWindows
```

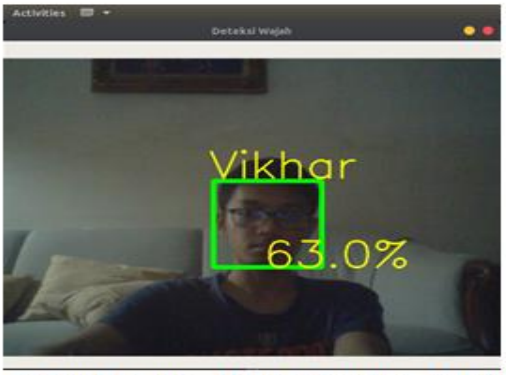
Gambar 4c Program rec.py (Bagian 3)

Gambar 5a Hasil Gambar Faktor Pencahayaan Terang

Gambar 5b Hasil Gambar Faktor Pencahayaan Gelap



Gambar 6a Hasil Gambar Faktor Jarak Wajah Dekat



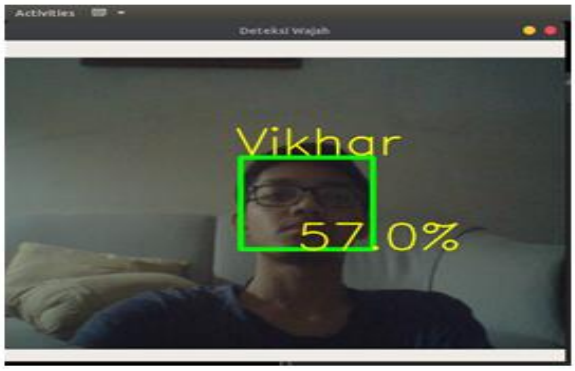
Gambar 6b Hasil Gambar Faktor Jarak Wajah Jauh

Tabel 1.
Hasil Faktor Pencahyaan Terang dan Gelap

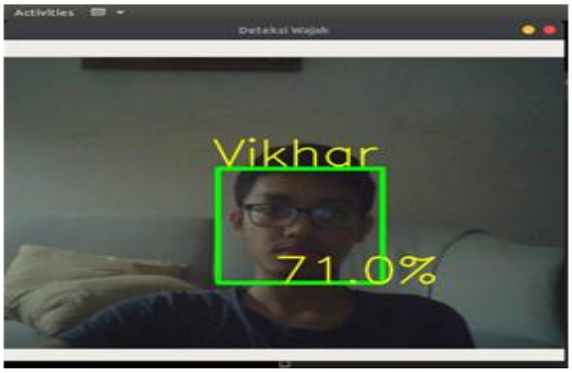
Faktor		Nilai
Pencahayaan	Terang	1. Tidak dikenali sebagai pengguna awal nilai akurasi 28% 2. Tidak dikenali sebagai pengguna awal nilai akurasi 36% 3. Tidak dikenali sebagai pengguna awal nilai akurasi 24% 4. Tidak dikenali sebagai pengguna awal nilai akurasi 15% 5. Tidak dikenali sebagai pengguna awal nilai akurasi 21%
	Gelap	1. Tidak dikenali sebagai pengguna awal nilai akurasi 37% 2. Tidak dikenali sebagai pengguna awal nilai akurasi 35% 3. Tidak dikenali sebagai pengguna awal nilai akurasi 34% 4. Tidak dikenali sebagai pengguna awal nilai akurasi 31% 5. Tidak dikenali sebagai pengguna awal nilai akurasi 32%

Tabel 2.
Hasil Faktor Jarak Wajah Dekat dan Wajah Jauh

Faktor		Nilai
Jarak Wajah	Dekat (20-30 m)	1. Wajah dikenali nilai akurasi 70% 2. Wajah dikenali nilai akurasi 67% 3. Wajah dikenali nilai akurasi 70% 4. Wajah dikenali nilai akurasi 65% 5. Wajah dikenali nilai akurasi 68%
	Jauh (31-60 m)	1. Wajah dikenali nilai akurasi 60% 2. Wajah dikenali nilai akurasi 60% 3. Wajah dikenali nilai akurasi 60% 4. Wajah dikenali nilai akurasi 63% 5. Wajah dikenali nili akurasi 62%



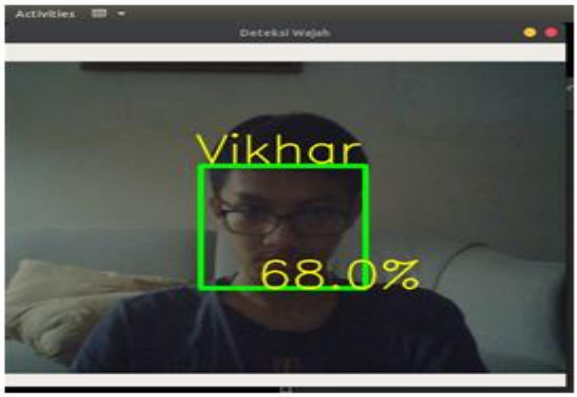
Gambar 7a Hasil Gambar Faktor Posisi Wajah Hadap Atas



Gambar 7b Hasil Gambar Faktor Posisi Wajah Hadap Depan

Tabel 3.
Hasil Faktor Posisi Wajah Hadap Atas, Depan dan Bawah

Faktor		Nilai
Posisi Wajah	Atas	1. Tidak dikenali sebagai pengguna awal nilai akurasi 57% 2. Tidak dikenali sebagai pengguna awal nilai akurasi 55% 3. Tidak dikenali sebagai pengguna awal nilai akurasi 45% 4. Tidak dikenali sebagai pengguna awal nilai akurasi 51% 5. Tidak dikenali sebagai pengguna awal nilai akurasi 52%
	Depan	1. Wajah dikenali nilai akurasi 69% 2. Wajah dikenali nilai akurasi 71% 3. Wajah dikenali nilai akurasi 68% 4. Wajah dikenali nilai akurasi 70% 5. Wajah dikenali nilai akurasi 67%
	Bawah	1. Wajah dikenali nilai akurasi 60% 2. Wajah dikenali nilai akurasi 65% 3. Wajah dikenali nilai akurasi 68% 4. Wajah dikenali nilai akurasi 62% 5. Wajah dikenali nilai akurasi 66%



Gambar 7c Hasil Gambar Faktor Posisi Wajah Hadap Bawah

KESIMPULAN DAN SARAN

Berdasarkan uraian sebelumnya, dapat disimpulkan bahwa:

1. Sistem berhasil mengimplementasikan Face Recognition untuk mendeteksi wajah seseorang yang dikenal secara live video realtime dengan metode Haar Cascade Classifier menggunakan opencv berbasis python**.**
2. Sistem dapat mengenali wajah yang dikenal sebagai pengguna awal pada faktor jarak wajah dekat dan jauh, pada faktor posisi wajah hadap depan dan hadap bawah. Pada faktor pencahayaan terlalu terang dan gelap, dan posisi wajah hadap atas, program menghasilkan output tidak dikenal sebagai pengguna awal karena memiliki nilai akurasi dibawah 60%.

Saran dalam penulisan ini diharapkan dalam penelitian selanjutnya sistem dapat digunakan pada sistem operasi yang lain, agar dapat digunakan oleh banyak pihak. Hasil akurasi dalam penelitian ini menunjukkan sistem sudah cukup baik dan siap diimplementasikan pada berbagai bidang, misalnya sistem absensi pada suatu institusi.

DAFTAR PUSTAKA

Informatika. (2013). Operasi Cropping. Diperoleh dari: http://informatika.web.id/operasicropping.html. (Diakses pada 29 Juli 2017).

R. Kumar and S. Singh, Face. (2013). Recognition With Its Various Techniques: A Review , International Journal of Innovative Research and Studies, September, Vol 2 Issue 9.

Shapiro, Linda ; Stockman, George. Computer Vision. New Jersey: Prentice Hall.(2001).

Kandir, Nor. (2016). OpenCv dengan Python. Diambil dari https://norkandirblog.files.wordpress.com/2016/12/opencv-dengan-python-nor-kandir.pdf

Jurnal Komputer Terapan. (2016). *Faktor-Faktor yang Mempengaruhi Sistem Pengenalan Wajah Menggunakan Metode Eigenface pada Perangkat Mobile Berbasis Android.* Vol.2, No. 2, November 2016, 127-136. Diambil dari https://www.semanticscholar.org/