Terbit online pada laman web jurnal: **https://ejurnalunsam.id/index.php/jitkom/**

Jurnal Informatika dan Teknologi Komputer

Vol. 01 No. 01 (2020) x - x ISSN Media Elektronik: -

Sistem Deteksi Multi Wajah Menggunakan Metode Haar Cascade Classifier

Mariana Fitri Sitorus[1], Ruci Fatharani[2], Nurul Fadhillah[3]
[1,2,3]Teknik Informatika, Fakultas Teknik, Universitas Samudra
[1]marianafitri98@gmail.com*, [2]rucifatharani@gmail.com, [3] nurulfadillah@unsam.ac.id

Abstract

Face detection with both still and moving images is an important topic today. Human face detection is also one of the domains in applications in both the security and social media fields. The Haar Cascade Classifier method is good for detecting multiple faces in real time. Based on the indication of face detection in real time, the number of faces will be known. So that the authors use this method for articles created, because this study aims to create a system that can calculate the number of human faces detected in real time by utilizing the OpenCV and Python libraries. The result of this test is that the system can detect the number of faces well, but the face that you want to detect the position must be straight facing the camera and get adequate lighting.

Keywords: Face Detection, Haarcascade Classifier, OpenCV and Python.

Abstrak

Deteksi wajah baik dengan gambar diam maupun bergerak merupakan topik penting saat ini. Deteksi wajah manusia juga merupakan salah satu domain dalam aplikasi baik pada bidang keamanan maupun sosial media. Metode *Haar Cascade Classifier* baik digunakan untuk mendeteksi banyak wajah secara *real time*. Berdasarkan indikasi dari deteksi wajah secara *real time*, maka jumlah wajah akan diketahui. Sehingga penulis menggunakan metode tersebut untuk artikel yang dibuat, dikarenakan penelitian ini bertujuan untuk membuat sistem yang dapat menghitung jumlah wajah manusia yang terdeteksi secara *real time* dengan memanfaatkan *library OpenCV* dan *Python*. Hasil dari pengujian ini adalah sistem dapat mendeteksi jumlah wajah dengan baik, akan tetapi wajah yang ingin dideteksi posisinya haruslah lurus menghadap ke kamera dan mendapatkan pencahayaan yang memadai.

Kata kunci: Deteksi wajah, *Haarcascade Classifier*, *OpenCV* dan *Python.*

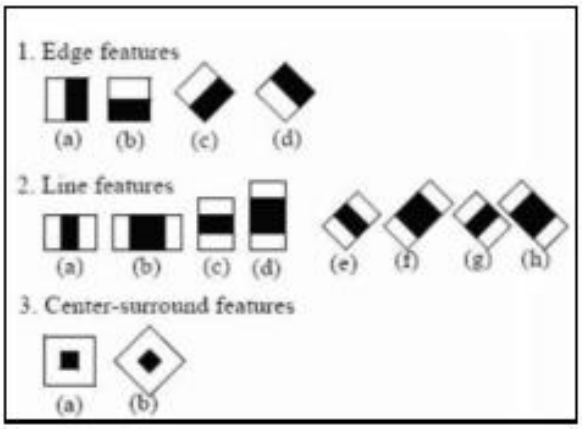
1. Pendahuluan

Deteksi wajah baik dengan gambar diam maupun bergerak merupakan topik penting saat ini. Proses deteksi keberadaan wajah ini menjadi dasar proses pengenalan wajah yang mempunyai banyak implementasi baik pada bidang keamanan maupun sosial media [1]. Deteksi wajah manusia juga merupakan salah satu domain dalam aplikasi *computer vision* [2].

Pemanfaatan *computer vision* dalam mengambil citra wajah manusia sudah banyak diterapkan dan pemanfaatannya pun bermacam-macam [3]. Tujuan dari proses deteksi wajah adalah untuk mengetahui adanya wajah dari suatu citra dan menemukan letak keberadaan wajah.

Proses deteksi wajah dan perhitungan jumlah yang dapat dideteksi memerlukan metode tertentu yang didukung dengan suatu perangkat lunak. Oleh karena itu, perlu dibuat sistem yang mampu mengidentifikasi dan menghitung semua daerah citra yang mengandung wajah [4] .Dalam penelitian ini digunakan metode *Haar Cascade Classifier* yang merupakan *rectangular* (persegi) *feature*, yang memberikan indikasi secara spesifik pada sebuah gambar atau *image* yang mana sangat ideal digunakan untuk mendeteksi banyak wajah diruang kelas secara *real time*. *Haar Cascade Classifier* berasal dari gagasan Paul Viola dan Michael Jhon, karena itu dinamakan metode Viola & Jhon. Ide dari *Haar like feature* adalah mengenali objek berdasarkan nilai sederhana dari fitur tetapi bukan merupakan nilai piksel dari *image* objek tersebut. Metode ini memiliki kelebihan yaitu komputasi yang sangat cepat, karena

tergantung pada jumlah piksel dalam persegi bukan setiap nilai piksel dari sebuah image. Metode ini merupakan metode yang menggunakan statistikal model *(Classifier)*. Pendekatan untuk mendeteksi objek dalam gambar menggabungkan empat kunci utama yaitu *Haar like feature, Integral Image, Adabootst learning* dan *Cascade Classifier* [5].



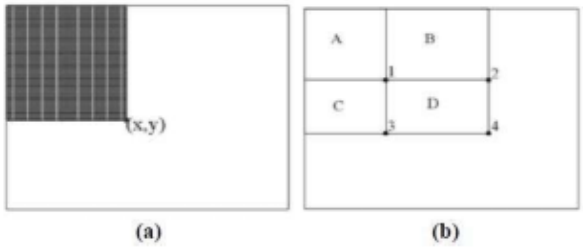
Gambar 1. Haar Cascade Classifier(Haar Like Feature)

Haar feature sebagai dasar fitur pada *Walvelet Haar*. *Wavelet Haar* sebagai gelombang tunggal bujur sangkar (satu interval tinggi dan satu interval rendah). Untuk dua dimensi, satu terang dan satu gelap. Selanjutnya kombinasi-kombinasi kotak yang digunakan untuk pendeteksian objek visual yang lebih baik. Setiap *Haar Like Feature* terdiri dari gabungan kotak-kotak hitam dan putih [5].

$$f(x) = SumBlack\ retangle - SumWhite\ retangle \quad (1)$$

Adanya *feature Haar* ditentukan dengan cara mengurangi rata-rata piksel pada daerah gelap dari rata-rata piksel pada daerah terang. Jika nilai perbedaannya itu diatas nilai ambang atau *treshold*, maka dapat dikatakan bahwa fitur tersebut ada. Nilai dari *Haar Like Feature* sebagai perbedaan antara jumlah nilai-nilai piksel *gray* level dalam daerah kotak hitam dan daerah kotak putih. Dimana untuk kotak pada *Haar like feature* dapat dihitung secara cepat menggunakan *"integral image"* [5].

Integral Image digunakan sebagai penentu ada atau tidaknya ratusan fitur *Haar* pada sebuah gambar dan pada skala yang berbeda secara efisien [5].



Gambar 2. Integral Image

Seperti yang ditunjukkan oleh gambar diatas setelah pengintegrasian, nilai pada lokasi piksel (x,y) berisi jumlah dari semua piksel di dalam daerah segi empat dari kiri atas sampai pada lokasi (x,y) atau daerah yang diarsir. Guna mendapatkan nilai rata-rata piksel pada area segi empat (daerah yang diarsir) ini dapat diklakukan hanya dengan membagi nilai pada (x,y) oleh area segi empat [5].

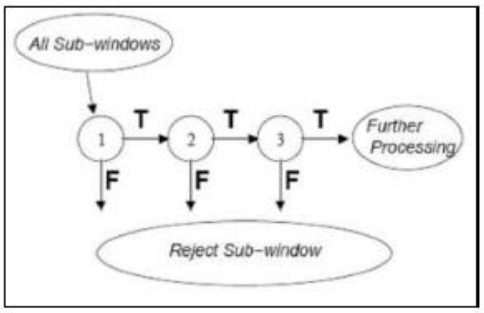
$$\mathrm{ii}(x,y) = \sum_{x^1 dx_1\ y^1 dy} \mathrm{i}(x^1, y^1) \quad (2)$$

Dimana :

ii(x,y) = integral image

i(x,y) = original image.

Sebuah metode untuk mengkombinasikan *classifier* yang kompleks dalam sebuah struktur bertingkat yang dapat meningkatkan kecepatan pendeteksian objek dengan memfokuskan pada daerah citra yang berpeluang saja [5].



Gambar 3. Cascade Classifier

Berdasarkan indikasi dari deteksi wajah secara *real time*, maka jumlah wajah akan diketahui. Jika citra wajah terhalang oleh objek lain maka citra wajah tersebut tidak akan terdeteksi [4].

Penelitian ini bertujuan untuk membuat sistem yang dapat menghitung jumlah wajah manusia yang terdeteksi secara *real time* dengan memanfaatkan *library OpenCV* dan *Python*.

OpenCV (Intel Open Source Computer Vision Library) terdiri dari sekurang-kurangnya 300 fungsi-fungsi C, bahkan lebih. *OpenCV* dapat beroperasi pada komputer berbasis *Windows* atau *Linux. Library OpenCV* digunakan untuk meng-*update* penerapan *computer vision*. *Software* ini menyediakan sejumlah fungsi-fungsi *image processing*, seperti fungsi-fungsi analis gambar dan pola [4].

Python adalah salah satu bahasa pemrograman tingkat tinggi yang bersifat *interpreter, interactive, object oriented,* dan dapat beroperasi hampir disemua platform: *Mac, Linux, dan Windows, python* termasuk bahasa pemrograman yang mudah dipelajari karena sintaks yang jelas, dapat dikombinasikan dengan penggunaan modul-modul siap pakai dan struktur data tingkat tinggi yang efisien. Distribusi *Python* dilengkapi dengan suatu fasilitas seperti shell di *Linux*. Okasi

penginstalan *Python* biasa terletak di *"/usr?bin/python"*, dan bisa berbeda. Menjalankan *Python* cukup dengan mengetikkan "*Python"*, tunggu sebentar lalu muncul tampilan *">>>"*, berarti *Python* telah siap menerima perintah. Ada juga tanda "…" yang berarti baris berikutnya dalam suatu blok prompt *">>>". Text edition* digunakan sebagai modus skrip [6].

Contoh penerapan *OpenCV* dengan *Python* adalah kamera yang dipasang diparkiran yang mampu membaca plat nomor. Plat nomor ini dikonversi dari analog ke digital lalu diolah menjadi karakter sehingga menjadi data yang bida dijadikan sebagai informasi penting [7].

Intinya, *OpenCV* bersama *Python* dimanfaatkan untuk mengolah image atau video (tumpukan *frame/image*) sesuai dengan tujuan masing-masing yang melibatkan kamera yang menangkap gambar lalu diolah di komputer [7].

2. Metode Penelitian

Dalam bab ini, meliputi penjelasan mengenai sistem yang dapat mengitung jumlah wajah yang terdeteksi pada *webcam*, yang mana metode yang digunakan adalah *Haar Cascade Classifier*.

Berikut ini adalah cara kerja dari sistem deteksi wajah.

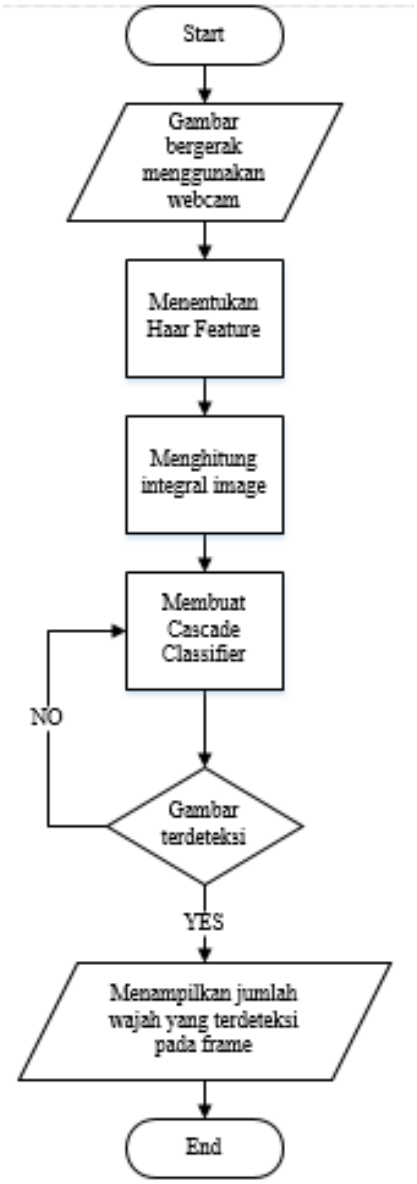
Gambar 4. Flowchart Proses Sistem

A. Inisialisasi Kamera

Inisialisasi kamera dilakukan mulai dari menginisialisasikan video masukan sampai pemrosesan *image*. Inisialisasi kamera sebagai proses penentuan awal semua hal yang diperlukan untuk menjalankan proses selanjutnya. Citra wajah yang terdeteksi adalah wajah yang posisinya menghadap ke kamera.

B. Pendeteksian Wajah dengan Metode *Haar Cascade Classifier*

Pada penelitian ini digunakan metode *Haar Cascade Classifier* sebagai metode penghitung jumlah wajah yang terdeteksi. Berikut ini alur proses metode *Haar Cascade Classfier.*



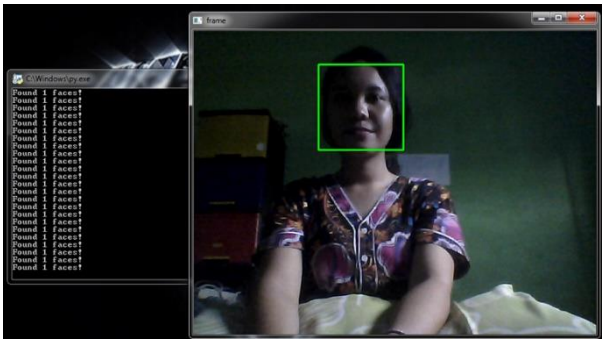
Gambar 5. Flowchart Metode Haar Cascade Classifier.

Gambar diatas menunjukan aliran dari cara kerja metode *Haar Cascade Claassifier* pada penelitian ini. Yang mana gambar secara *real time* yang ditangkap oleh *webcam* akan dideteksi dengan menentukan *Haar*

Feature terlebih dulu, lalu dilanjutkan dengan memproses *integral image* yang terhitung kemudian membuat *Cascade Classifier*. Apabila gambar terdeteksi maka hasilnya adalah jumlah dari wajah yang terdetaksi, tetapi apabila tidak terdeteksi maka sistem akan mengulang kembali pembuatan *Cascade Classifier*, hingga sampai mendapatkan keluaran seperti yang dinginkan.

3. Hasil dan Pembahasan

Pengujian dilakukan untuk mengetahui jumlah dari wajah yang terdeteksi dengan menggunakan metode *Haar Cascade Classifier*. Gambar dibawah ini merupakan *screenshot* hasil dari sistem yang dibuat dengan menggunakan *OpenCV* dan *Python*. Sistem melakukan pencarian wajah ke berbagai lokasi citra dan mendeteksi wajah dengan *detecMultiScale* yang merupakan fungsi umum untuk mendeteksi objek. *ScaleFactor* yang digunakan pada sistem adalah 1.1. Hal itu dikarenakan adanya beberapa wajah yang lebih dekat ke kamera, sehingga wajah tersebut akan terlihat lebih besar dari pada wajah yang lainnya. Oleh karena itu dibutuhkan *ScaleFactor* untuk memperbaiki hal ini. Algoritma pendeteksian menggunakan window yang bergerak untuk mendeteksi objek dan *minNeighbors yang* mendefinisikan seberapa banyak objek yang terdeteksi sebelum fungsi menyatakan menemukan wajah. *Window* pada sistem ini hanya berukuran 30x30.



Gambar 6. Satu wajah

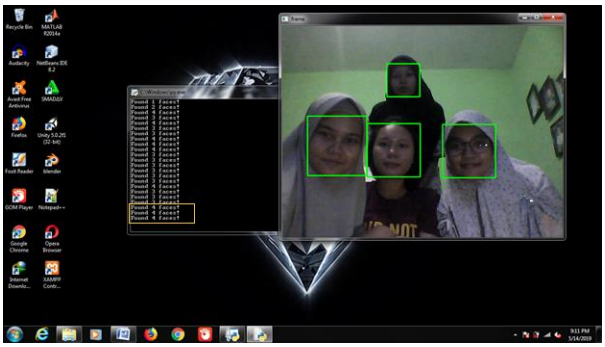
Gambar diatas menunjukkan bahwa sistem dapat menemukan wajah yang terdeteksi dengan jarak 49 cm dari kamera ke wajah, dan sistem menemukan jumlah satu wajah, itu terlihat dari *window* yang ada di sebelah kiri *"Found 1 Face!"*.

Gambar 7. Dua wajah

Gambar diatas menunjukan bahwa sistem dapat menemukan wajah yang terdeteksi yaitu dengan jumlah dua wajah, itu terlihat dari *window* yang ada di sebelah kiri *"Found 2 Face!"*. Jarak antar kamera ke wajah adalah 62 cm.

Gambar 8. Tiga wajah

Gambar diatas menunjukan bahwa sistem dapat menemukan wajah yang terdeteksi yaitu dengan jumlah tiga wajah, itu terlihat dari *window* yang ada di sebelah kiri *"Found 3 Face!"*. Jarak antara kamera ke wajah adalah 62 cm.



Gambar 9. Empat wajah

Gambar diatas menunjukkan bahwa sistem dapat menemukan wajah yang terdeteksi, yaitu dengan jumlah empat wajah yang terlihat pada *window* yang ada di sebelah kiri *"Found 4 Face!"*. Sebelumnya sistem tidak dapat mendeteksi jumlah wajahnya dengan baik, terlihat pada *window* sebelah kiri yang tidak ditandai, itu karenakan faktor posisi wajah dan pencahayaan sehingga membutuhkan waktu yang lebih lama untuk menyesuaikan antara kamera, posisi wajah dan pencahayaan. Jarak antar kamera ke wajah terdekat adalah 50 cm dan jarak terjauhnya adalah 93 cm.

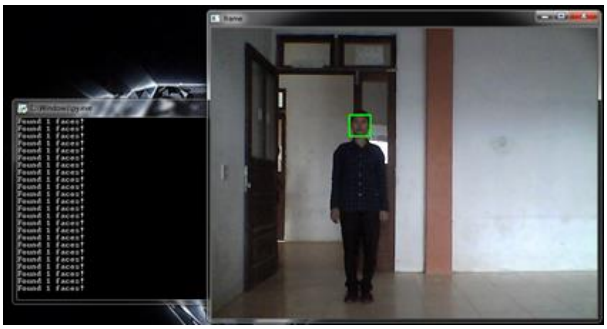
Gambar 10. Lima wajah

Gambar diatas menunjukkan bahwa sistem dapat menemukan wajah yang terdeteksi yaitu dengan jumlah lima wajah, itu terlihat dari *window* yang ada di sebelah

kiri *"Found 5 Face!"*. Disini sistem dapat mendeteksi jumlah wajah dengan baik secara cepat dan akurat. Dikarenakan faktor pencahayaan yang memadai dan posisi wajah yang lurus menghadap ke kamera, jarak antar kamera ke wajah terdekat adalah 51 cm dan terjauhnya adalah 100 cm.

Gambar 11. Enam wajah

Gambar diatas menunjukkan bahwa sistem dapat menemukan wajah yang terdeteksi yaitu dengan jumlah enam wajah, itu terlihat dari *window* yang ada di sebelah kiri *"Found 5 Face!"*. Pada bagian ini sistem dapat mendeteksi jumlah wajah dengan baik, cepat dan akurat walaupun beberapa wajah mempunyai jarak yang jauh dari kamera yaitu 296 cm.



Gambar 12. Jarak jangkauan terjauh

Gambar diatas menunjukkan bahwa sistem dapat menemukan wajah yang terdeteksi yaitu dengan jumlah satu wajah, itu terlihat dari *window* yang ada di sebelah kiri *"Found 1 Face!"*. Gambar tersebut juga menunjukkan bahwa sistem dapat mendeteksi dan menghitung jumlah wajah dari jarak yang jauh, yaitu dengan jarak jangkauan terjauhnya adalah 361 cm.

Tabel 1. Hasil Pengujian

No.	Nama Image	Jmlh Objek Wajah	Hasil Deteksi Jumlah Wajah		Jarak Objek ke Kamera	Hasil
			Objek Wajah	Objek Lain		
1.	Gmbr 6	1	1	0	46 cm	Benar
2.	Gmbr 7	2	2	0	62 cm	Benar
4	Gmbr 8	3	3	0	62 cm	Benar
5.	Gmbr 9	4	4	0	50 cm; 93 cm	Benar
6.	Gmbr 10	5	5	0	51 cm ; 100 cm	Benar
7.	Gmbr 11	6	6	0	51 cm ; 296 cm	Benar
8.	Gmbr 12	1	1	0	361 cm	Benar

4. Kesimpulan

Setelah melakukan pengujian pada sistem deteksi jumlah wajah menggunakan metode *Haar Cascade Classifier*, maka diperoleh hasil pengujian sebagai berikut :

a. Pada pengujian ini jika objek tidak lurus menghadap ke kamera maka sistem tidak dapat mendeteksi jumlah wajah tersebut. Seperti yang terlihat pada gambar 9.
b. Pencahayaan harus memadai karena hal tersebut berpengaruh pada sistem dalam mendeteksi jumlah wajah.
c. Sistem dapat mendeteksi jumlah wajah baik dari jarak dekat (46 cm) maupun dari jarak terjauh (361 cm).
d. Sistem dapat mendeteksi dengan baik, seperti yang terlihat pada tabel hasil pengujian.

Daftar Rujukan

[1] Hidayat N., M.A Rahman, 2015, Cara Cepat Untuk Mendeteksi Keberadaan Wajah Pada Citra Yang Mempunyai Background Kompleks Menggunakan Model Warna YCbCr Dan HSV, *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)*, Volume 2 Nomor 2, Oktober 2015, Halaman 138-142.

[2] Saubari N., 2019, Deteksi Citra Wajah Dengan Metode Haar Feature Selection, *JTIULM*, Volume 04 Nomor 1, April 2019: 7-12.

[3] M. I. Firdaus., I. I. Ridho, 2016, Aplikasi Pendeteksi Wajah Menggunakan Metode Haar, *Media Sains*, Volume 9 Nomor 1, April 2016, ISSN ELEKTRONIK 2355-9136.

[4] Purnamawati S., R. F. Rahmat., dan Santana M., 2015, Aplikasi Pendeteksi Wajah Manusia Untuk Menghitung Jumlah Manusia, *Lentera*, Volume 15, juli 2015.

[5] Syarif M., Wijanarto, 2015, Deteksi Kedipan Mata Dengan Haar Cascade Classifier Dan Contour Untuk Password Login Sistem, *Techno.Com*, Volume 14 Nomor 4, November 2015: 242-249.

[6] D. A. Prasetya, Nurviyanto I., 2012, Deteksi Wajah Metode Viola Jones Pada OpenCV Menggunakan Pemrograman Python, *Simposium Nasional RAPI XI FT UMS* -2012, ISSN : 1412-9612.

[7] Zein A., 2018, Pendeteksi Kantuk Secara Real Time Menggunakan Pustaka OpenCV dan Dlib Python, *Sainstech*, Volume 28 Nomor 2, Juli 2018, ISSN : 1410 - 7104.