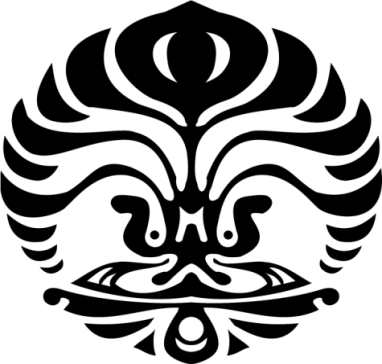
**Proposal Proyek Akhir**

***Digital Signal Processing***

****

***Facial Landmark Detection With Phyton***

|  |  |
| --- | --- |
| Annisa Nuraini | 1606828854 |
| Aditya Bariq  William Yangjaya | 1606889742  1605674321 |

**Departemen Fisika  
Fakultas Matematika Dan Ilmu Pengetahuan Alam  
Universitas Indonesia  
2019**

**DAFTAR ISI**

**DAFTAR ISI i**

**BAB 1 PENDAHULUAN 2**

**1.1.** **Latar Belakang 2**

**1.2.** **Perumusan Masalah 3**

**BAB 2 TINJAUAN PUSTAKA DAN METODE 4**

**2.1.** **Kriteria Evaluasi 4**

**2.2.** **Pendekatan Masalah 4**

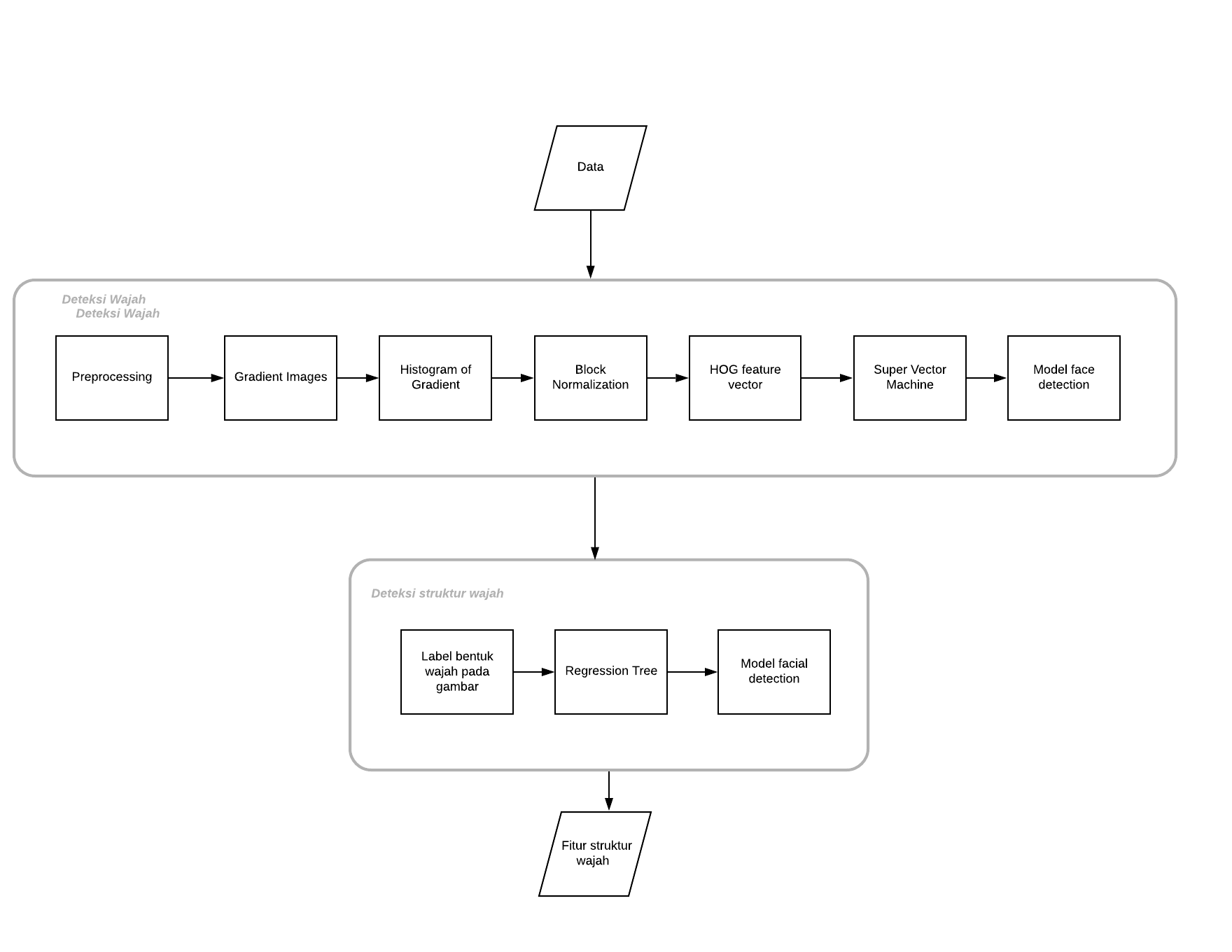
**DAFTAR REFERENSI 7**

**BAB 1  
PENDAHULUAN**

1. **Latar Belakang**

Semakin banyak peneliti yang berfokus pada deteksi landmark wajah *(Facial Landmark Detection)*. Tujuan dari deteksi ini adalah untuk mendeteksi titik-titik kunci di wajah manusia. Misalnya, ujung hidung, alis, sudut mata dan mulut. Deteksi landmark wajah merupakan prasyarat untuk berbagai visi aplikasi komputer. Ia juga merupakan komponen mendasar dalam banyak pekerjaan analisis wajah, seperti inferensi atribut wajah, verifikasi wajah, dan pengenalan wajah.

1. **Perumusan Masalah**



Gambar 1. Pengolahan *facial landmark detection*

Terdapat dua proses dalam melaksanakan proses ini

1. Mendeteksi wajah pada gambar
2. Mendeteksi fitur struktur wajah

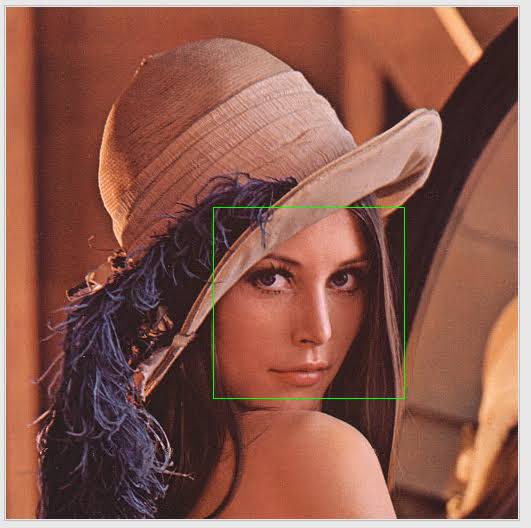
terdapat Penulis berargumen bahwa respon sinyal EEG terdiri dari beberapa *frequency* *band*, yaitu Delta (δ), Theta (θ), Alpha (α), dan Beta (β), dari setiap kondisi untuk setiap pemain, akan berbeda-beda bergantung pada pengalaman pemain. Hasil kuantitatif ini kemudian dapat dianalisis lebih lanjut untuk mengevaluasi *rules* tersebut, seiring dengan bertambahnya pemain maupun pengalaman-pengalaman pemain.

**BAB 2  
TINJAUAN PUSTAKA DAN METODE**

1. **Kriteria Evaluasi**

Kriteria untuk evaluasi yang digunakan adalah *landmark* dapat memprediksi struktur wajah manusia seperti mulut, alis kanan, alis kiri, mata kanan, mata kiri, hidung dan rahang

1. **Pendekatan Masalah**

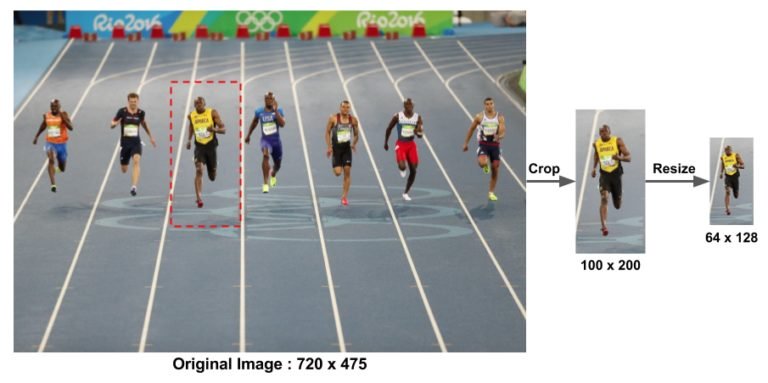


Gambar 2. Deteksi Wajah pada Gambar

Ada berbagai cara untuk mendeteksi wajah pada gambar, pada proyek kali ini akan digunakan metode *Histogram of Oriented Gradients* (HOG) dengan model machine learning *super vector machine* (SVM), model ini dibuat dengan 5 HOG filter, hadap depan, hadap kiri, hadap kanan, hadap depan tetapi memutar kiri, hadap depan tetapi memutar kanan, metode ini dipilih karena merupakan salah satu metode tercepat pada CPU, bekerja cukup baik untuk wajah hadap depan, termasuk model yang ringan, didalam HOG sendiri terdapat beberapa proses sebelum data gambar dapat dilatih dalam model SVM, antara lain:

* Preprocessing

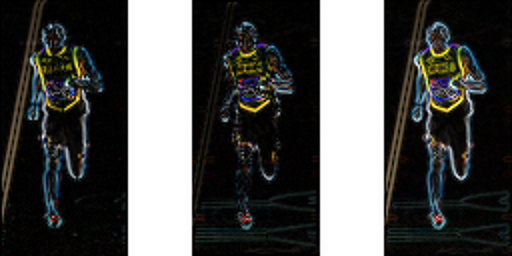
Pada tahap ini gambar perlu *rescale* untuk dianalisis ratio yang dibutuhkan 1:2



Gambar 2.1 *preprocessing*

* Menghitung *gradient image*

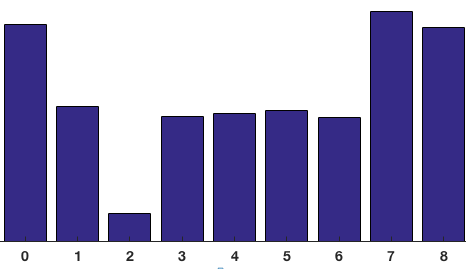
Gambar yang sudah melalui tahap *preprocessing akan* akan difilter dengan horizontal yang merupakan vektor [-1, 0, 1] dan vertikal gradient yang merupakan tranpose dari horizontal gradient, lalu dapat dicari besaran dan arah gradientnya dengan menggunakan rumus:



Gambar 2.2 kiri : nilai mutlak dari gradien vertikal. tengah : nilai mutlak dari gradien horizontal. kanan : besar nilai gradient

Dengan filter gradien akan menghilangkan banyak informasi yang tidak dibutuhkan seperti background tetapi akan menebalkan garis tepi pada objek, sehingga lebih mudah mengenali seuatu objek pada gambar.

* Menghitung *histogram gradient*

Pada tahap ini, gamabr akan dibagi dalam 8x8 seldan menghitung menghitungnya *histogram gradient* yang terdiri dari 9bit dengan cara besar gradient akan diposisikan pada 9bit yang letaknya bergantung pada arah gradien, jika semua digabung maka dapat dibentuk 9 bit histogram. 

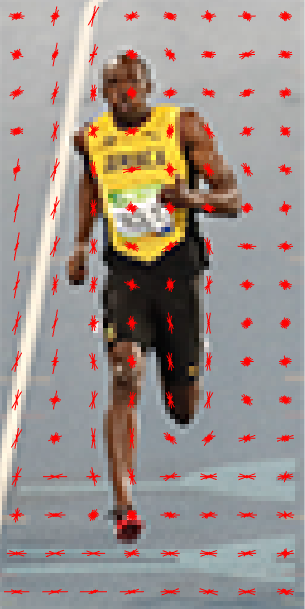
Gambar 2.3 Menitung *histogram gradient*

* Normalisasi

Tahap ini diperlukan untuk mengurangu bias karena pencahayaan pada gambar, menormalisasikan 9x1 histogram akan sama dengan menormalisasikan 3x1 vektor, dengan cara membagi masing-masing komponen vektor dengan kuadrat jumlah lalu diakarkan dari tiap komponen vektor.

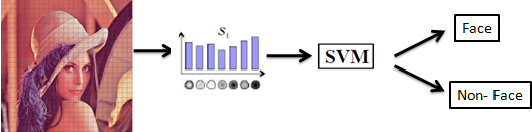
* Menghitung *HOG feature vector*

Vektor-vektor yang telah didapatkan digabungkan menjadi satu vektor.



Gambar 2.4 Visualisasi *Histogram of Oriented Gradients*

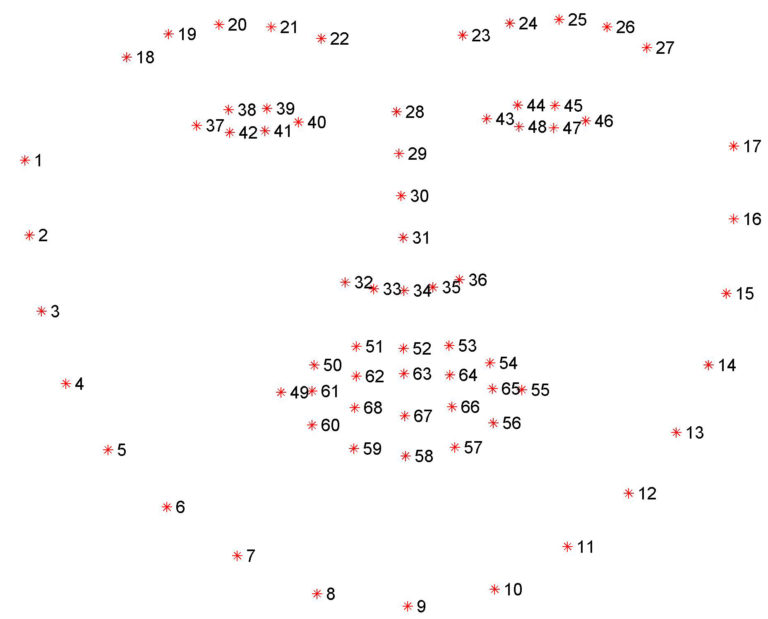
Setelah melakukan HOG gambar dilatih pada metode pembelajaran mesin SVM sehingga didapatkan model yang dapat menentukan yang mana wajah yang mana bukan.

’Gambar 2.5 *Face Detection Method*

Pada tahap kedua yaitu deteksi struktur wajah yang akan dideteksi antara lain bagian:

* Mulut
* Alis Kanan
* Alis Kiri
* Mata Kanan
* Mata Kiri
* Hidung
* Rahan

Metode yang digunakan untuk mendapatkan model deteksi struktur tidak jauh berbeda tetapi kali ini menggunakan *Regression Tree* yang telah dilakukan oleh Kazemi dan Sullivan(2014) dalam paper *One Millisecond Face Alignment with an Ensemble of Regression Trees,* perbedaanya terletak pada bagian melatih model yang perlu menyiapkan data yang dilabel secara manual kordinat spesifik dari daerah muka yang ingin dipresiksi, terdapat 68 kordinat yang diprediksi oleh *paper* tersebut.



Gambar 2.6 36 index yang mewakili struktur wajah manusia

**DAFTAR REFERENSI**

1. Smith M. E., McEvoy L. K., Gevins A. (1999). Neurophysiological indices of strategy development and skill acquisition. Cogn. Brain Res. 7, 389–404. 10.1016/S0926-6410(98)00043-3
2. Anderson J. R., Bothell D., Fincham J. M., Anderson A. R., Poole B., Qin Y. (2011). Brain regions engaged by part- and whole-task performance in a video game: a model-based test of the decomposition hypothesis. J. Cogn. Neurosci. 23, 3983–3997. 10.1162/jocn\_a\_00033
3. Prakash R. S., De Leon A. A., Mourany L., Lee H., Voss M. W., Boot W. R., et al. (2012). Examining neural correlates of skill acquisition in a complex videogame training program. Front. Hum. Neurosci. 6:115. 10.3389/fnhum.2012.00115
4. Palaus, M., Marron, E. M., Viejo-Sobera, R., & Redolar-Ripoll, D. (2017). Neural Basis of Video Gaming: A Systematic Review. Frontiers in Human Neuroscience, 11, 248. http://doi.org/10.3389/fnhum.2017.00248