BAB II
TINJAUAN PUSTAKA

Berikut ini adalah beberapa tinjauan pustaka yang menjadi pokok bahasan dalam pembuatan proposal tugas akhir.

2.1 Microsleep

Microsleep merupakan suatu kondisi tidur pendek berkisar antara 1 sampai 30 detik, dimana orang yang mengalami kondisi ini gagal merespon sensor motorik dan menjadi tidak sadarkan diri [4]. Hal ini terjadi dikarenakan rasa lelah ataupun kurangnya istirahat yang menimbulkan rasa kantuk tak tertahankan sehingga dapat menyebabkan terjadinya kecelakaan.

2.2 Image Processing

Pengolahan citra atau *image processing* merupakan suatu sistem dimana proses dilakukan dengan masukan (*input*) berupa citra (*image*) dan hasilnya (*output*) juga berupa citra (*image*). Pada awalnya pengolahan citra digunakan untuk memperbaiki kualitas citra bergerak sehingga hasil yang ditampilkan sesuai dengan citra masukan namun seiring berkembang dalam berbagai bidang sehingga munculah *computer vision.*[7].

2.3 Face Recognition

Pengenalan wajah atau lebih dikenal dengan *face recognition* merupakan salah satu teknik pengenalan wajah yang sama seperti sidik jari dan retina mata, dimana hasil tangkapan kamera nantinya akan dicocokan dengan foto dan lekuk wajah yang sudah ada di dalam *database. Face recognition* juga termasuk salah satu teknologi biometrik yang telah dipelajari dan dikembangkan oleh para ahli, karena menggunakan algoritma pengenalan wajah untuk membedakan individu yang satu dengan lainnya berdasarkan data yang sudah ada di dalam *database.*[8].

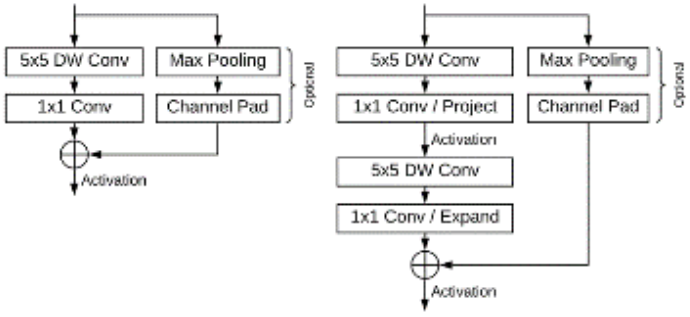
Pengenalan wajah juga dapat digunakan untuk mendeteksi bagaimana kondisi subjek seperti halnya mengantuk ataupun kelelahan, hal ini dengan cara

mengetahui bagaimana kondisi pengendara apakah dalam keadaan mengantuk atau tidak dengan cara menyesuaikan dengan parameter yang sebelumnya telah dibuat lalu diterapkan kepada kondisi di dunia nyata sehingga akan didapati hasil yang sesuai dengan parameter yang sudah ada sebelumnya.

2.4 Algoritma *Blazeface*

Blazeface model ini bergerak dan dibangun pada empat pertimbangan desain penting yaitu memperbesar ukuran bidang reseptif, fitur ekstraksi, skema jangkar atau *anchor scheme, post-processing,* yang mendasari algoritma *blazeface* ini.

Sementara Sebagian besar arsitektur jaringan konvolusi cenderung mendukung model kernel konvolusi *3 x 3* di sepanjang grafik model, sedangkan model *blazeface* mencatat bahwa perhitungan konvolusi ini dapat dipisahkan secara mendalam dan didominasi oleh bagian titik titiknya. Pada tensor masukan *s x s x c depthwise convolution k x k* melibatkan s^2 ck^2 operasi perkalian, sedangkan konvolusi *1 x 1* berikutnya ke *output chanel* d terdiri dari s^2cd, dalam factor d/k^2 dari bagian *depthwise,* dalam praktiknya kita lakukan pemisalan pada iphone x dengan implementasi *Metal Performance Shades,* 3 x 3 depthwise convolution dalam 16 bit *floating* aritmatika membutuhkan 0,07ms untuk 56x56x128 sedangkan konvolusi 1x1 dari 128 sampai 128 saluran adalah 4.3 kali lebih lambat pada 0.3ms, pengamatan ini menyiratkan bahwa meningkatnya ukuran kernel dari bagian depthwise relative lebih murah. Dengan menggunakan kernel 5x5 pada arsitektur *bottlenecks,* dapat meningkatkan ukuran kernel untuk penurunan untuk jumlah total penurunan hambatan yang diperlukan untuk mencapai penerimaan tertentu pada ukuran bidang.



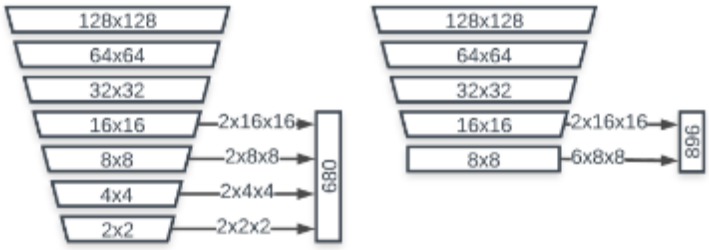
Gambar 2. 1 Blazeblock dan double blazeblock

Fitur ekstraksi ini memiliki contoh spesifik, algoritma *blazeface* berfokus kepada fitur ekstraktor kamera, dengan memperhitungkan rentang skala objek yang lebih kecil dan karenanya memiliki tuntutan komputasi yang lebih rendah, ekstraktor mengambil input RGB 128x128 piksel dan terdiri dari konvolusi 2D diikuti oleh 5 blazebock tunggal dan 6 blazeblock ganda. Dengan kedalaman tensor tertinggi adalah 96 sedangkan resolusi terendah adalah 8x8 sangat berbeda dengan SSD yang mengurangi resolusi hingga 1x1.

Layer/block	Input size	Conv. kernel sizes
Convolution	128×128×3	5×5×3×24 (stride 2)
Single BlazeBlock	64×64×24	5×5×24×1 1×1×24×24
Single BlazeBlock	64×64×24	5×5×24×1 1×1×24×24
Single BlazeBlock	64×64×24	5×5×24×1 (stride 2) 1×1×24×48
Single BlazeBlock	32×32×48	5×5×48×1 1×1×48×48
Single BlazeBlock	32×32×48	5×5×48×1 1×1×48×48
Double BlazeBlock	32×32×48	5×5×48×1 (stride 2) 1×1×48×24 5×5×24×1 1×1×24×96
Double BlazeBlock	16×16×96	5×5×96×1 1×1×96×24 5×5×24×1 1×1×24×96
Double BlazeBlock	16×16×96	5×5×96×1 1×1×96×24 5×5×24×1 1×1×24×96
Double BlazeBlock	16×16×96	5×5×96×1 (stride 2) 1×1×96×24 5×5×24×1 1×1×24×96
Double BlazeBlock	8×8×96	5×5×96×1 1×1×96×24 5×5×24×1 1×1×24×96
Double BlazeBlock	8×8×96	5×5×96×1 1×1×96×24 5×5×24×1 1×1×24×96

Gambar 2. 2 *Blazeface* Fitur Ekstraksi

Model deteksi objek seperti SSD mengandalkan pada kotak pembatas dasar ukuran tetap yang telah ditentukan sebelumnya yang disebut dengan *priors* atau jangkar pada *faster R-CNN.*



Gambar 2. 3 Anchor Computation SSD (Kiri) *Blazeface* (Kanan)

Satu set parameter regresi sebagai offset tengah dan penyesuaian dimensi diprediksi untuk setiap *anchor* atau jangkar. Dan digunakan untuk menyesuaikan jangkar yang telah ditentukan sebelumnya posisi menjadi persegi panjang dengan pembatas yang ketat.

Hal ini merupakan praktik umum untuk menentukan jangkar pada berbagai tingkatan resolusi sesuai dengan rentang skala objek, downsampling yang agresif juga merupakan sarana untuk optimasi sumber komputasional. Model SSD pada umumnya menggunakan prediksi fitur 1x1, 2x2, 4x4, 8x8 dan 16x16 fitur map size namun bagaimanapun *Pooling Pyramid Network (*PPN) menyiratkan bahwa perhitungan tambahan dapat menjadi berlebihan setelah mencapai fitur tertentu pada resolusi peta.

Fitur utama khusus untuk GPU dibandingkan dengan komputasi CPU adalah biaya tetap yang nyata untuk pengiriman tertentu layer komputasi, yang menjadi relative signifikan untuk lapisan dalam resolusi rendah yang melekat pada *CPU Tailored Architecture* yang popular. Sebagai contoh dalam satu percobaan mengamati bahwa dari 4.9ms waktu inferensi MobileNetV1 hanya 3.9ms yang dihabiskan dalam komputasi shader GPU.

2.5 Inverse Kinematics

Inverse kinematics merupakan algoritma yang digunakan untuk menghitung nilai derajat kemiringan pada wajah, algoritma *inverse kinematics* ini menghitung nilai bukaan mulut dan mata serta menentukan derajat kemiringan dagu dan jidat sehingga nantinya akan ditentukan berapa panjang dan sudut yang diperoleh, yang nantinya akan disesuaikan dengan parameter mengantuk yang ada sebelumnya. Dengan menggunakan rumus persamaan matematis sebagai berikut.

$$x,y = r\ \theta \tag{2.1}$$

merupakan rumus persamaan dasar dimana r sebagai radian yang menentukan besaran nilai dari mata dan juga mulut maka akan didapati perhitungan sebagai berikut untuk mulut.

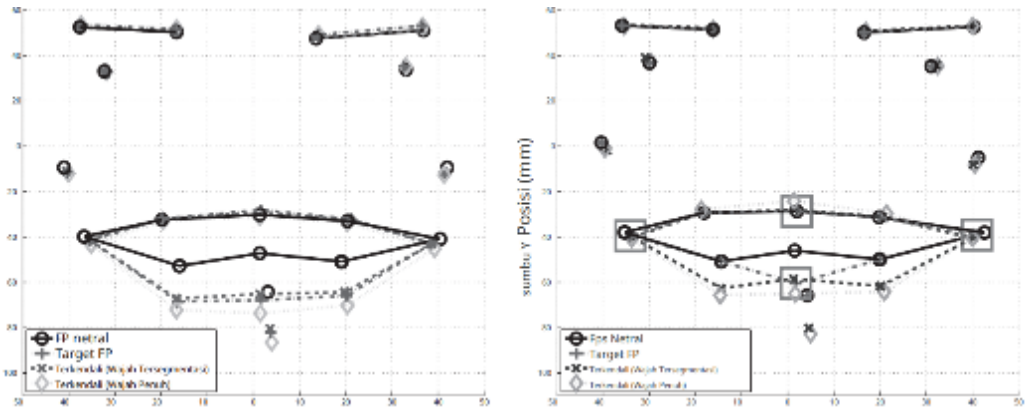
$$r_{mata} = \sqrt{x_1 + y_1} \tag{2.2}$$

$$r_{mulut} = \sqrt{x_2 + y_2} \tag{2.3}$$

selanjutnya dilakukan perhitungan untuk kemiringan kepala dimana kemiringan akan ditentukan melalui dagu dan juga jidat sebagai titik utama yang nantinya akan dilakukan perhitungan kemiringan, sehingga didapati rumus sebagai berikut.

$$\theta = arc\ cos \frac{y}{x} \times rad \quad (2.4)$$

$$\theta = arc\ cos \left(\frac{y}{x} \times \frac{\varphi}{180} \right) \quad (2.5)$$



Gambar 2. 4 Titik Posisi Wajah

2.6 SVM

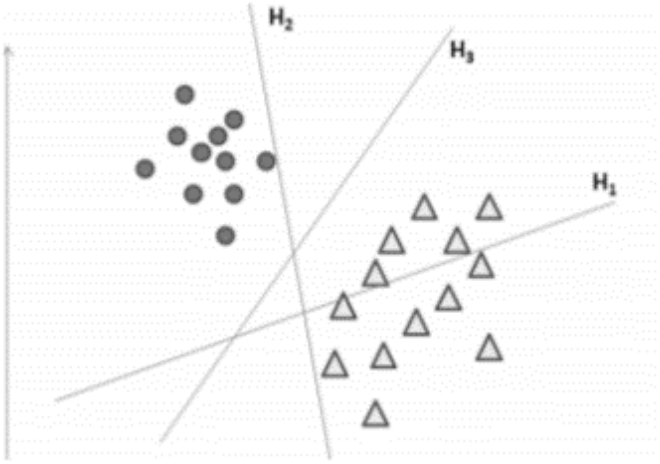
Support vector machine (SVM) yaitu metode klasifikasi yang diperkenalkan pertama kali oleh Vapnik pada tahun 1998. Pada dasarnya, metode ini bekerja dengan cara mendefinisikan batas antara dua kelas dengan jarak maksimal dari data yang terdekat. Algoritma ini diterapkan pada feature space. Dimensi tinggi feature space meningkatkan kesulitan untuk menghitung produk skalar dari feature space yang ada. Fungsi kernel digunakan untuk menghitung produk skalar ini. Dengan menggunakan fungsi kernel tidak memerlukan perhitungan *feature space* secara eksplisit[9].

Metode ini adalah algoritma yang bekerja menggunakan pemetaan non-linier kemudian mengubah data *training* asli ke dimensi yang lebih tinggi, kemudian dimensi baru ini akan mencari *hyperplane* untuk melakukan pemisahan linier. Sehingga kelas tersebut dipisahkan dengan *hyperplane*, *hyperplane* tersebut dapat diperoleh dengan menggunakan support vector dan margin pada SVM ini hanya menyimpan Sebagian kecil dari data latih yang kemudian digunakan pada saat klasifikasi. Tentunya ini menjadi kelebihan dari SVM karena tidak semua data latih dimasukan kedalam setiap iterasi pelatihannya.

Pada dasarnya karakteristik dari SVM terbagi menjadi beberapa bagian diantaranya adalah sebagai berikut.

1. *Support Vector Machine* merupakan *linear classifier.*
2. Untuk memperoleh pola yang dapat dikenali, dilakukanlah transformasi data input ke ruang yang berdimensi lebih tinggi.
3. SVM menerapkan strategi *structural risk minimization.*
4. SVM pada dasarnya hanya mampu menangani dua klasifikasi kelas.

Pada gambar dibawah ini dapat dilihat bahwa H1 tidak memisahkan dua kelas, H2 terpisah tetapi dengan *margin* yang sangat kecil antar kelas dan H3 memisahkan kedua kelas dengan *margin* yang jauh lebih baik daripada H2. *Margin* adalah jarak antara *hyperplane* tersebut dengan data terdekat dari masing-masing kelas. Apabila terdapat *hyperplane* demikian maka batas pemisahan terbaik antara dua kelas disebut dengan *hyperplane margin* maksimum dan *classifier linier* yang demikian dikenal dengan *classifier linier* maksimum.



Gambar 2. 5 Pemisahan *Hyperplanes*

Multiclass SVM digunakan untuk melakukan proses penetapan label ke beberapa elemen himpunan terbatas berdasarkan set SVM linier atau non linier dasar. Caranya yaitu dengan mengurangi masalah multi kelas menjadi masalah biner.

$$\frac{\partial lp}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N} \alpha_i y_i , x_i \quad (2.6)$$

$$\frac{\partial lp}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N} \alpha_i y_i , = 0 \quad (2.7)$$

$$\alpha i \, [yi \, (w. \, xi + b) - 1] \quad (2.8)$$

$$\alpha i \geq 0, i = 1,2, \ldots .N \quad (2.9)$$

Selanjutnya sebagai klasifikasi biner diasumsikan untuk menghasilkan suatu fungsi keluaran yang memberikan nilai relative besar untuk contoh yang termasuk ke dalam kelas positif dan nilai kecil yang termasuk ke dalam nilai negatif, ada dua metode umum yang digunakan pada pengklasifikasian dimana pengklasifikasian dilatih untuk membedakan salah satu label terhadap lainnya atau dikenal sebagai *one against all* dan pasangan kelas yang dikenal sebagai *one against one.*

Klasifikasi untuk *one against all* adalah pemenang mengambil semua strategi yang dimana *classifier* dengan fungsi keluaran tertinggi adalah kelas yang dimaksud, sedangkan untuk *one against one* dilakukan dengan strategi pemungutan suara *max-win* yang dimana setiap *classifier* memberikan pengumpamaan ke salah satu dari dua kelas. Suara untuk kelas dengan lebih banyak suara yang dimaksud [10].

2.7 Python dan OpenCV

Python merupakan sebuah bahasa pemrograman yang bersifat *open source*. Python dibuat oleh Guido van Rossum pertama kali di Centrum Wiskunde & Informatica (CWI) di Belanda pada awal tahun 1990-an. Python dapat dijalankan pada berbagai sistem operasi. Kelebihan lain dari pemrograman Python adalah bentuk program yang sederhana sehingga mudah untuk dipelajari [11].

OpenCV adalah sebuah pustaka yang bersifat open source yang dikembangkan oleh intel yang fokus untuk menyederhanakan pemrograman terkait dengan pengolahan citra digital. OpenCV memiliki banyak fitur terkait visi komputer (*computer vision*) antara lain : pengenalan wajah, deteksi wajah, kalman filtering, dan berbagai jenis metoda AI (*Artificial Intelligence*). OpenCV dapat bekerja di berbagai bahasa pemrograman, seperti C, C++, Java, Python, dan juga mendukung berbagai platform sistem operasi seperti Windows, Linux, Mac OS, iOS dan Android [11].

2.8 Mediapipe

Mediapipe merupakan salah satu framework yang digunakan untuk membangun *machine learning* yang dapat berjalan lintas *platform* seperti pada

perangkat *mobile* dan desktop tentunya mediapipe sangat memudahkan dalam membangun *machine learning,* dengan menggunakan *multithreading* dan *GPU acceleration* membuat mediapipe dapat berjalan dengan cepat saat memproses data [12]. Mediapipe memiliki banyak solusi *machine learning* yang digunakan, penulis menggunakan *face detection* menggunakan *face mesh* yang berbasis algoritma *blazeface* dalam penelitian kali ini untuk menentukan letak wajah sehingga program dapat berjalan secara *real time* dan dengan komputasi yang rendah.

2.9 Kecerdasan Buatan

Kecerdasan buatan merupakan kecerdasan yang dibuat untuk menyamai kecerdasan manusia dengan kata lain Kecerdasan buatan berarti kemampuan sebuah mesin untuk meniru perilaku manusia untuk menangani masalah tertentu [13]. Kecerdasan buatan sendiri adalah teknologi yang memerlukan data untuk dapat bekerja secara baik, layaknya manusia yang memerlukan pengetahuan dasar begitu juga kecerdasan buatan, oleh karena itu diperlukannya proses *train* atau latih pada kecerdasan buatan sehingga selanjutnya data dapat diuji. Kecerdasan buatan ini pada akhirnya dapat diterapkan ke dalam beberapa bidang, baik kesehatan, keamanan, pada kendaraan dan lainnya.

2.10 Arduino

Arduino Merupakan papan elektronik berbasis mikrokontroler ATMega yang memenuhi sistem minimum mikrokontroler agar dapat bekerja secara mandiri (*standalone controller*). Komponen utama didalam papan Arduino adalah sebuah mikrokontroler 8 bit dengan merk ATMega yang dibuat oleh Atmel corporation. Berbagai papan Arduino menggunakan tipe Atmega yang berbeda -beda tergantung dari spesifikasinya, sebagai contoh Arduino Uno menggunakan ATmega328 sedangkan Arduino Mega 2560 yang lebih canggih menggunakan ATmega2560[14].