SKRIPSI

SISTEM PRESENSI MAHASISWA BERBASIS MULTI-FACE RECOGNITION DENGAN HISTOGRAM OF ORIENTED GRADIENTS

Disusun dan diajukan oleh:

MUHAMMAD ZUL FAHMI SADRAH

D121171316

DEPARTEMEN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HASANUDDIN

2022

LEMBAR PENGESAHAN SKRIPSI

SISTEM PRESENSI MAHASISWA BERBASIS MULTI-FACE RECOGNITION DENGAN HISTOGRAM OF ORIENTED GRADIENTS

Disusun dan diajukan oleh

MUHAMMAD ZUL FAHMI SADRAH

D121171316

Telah dipertahankan di hadapan Panitia Ujian yang dibentuk dalam rangka Penyelesaian Studi Program Sarjana Program Studi Teknik Informatika Fakultas Teknik Universitas Hasanuddin pada tanggal 26 Agustus 2022 dan dinyatakan telah memenuhi syarat kelulusan.

Menyetujui,

Pembimbing Utama,	Pembimbing Pendamping,
Dr. Ir. Ingrid Nurtanio, MT	Iqra' Aswad, S.T., M.T
Nip. 196108131988112001	Nip. 199011282019043001

Ketua Program Studi,

Dr. Indrabayu, S.T., M.T., M.Bus.Sys

Nip. 19750716 200212 1 004

PERNYATAAN KEASLIAN

Yang bertanda tangan di bawah ini:

Nama : Muhammad Zul Fahmi Sadrah

NIM : D121171316

Departemen : Teknik Informatika

Jenjang : S1

Menyatakan dengan ini karya tulisan saya berjudul:

SISTEM PRESENSI MAHASISWA BERBASIS MULTI-FACE RECOGNITION DENGAN HISTOGRAM OF ORIENTED GRADIENTS

Adalah karya tulisan saya sendiri dan bukan merupakan pengambilalihan tulisan orang lain bahwa skripsi yang saya tulis ini benar-benar merupakan hasil karya saya sendiri.

Apabila di kemudian hari terbukti atau dapat dibuktikan bahwa sebagian atau keseluruhan skripsi ini hasil karya orang lain, maka saya bersedia menerima sanksi atas perbuatan tersebut.

Gowa, 25 Agustus 2022

Yang menyatakan,

Muhammad Zul Fahmi Sadrah

KATA PENGANTAR

Segala puji dan syukur kami panjatkan ke hadirat Allah SWT. Tuhan Yang Maha Esa yang dengan limpahan rahmat dan hidayah-Nya sehingga tugas akhir dengan judul "Sistem Presensi Mahasiswa Berbasis *Multi-Face Recognition* dengan *Histogram of Oriented Gradients*" ini dapat diselesaikan sebagai salah satu syarat dalam menyelesaikan jenjang Strata-1 pada Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin.

Dalam penyusunan penelitian ini disajikan hasil penelitian terkait judul yang telah diangkat dan telah melalui proses pencarian dari berbagai sumber baik jurnal penelitian, prosiding pada seminar-seminar nasional/internasional, buku maupun dari situs-situs di internet.

Penulis menyadari bahwa tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai dengan masa penyusunan Tugas Akhir, sangatlah sulit untuk menyelesaikan Tugas Akhir ini. Oleh karena itu, penulis berterima kasih kepada:

1) Kedua Orang tua penulis yang tidak pernah lelah dalam mendidik dan memberikan dukungan, doa, serta semangat kepada penulis.
2) Ibu Dr. Ir. Ingrid Nurtanio, M.T., selaku pembimbing utama dan Bapak Iqra Aswad, S.T., M.T., selaku pembimbing pendamping yang selalu menyediakan waktu, tenaga, pikiran dan perhatian yang luar biasa untuk mengarahkan penulis dalam penyusunan Tugas Akhir.
3) Bapak Dr. Indrabayu, S.T., M.T., M.Bus.Sys. dan Bapak Dr. Adnan, S.T., M.T. selaku penguji yang telah memberikan banyak masukan konstruktif agar Tugas Akhir ini dapat memberikan manfaat sesuai dengan yang diharapkan.
4) Bapak Dr. Indrabayu, S.T., M.T., M.Bus.Sys. selaku Ketua Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas segala bimbingan dan dukungannya selama masa perkuliahan.
5) Ibu Anugrayani Bustamin, S.T., M.T., selaku Dosen Pembimbing Akademik penulis yang selalu membimbing dan menyediakan waktu, tenaga, dan perhatiannya selama masa perkuliahan.

6) Segenap Dosen dan Staf Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin yang telah banyak membantu semasa perkuliahan dan dalam penyelesaian tugas akhir.
7) Seluruh teman-teman angkatan 2017 (RECOGN17ER) Departemen Teknik Informatika Fakultas Teknik Universitas Hasanuddin atas dukungan, bantuan, semangat, serta pengalamannya yang diberikan selama ini.
8) Para sahabat, teman-teman yang telah memberikan begitu banyak bantuan, semangat, inspirasi, ilmu dan hiburan selama penelitian, pengambilan data dan diskusi *progress* penyusunan Tugas Akhir.
9) Orang-orang berpengaruh lainnya yang tanpa sadar telah menjadi inspirasi penulis.

Akhir kata, penulis menyadari bahwa tugas akhir ini masih jauh dari kata sempurna, oleh karenanya diharapkan segala bentuk saran serta masukan yang membangun dari berbagai pihak. Semoga tugas akhir ini dapat memberikan sumbangsih dan manfaat besar bagi kepentingan bersama. Aamiin.

Gowa, 26 Agustus 2022

Penulis

ABSTRAK

Presensi atau kehadiran mahasiswa merupakan hal yang penting pada pelaksanaan perkuliahan. Beberapa kampus saat ini masih menerapkan metode pengambilan presensi konvensional yang mencatat kehadiran mahasiswa menggunakan media kertas. Metode tersebut tidak efisien karena membutuhkan banyak waktu, mudah dimanipulasi, dan sulit direkapitulasi. Saat ini telah banyak alternatif metode presensi otomatis dengan biaya yang relatif mahal dan akurasi yang beragam. Penelitian ini memberikan alternatif lain dengan membuat sistem presensi mahasiswa berbasis *multi-face recognition* dengan metode *Histogram of Oriented Gradients* (HOG) yang digunakan untuk mengekstraksi fitur wajah. Selain itu, sistem ini juga menerapkan metode *Multi-Task Cascaded Convolutional Neural Network* (MTCNN) untuk pendeteksian wajah dan *Support Vector Machine* (SVM) untuk proses klasifikasi fitur wajah yang telah diekstraksi. Sistem ini dibangun pada *platform website* dengan pengoptimalan pada perangkat *smartphone* dengan menerapkan teknologi *Single Page Application* (SPA). Dengan adanya sistem ini, dosen dapat mengambil presensi mahasiswa dengan memotret seluruh mahasiswa yang hadir di ruangan kelas menggunakan kamera *smartphone*-nya. Data latih pada penelitian ini menggunakan 500 citra wajah dari 25 mahasiswa yang diambil dari lima sisi dan empat ekspresi wajah yang berbeda pada masing-masing sisinya. Untuk pengujian sistem, digunakan 10 citra yang pada setiap citra terdapat 7 mahasiswa dengan jarak 1 hingga 5 meter serta pencahayaan ruangan yang terang dan redup. Hasil pengujian sistem menunjukkan akurasi tertinggi sebesar 95,71% pada jarak dan pencahayaan yang bervariasi. Sistem juga dapat mengenali wajah bermasker dengan akurasi 82,35%.

Kata Kunci **:** sistem presensi, pengenalan wajah, HOG, MTCNN, SVM

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi yang sangat pesat saat ini memberikan banyak kemudahan dan efisiensi pada aktivitas manusia. Perkembangan tersebut tentunya diharapkan dapat diterapkan ke seluruh bidang, termasuk bidang administrasi akademik di perguruan tinggi. Salah satu aktivitas yang cukup ruwet pada bidang tersebut adalah presensi mahasiswa.

Presensi merupakan salah satu tolak ukur utama untuk menjunjung tinggi nilai kedisiplinan seseorang dalam suatu organisasi/instansi (Rosyadi, 2015). Sistem presensi mahasiswa konvensional yang banyak digunakan saat ini masih terbilang kurang efisien. Hal ini terlihat dari masih banyaknya penggunaan kertas dan pemasukan data presensi yang masih manual oleh admin sehingga menguras banyak waktu dan tenaga. Selain itu, sistem yang diterapkan saat ini juga memungkinkan mahasiswa untuk memalsukan presensinya dengan mudah yang umumnya dikenal sebagai fenomena “titip absen”. Berdasarkan penelitian yang dilakukan oleh Sulalah pada tahun 2018 di UIN Sunan Ampel, 71% mahasiswa pernah melakukan titip absen.

Dengan demikian, dibutuhkan suatu sistem presensi yang praktis, aman dan efisien. Salah satu solusi yang dapat diterapkan adalah membangun sistem presensi yang menggunakan teknologi pengenalan wajah (*face recognition)* dengan metode *Histogram of Oriented Gradients* (HOG). HOG merupakan metode ekstraksi fitur

dengan menghitung nilai gradien dalam daerah tertentu pada citra (Afifah, 2017). Sistem ini akan dibangun menggunakan *platform website* serta dengan arsitektur REST API. Selain itu, akan diterapkan juga teknologi *Single Page Application* (SPA), yaitu teknologi pengembangan *website* yang memuat keseluruhan *resources* pada permintaan awal dan komponen-komponen halamannya digantikan oleh komponen lain berdasarkan interaksi yang dilakukan pengguna (Jadhav et al., 2015).

Dengan adanya sistem presensi ini, pada saat suatu pertemuan mata kuliah sedang berlangsung, dosen cukup mengambil citra seluruh mahasiswa yang hadir dengan menggunakan kamera *smartphone*-nya. Setelah itu, sistem akan mengenali seluruh wajah mahasiswa yang terdeteksi pada citra tersebut. Kemudian, seluruh wajah mahasiswa yang berhasil dikenali oleh sistem akan dinyatakan hadir pada pertemuan mata kuliah tersebut.

Penelitian serupa dilakukan oleh Arceo et al. (2020) yang membuat sistem presensi mahasiswa untuk perkuliahan di ruangan kelas. Metode *Histogram of Oriented Gradients* (HOG) dan *Support Vector Machine* (SVM) digunakan untuk menentukan pengaruh pencahayaan, orientasi wajah mahasiswa, dan jarak mereka dengan kamera pada proses pendeteksian dan pengenalan wajah. Hasil penelitian ini menunjukkan akurasi 95,65% dan dapat mendeteksi dan mengenali hingga 37 mahasiswa pada level pencahayaan ruangan di atas 217,39 *lux* dan wajah menghadap lurus ke kamera.

Penelitian lainnya dilakukan oleh Nyein (2019) yang membuat sistem presensi mahasiswa menggunakan FaceNet untuk ekstraksi fitur wajah dan SVM

untuk mengklasifikasikan data hasil ekstraksi FaceNet. Penelitian tersebut berfokus untuk membandingkan akurasi antara FaceNet dengan VGG16 dalam hal *multi-face recognition* menggunakan *dataset* yang sama. Dari penelitian tersebut, diperoleh hasil yang menunjukkan bahwa metode yang diusulkan berhasil mencapai akurasi 99,6% serta lebih baik dari metode VGG16.

Arafah et al. (2019) melakukan penelitian serupa dengan membuat sistem pengenalan wajah pada tempat inspeksi penumpang pesawat di bandara. Tujuan dari penelitian ini adalah untuk menentukan jarak terbaik penempatan CCTV untuk mengenali wajah penumpang. Data latih yang digunakan diambil dari lima sisi serta sedangkan data uji diambil dari *frame* video CCTV. Penelitian ini menggunakan metode HOG untuk ekstraksi fitur wajah dan *Multi-class Support Vector Machine* (MSVM) untuk tahap klasifikasi. Hasil penelitian ini menunjukkan akurasi 86,76% pada CCTV dengan jarak 300 cm dan ketinggian 250 cm.

Berdasarkan penelitian di atas, peneliti akan mengembangkan sistem presensi mahasiswa yang telah dilakukan peneliti sebelumnya. Sistem yang dikembangkan akan menggunakan HOG untuk mengenali wajah, MTCNN untuk mendeteksi wajah, dan SVM untuk klasifikasi wajah. Pengambilan presensi mahasiswa dilakukan dengan mengambil citra wajah mahasiswa menggunakan kamera dari perangkat *smartphone* dosen untuk dikenali dan dinyatakan hadir. Sistem ini akan dibangun pada *platform website* agar dapat berjalan di berbagai perangkat dan sistem operasi yang berbeda. Selain itu, s*istem* ini dirancang dengan pengoptimalan pada tampilan antarmuka untuk perangkat *smartphone*. Sistem ini juga akan menerapkan teknologi SPA agar sistem tidak perlu memuat ulang halaman pada

setiap aksi yang dilakukan pengguna sehingga pengguna memperoleh pengalaman navigasi antar halaman yang lebih baik.

Dengan adanya aplikasi ini, diharapkan sistem presensi mahasiswa di perguruan tinggi dapat menjadi lebih praktis, aman dari manipulasi, dan efisien. Selain itu, mahasiswa juga diharapkan dapat menjunjung tinggi nilai kedisiplinan dan kejujuran. Dengan demikian, akan tercipta aktivitas akademik yang lancar dan generasi muda yang unggul untuk Indonesia maju.

1.2 Rumusan Masalah

Berdasarkan latar belakang, maka rumusan masalah pada tugas akhir ini adalah sebagai berikut:

1. Bagaimana rancang bangun sistem presensi mahasiswa berbasis *multi-face recognition* dengan HOG?
2. Bagaimana tingkat akurasi dari sistem presensi mahasiswa berbasis *multi-face recognition* dengan HOG?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah sebagai berikut:

1. Merancang dan membangun sistem presensi mahasiswa berbasis *multi-face recognition* dengan HOG.
2. Menunjukkan keakuratan dari sistem presensi mahasiswa berbasis *multi-face recognition* dengan HOG.

1.4 Manfaat Penelitian

Adapun manfaat yang dapat diperoleh dari penelitian ini adalah sebagai berikut:

1. Manfaat ilmiah, yaitu untuk memberikan sumbangan pemikiran atau menambah informasi mengenai perancangan sistem presensi mahasiswa.
2. Manfaat praktis, yaitu untuk memberikan alternatif sistem presensi mahasiswa yang dapat diterapkan di perguruan tinggi.

1.5 Batasan Masalah

Batasan Masalah dari penelitian ini adalah sebagai berikut:

1. Sistem ini menargetkan presensi untuk perkuliahan di kampus, bukan kuliah daring.
2. *Input* dari proses pengenalan wajah untuk presensi mahasiswa adalah citra, bukan video.
3. Jarak antar dosen dan mahasiswa saat pengambilan citra yang digunakan untuk pengujian sistem adalah 1 hingga 5 meter.
4. Sistem dibangun pada *platform website* dengan pengoptimalan pada *mobile view* dan diuji pada *browser Google Chrome*.

1.6 Sistematika Penulisan

Untuk memberikan gambaran singkat mengenai isi tulisan secara keseluruhan, maka akan diuraikan beberapa tahapan dari penulisan secara sistematis, yaitu:

BAB I PENDAHULUAN

Bab ini menguraikan secara umum mengenai hal yang menyangkut latar belakang, perumusan masalah dan batasan masalah, tujuan, manfaat, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi teori-teori tentang hal-hal yang berhubungan dengan visi komputer, pemrosesan citra dan metode yang digunakan.

BAB III METODOLOGI PENELITIAN

Bab ini berisi tentang tahapan penelitian, instrumen penelitian, perencanaan dan penerapan algoritma serta teknik pengolahan data.

BAB IV HASIL DAN PEMBAHASAN

Bab ini berisi tentang hasil pengolahan data serta pembahasan yang disertai tabel hasil penelitian.

BAB V PENUTUP

Bab ini berisi tentang kesimpulan yang didapatkan berdasarkan hasil penelitian yang telah dilakukan serta saran-saran untuk pengembangan lebih lanjut.

BAB II

TINJAUAN PUSTAKA

2.1 Presensi

Adanya kesalahpahaman penggunaan istilah absensi dan presensi di masyarakat membuat penulis merasa perlu untuk menjelaskan perbedaan dari kedua istilah tersebut.

Menurut Kamus Besar Bahasa Indonesia (KBBI), presensi adalah kehadiran. Kehadiran adalah perihal hadir; adanya (seseorang, sekumpulan orang) pada suatu tempat. Dengan demikian, presensi bisa diartikan sebagai suatu kegiatan yang dilakukan oleh suatu organisasi/instansi sebagai standar untuk menilai kedisiplinan anggotanya berdasarkan kuantitas kehadiran.

Presensi berbeda dengan absensi yang berarti ketidakhadiran. Absensi adalah suatu cara yang dilakukan untuk menghitung jumlah ketidakhadiran anggota. Namun, pada kenyataannya sering kali istilah absensi digunakan sebagai standar kuantitas kehadiran.

Berdasarkan penjelasan di atas, dapat disimpulkan bahwa presensi merupakan istilah yang tepat untuk digunakan dalam pencatatan kehadiran anggota dari suatu organisasi atau instansi.

2.2 Visi Komputer

Visi Komputer merupakan bidang yang mendalami metode, teknik dan teknologi untuk memperoleh, memproses dan memahami citra (Fernández, 2014). Visi komputer mencoba meniru cara kerja sistem visual manusia, bagaimana

manusia dapat melihat objek dengan mata, kemudian citra objek diteruskan ke otak untuk diinterpretasi sehingga manusia dapat mengetahui objek apa yang tampak dalam pandangan mata dan selanjutnya hasil interpretasi digunakan untuk pengambilan keputusan. Beberapa aplikasi dari visi komputer adalah sebagai berikut (Szeliski, 2010).

- *Optical Character Recognition* (OCR): sering digunakan untuk mengidentifikasi citra huruf untuk kemudian diubah ke dalam bentuk *file* tulisan, seperti pengenalan otomatis nomor plat atau *Automatic Number Plate Recognition* (ANPR).
- *Medical Imaging*: mencatat atau merekam citra untuk tujuan klinis atau melakukan studi jangka panjang dari morfologi citra bagian tubuh manusia.
- *Automotive safety*: mendeteksi rintangan yang tak terduga seperti pejalan kaki di jalan raya.
- *Surveillance*: pemantauan untuk penyusup dalam sebuah gedung atau perkantoran.
- *Face detection*: meningkatkan fokus kamera serta mencari gambar yang lebih relevan.

Visi komputer adalah ilmu yang mempelajari bagaimana komputer dapat mengenali objek yang akan diamati atau diobservasi. Hal ini dilakukan untuk dapat meniru visualisasi dari manusia yang diaplikasikan ke dalam komputer. Visi komputer merupakan kombinasi dari pengolahan citra (*image processing*) dan pengenalan pola (*pattern recognition*).

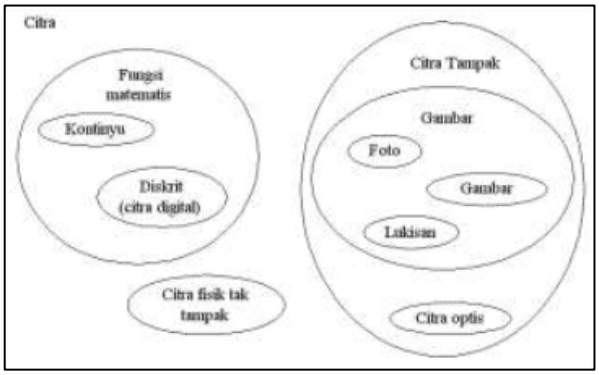
2.2.1 Pengolahan Citra

Pengolahan citra merupakan ilmu yang mempelajari bagaimana memperbaiki kualitas citra agar mendapatkan hasil citra yang baik dan mudah dikenali oleh manusia atau mesin. Pengolahan citra memiliki keterkaitan yang sangat erat dengan disiplin ilmu yang lain, jika sebuah disiplin ilmu dinyatakan dengan bentuk proses suatu *input* menjadi *output*, maka pengolahan citra memiliki *input* berupa citra serta *output* juga berupa citra (Yogi, 2014).

1) Citra

Citra adalah suatu representasi, kemiripan, atau imitasi dari suatu objek atau benda. Setiap citra mempunyai beberapa karakteristik, antara lain ukuran citra, resolusi, dan format nilainya. Umumnya citra berbentuk persegi panjang yang memiliki lebar dan tinggi tertentu (Rinaldi, 2004).

Ukuran ini biasanya dinyatakan dalam banyaknya titik atau piksel, sehingga ukuran citra selalu bernilai bulat. Ukuran citra dapat juga dinyatakan secara fisik dalam satuan panjang. Dalam hal ini tentu saja harus ada hubungan antara ukuran titik penyusunan citra dengan satuan panjang. Hal tersebut dinyatakan dengan resolusi yang merupakan ukuran banyaknya titik untuk setiap satuan panjang. Biasanya satuan yang digunakan adalah dpi. Makin besar resolusi makin banyak titik yang terkandung dalam citra dengan ukuran fisik yang sama, sehingga hal ini memberikan efek pemampatan citra menjadi semakin halus.



Gambar 2.1. Pengelompokan jenis-jenis citra

Citra digital mengandung sejumlah elemen-elemen dasar. Elemen-elemen dasar yang paling penting diuraikan sebagai berikut (Aan dan Angki, 2010).

a) Kecerahan (*Brightness*) merupakan intensitas cahaya rata-rata dari suatu area yang melingkupinya.
b) Kontras (*Contrast*) merupakan sebaran terang (*lightness*) dan gelap (*darkness*) di dalam sebuah citra. Citra dengan kontras rendah komposisi citranya sebagian besar terang atau sebagian besar gelap. Citra dengan kontras yang baik, komposisi gelap dan terangnya, tersebar merata.
c) Kontur (*Contour*) merupakan keadaan yang ditimbulkan oleh perubahan intensitas pada piksel-piksel tetangga, sehingga dapat dideteksi tepi objek di dalam citra.
d) Warna (*Color*) merupakan persepsi yang dirasakan oleh sistem visual manusia terhadap panjang gelombang cahaya yang dipantulkan oleh objek. Warna-warna yang dapat ditangkap oleh mata manusia merupakan kombinasi cahaya dengan panjang berbeda. Kombinasi yang memberikan rentang warna paling lebar adalah *red* (R), *green* (G) dan *blue* (B).

e) Bentuk (*Shape*) merupakan properti intrinsik dari objek tiga dimensi, dengan pengertian bahwa bentuk merupakan properti intrinsik utama untuk visual manusia. Umumnya citra yang dibentuk oleh manusia merupakan 2D, sedangkan objek yang dilihat adalah 3D.

f) Tekstur (*Texture*) merupakan distribusi spasial dari derajat keabuan di dalam sekumpulan piksel-piksel yang bertetangga.

Berdasarkan cakupan operasi yang dilakukan terhadap citra, operasi pengolahan citra dikategorikan sebagai berikut (Ginting, 2004).

a) Operasi titik, yaitu operasi yang dilakukan terhadap setiap piksel pada citra yang keluarannya hanya ditentukan oleh nilai piksel itu sendiri.

b) Operasi area, yaitu operasi yang dilakukan terhadap setiap piksel pada citra yang keluarannya dipengaruhi oleh piksel tersebut dan piksel lainnya dalam suatu daerah tertentu. Salah satu contoh dari operasi berbasis area adalah operasi ketetanggaan yang nilai keluaran dari operasi tersebut ditentukan oleh nilai piksel-piksel yang memiliki hubungan ketetanggaan dengan piksel yang sedang diolah.

2) Citra *Grayscale*

Sesuai dengan nama yang melekat, citra jenis ini menangani gradasi warna hitam dan putih, yang menghasilkan efek warna abu-abu. Intensitas berkisar antara 0 sampai dengan 255. Nilai 0 menyatakan hitam dan 255 menyatakan putih. Citra dalam komputer tidak lebih dari sekumpulan sejumlah triplet terdiri atas variasi tingkat keterangan dari elemen *Red* (R), *Green* (G), *Blue* (B). Angka RGB ini yang sering kali disebut dengan *color values*. Pada format .bmp citra setiap piksel pada

citra direpresentasikan dengan 24 bit, 8 bit untuk R, 8 bit untuk G, dan 8 bit untuk B. *Grayscale* adalah teknik yang digunakan untuk mengubah citra berwarna (RGB) menjadi bentuk *grayscale* atau tingkat keabuan (dari hitam ke putih). Dengan pengubahan ini matriks penyusun citra yang sebelumnya 3 matriks akan berubah menjadi 1 matriks saja.

3) *Thresholding*

Thresholding atau binerisasi merupakan proses mengonversi citra *grayscale* ke dalam bentuk citra biner. Citra biner sesuai namanya adalah citra yang hanya tersusun atas dua nilai yaitu 0 atau 1. Jika piksel citra nilainya di atas nilai threshold maka piksel tersebut akan diubah ke warna putih dan nilainya menjadi 1. Sebaliknya jika nilai piksel citra berada di bawah *threshold* maka citra akan diubah ke warna hitam dan nilainya menjadi 0 (Prasetyo, 2011).

Salah satu cara untuk mengekstrak objek dari *background* adalah dengan memilih *threshold T* yang membagi mode-mode ini. Kemudian sembarang titik (x,y) untuk di mana $f(x,y) \geq$ disebut *object point*. Sedangkan yang lain disebut *background point*. Dengan kata lain, citra yang di- *threshold* $g(x,y)$ didefinisikan sebagai berikut (Prasetyo, 2011).

$$g(x,y) = \begin{cases} 1 \; if \; f(x,y) \geq T \\ 0 \; if \; f(x,y) < T \end{cases} \qquad (2.1)$$

Di mana, piksel yang diberi 1 berkaitan dengan objek sedangkan piksel yang diberi nilai 0 berkaitan dengan *background*.

2.2.2 Pengenalan Pola

Pengenalan pola merupakan proses pengelompokan data numerik dan simbolik (termasuk citra) secara otomatis, yang bertujuan untuk dapat diidentifikasi objek pada citra. Pola adalah entitas yang terdefinisi dan dapat diidentifikasi melalui ciri-cirinya. Ciri-ciri tersebut digunakan untuk membedakan suatu pola dengan pola lainnya. Ciri yang bagus adalah ciri yang memiliki daya pembeda yang tinggi, sehingga pengelompokan pola berdasarkan ciri yang dimiliki dapat dilakukan dengan keakuratan yang tinggi (Syafitri 2011). Ciri pada suatu pola diperoleh dari hasil pengukuran terhadap objek uji. Khusus pada pola yang terdapat di dalam citra, ciri-ciri yang dapat diperoleh berasal dari informasi berikut.

1) Spasial: intensitas piksel, histogram.
2) Tepi: arah, kekuatan.
3) Kontur: garis, elips, lingkaran.
4) Wilayah/bentuk: keliling, luas, pusat massa.
5) Hasil transformasi Fourier: frekuensi.

Pengenalan pola bertujuan menentukan kelompok atau kategori pola berdasarkan ciri-ciri yang dimiliki oleh pola tersebut. Dengan kata lain, pengenalan pola membedakan suatu objek dengan objek lain. Gambar 2.2 memperlihatkan ilustrasi pengenalan pola.

Gambar 2.2. Pengenalan pola (Idhawati, 2007)

Terdapat dua pendekatan yang dilakukan dalam pengenalan pola: pendekatan secara statistik dan pendekatan secara sintaktik atau struktural (Rinaldi, 2004).

1) Pengenalan Pola secara Statistik

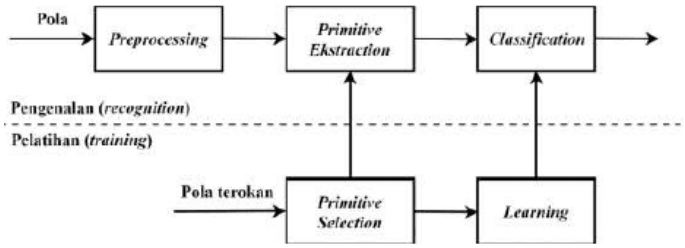
Pendekatan ini menggunakan teori-teori ilmu peluang dan statistik. Ciri-ciri yang dimiliki oleh suatu pola ditentukan distribusi statistiknya. Pola yang berbeda memiliki distribusi yang berbeda pula. Dengan menggunakan teori keputusan di dalam statistik, kita menggunakan distribusi ciri untuk mengklasifikasikan pola. Ada dua fase dalam sistem pengenalan pola: (i) fase pelatihan dan (ii) fase pengenalan. Pada fase pelatihan, beberapa contoh citra dipelajari untuk menentukan ciri yang akan digunakan dalam proses pengenalan serta prosedur klasifikasinya. Pada fase pengenalan, citra diambil cirinya kemudian ditentukan kelas kelompoknya.

Kumpulan ciri dari suatu pola dinyatakan sebagai vektor ciri dalam ruang bahumatra (multi dimensi). Jadi, setiap pola dinyatakan sebagai sebuah titik dalam ruang bahumatra. Ruang bahumatra dibagi menjadi sejumlah uparuang (sub-ruang). Tiap uparuang dibentuk berdasarkan pola-pola yang sudah dikenali kategori dan ciri-cirinya (melalui fase pelatihan) (Rinaldi, 2004).

2) Pengenalan Pola secara Sintaktik

Pendekatan ini menggunakan teori bahasa formal. Ciri-ciri yang terdapat pada suatu pola ditentukan primitif dan hubungan struktural antara primitif kemudian menyusun tata bahasanya. Dari aturan produksi pada tata

bahasa tersebut dapat ditentukan kelompok pola. Gambar 2.3 memperlihatkan sistem pengenalan pola dengan pendekatan sintaktik.



Gambar 2.3. Sistem pengenalan pola dengan pendekatan sintaktik

Pengenalan pola secara sintaktik lebih dekat ke strategi pengenalan pola yang dilakukan manusia, namun secara praktik penerapannya relatif sulit dibandingkan pengenalan pola secara statistik (Rinaldi, 2004).

2.3 *Face Detection*

Face Detection atau pendeteksian wajah dalam pengolahan citra adalah suatu proses untuk mendeteksi wajah manusia dengan cara menentukan letak dan ukuran wajah di dalam citra digital. Teknologi ini dapat mendeteksi wajah melalui ciri atau sifat wajah dan tidak memedulikan hal-hal lainnya, seperti bangunan, pohon dan badan manusia itu sendiri. Bidang-bidang penelitian yang juga berkaitan dengan pemrosesan wajah (*face processing*) adalah autentikasi wajah (*face authentication*), lokalisasi wajah (*face localization*), penjejakan wajah (*face tracking*), dan pengenalan ekspresi wajah (*facial expression recognition*) (Yang, 2002).

Deteksi wajah merupakan salah satu tahap awal (*preprocessing*) yang sangat penting sebelum dilakukan proses pengenalan wajah (*face recognition*). Deteksi wajah dapat juga diartikan dengan deteksi benda yang spesifik. Dalam kasus ini

benda yang dideteksi secara spesifik atau berupa wajah manusia yang sering disebut dengan istilah fitur. Yaitu bagian wajah manusia yang memiliki ciri khusus, seperti mata, hidung, mulut, pipi, dahi dan dagu. Adapun faktor yang dapat memengaruhi deteksi wajah, antara lain (Rinaldi, 2004).

1) Pose, yaitu bagian wajah yang terlihat pada citra bisa bervariasi (bagian depan terlihat jelas, bagian wajah ada yang tidak terlihat).
2) Komponen, yaitu struktural fitur pada wajah seperti kumis, jenggot, kacamata dan beberapa komponen yang bisa membuat wajah berbeda dari satu dengan yang lain. Seperti bentuk wajah, warna kulit, dan ukuran.
3) Ekspresi wajah, yaitu ekspresi wajah yang ada pada citra.
4) Orientasi citra, yaitu pengambilan gambar pada objek citra.
5) Kondisi citra seperti kondisi pencahayaan (spektrum), dan karakteristik kamera (sensor, *response*, lensa) berpengaruh terhadap tampilan wajah.

2.4 *Face Recognition*

Face Recognition adalah proses mengidentifikasi orang dalam gambar atau *frame* video dengan membandingkan tampilan wajah dalam citra yang diambil dengan yang ada pada *database*.

Sebagai sistem biometrik, *face recognition* beroperasi dalam salah satu atau kedua mode: (1) verifikasi wajah (atau otentikasi), dan (2) identifikasi wajah (atau pengenalan).

Verifikasi wajah merupakan sebuah sistem *one-to-one match* yang membandingkan citra wajah *query* dengan citra wajah pendaftaran yang

identitasnya diklaim. Salah satu aplikasi yang khas adalah pengguna *E-passport* untuk pembersihan imigrasi swalayan.

Identifikasi wajah merupakan sebuah sistem pencocokan *one-to-many match* yang membandingkan citra wajah *query* dengan beberapa citra wajah dalam *database* pendaftaran untuk mengaitkan identitas *query* wajah dengan salah satu yang ada di dalam *database*.

2.5 *Multi-Task Cascaded Convolutional Neural Networks*

MTCNN atau *Multi-Task Cascaded Neural Networks* adalah jaringan saraf yang digunakan untuk mendeteksi wajah dan *landmark* wajah pada gambar. MTCNN diterbitkan pada tahun 2016 oleh Zhang et al (Wang, 2018). MTCNN adalah suatu algoritma yang terdiri dari 3 jaringan saraf konvolusional, yaitu P-Net, R-Net, dan O-Net, yang mendeteksi *bounding box* pada wajah dan 5 *Point Face Landmarks* dalam sebuah gambar. Secara bertahap, setiap jaringan saraf meningkatkan hasil deteksi dengan melewatkan *input*-nya melalui CNN dan diikuti oleh *Non-Maximum Suppression*, atau NMS, yaitu metode yang mengurangi jumlah *bounding box* (Mühler, 2018).

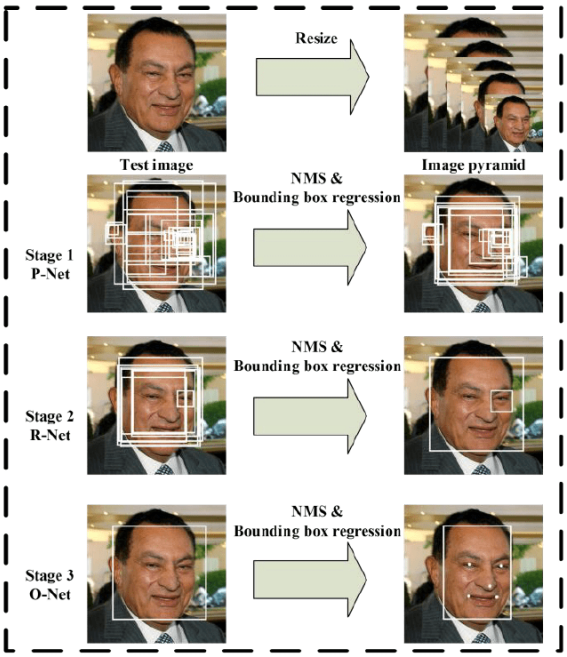
CNN (*Convolutional Neural Networks*) merupakan operasi konvolusi yang menggabungkan beberapa lapisan pemrosesan, menggunakan beberapa elemen yang beroperasi secara paralel dan terinspirasi oleh sistem saraf biologis (Hu et al., 2015). Setiap *neural networks* memiliki *input layer*, *hidden layers*, dan *output layer*. Yang membedakan suatu *neural network* dengan yang lainnya adalah tipe *hidden layers* yang digunakan. *Hidden layers* dan proses pada CNN terdiri dari

convolutional layers, ReLU (*Rectified Linear Unit*) *layers*, *Pooling Layers*, *Flattening Process*, dan *Fully Connected Layers*. Umumnya CNN digunakan untuk mendeteksi dan mengenali objek pada sebuah citra. CNN banyak diterapkan pada algoritma pendeteksian wajah modern, salah satunya adalah MTCNN. Konsep MTCNN dapat dijelaskan dalam tiga tahap yang terdiri dari beragam CNN dengan kompleksitas yang bervariasi.

Pada arsitekturnya, MTCNN menerapkan lebih dari satu teknik *Machine Learnin*g atau dikenal dengan istilah *multi-task learning*. Teknik yang digunakan yaitu klasifikasi dan regresi. Klasifikasi digunakan dalam pendeteksian wajah, sedangkan regresi digunakan dalam penentuan lokasi *bounding box* dan *face landmark*.

MTCNN menggabungkan *multi-task learning* dengan *framework* CNN dengan berbagi beberapa lapisan untuk tugas yang berbeda. MTCNN menggunakan CASIANet dengan tiga modifikasi. Pertama, normalisasi *batch* diterapkan untuk mempercepat proses pelatihan. Kedua, *contrastive loss* dihilangkan untuk menyederhanakan fungsi *loss*. Ketiga, dimensi dari *fully connected* layer sepenuhnya diubah sesuai dengan tugas yang berbeda (Ganidisastra, 2021).

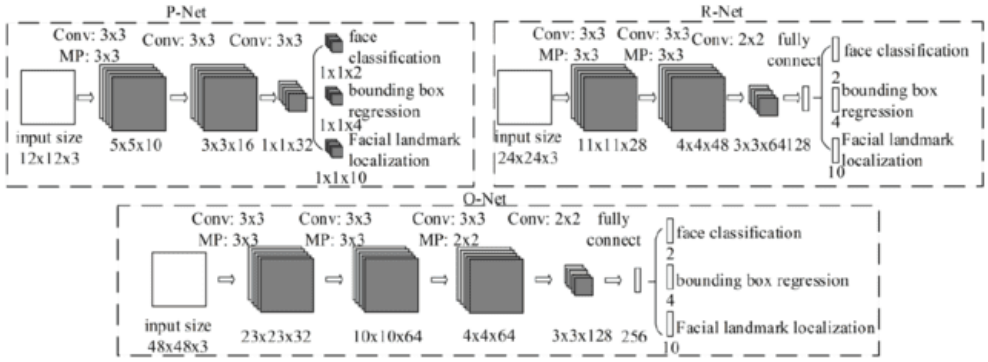
Proses pendeteksian wajah dengan MTCNN dimulai dengan menghasilkan beberapa kotak kandidat (*candidate windows*) melalui *Proposal Network* (P-Net). Setelah itu, kandidat-kandidat tersebut disaring melalui tahap *Refinement Network* (R-Net). Selanjutnya, *Output Network* (O-Net) menghasilkan *bounding box* akhir beserta posisi *landmark* wajah. Ilustrasi alur pendeteksian wajah dengan MTCNN dapat dilihat pada Gambar 2.4.



Gambar 2.4. Alur deteksi wajah MTCNN (Zhang et al., 2016)

Dalam sebuah studi yang dilakukan oleh Li et al. pada tahun 2015, beberapa CNN telah dirancang untuk melakukan pendeteksian wajah. Namun, kinerjanya dibatasi oleh beberapa hal antara lain: (1) Beberapa filter tidak memiliki bobot yang bervariasi yang dapat membatasi mereka untuk menghasilkan deskripsi yang diskriminatif. (2) Dibandingkan dengan metode pendeteksian objek *multi-class* dan proses klasifikasi lainnya, pendeteksian wajah adalah proses klasifikasi biner yang menantang, sehingga kemungkinan memerlukan lebih sedikit jumlah filter, tetapi dengan diskriminasi yang lebih banyak. Untuk mencapai hal tersebut, jumlah filter dikurangi dan filter 5x5 diubah menjadi 3x3 untuk mengurangi komputasi sekaligus meningkatkan kedalaman untuk memperoleh kinerja yang lebih baik. Dengan peningkatan ini, diperoleh kinerja yang lebih baik dengan waktu komputasi yang

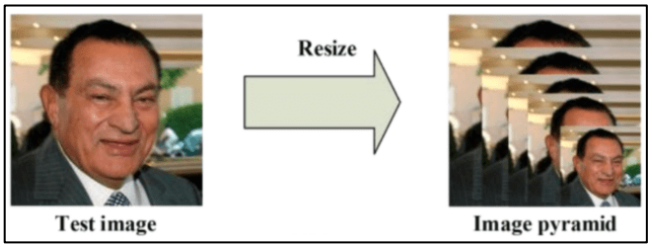
lebih sedikit dibandingkan dengan arsitektur sebelumnya. Arsitektur CNN dari MTCNN dapat dilihat pada Gambar 2.5.



Gambar 2.5. Arsitektur dari MTCNN (Zhang et al., 2016)

Berdasarkan Gambar 2.5, proses pendeteksian wajah pada seluruh jaringan MTCNN secara lebih rinci terdiri dari beberapa tahap sebagai berikut (Wang, 2018):

Tahap 1: Muat suatu citra ke dalam program. Kemudian akan dibuat suatu *image pyramid* yang bertujuan agar wajah dapat dideteksi pada ukuran yang berbeda-beda. *Image pyramid* merupakan kumpulan citra yang terdiri dari beberapa salinan citra masukan yang diperkecil dengan skala tertentu hingga mencapai ukuran minimum wajah yang telah ditentukan, misalnya 40x40. Ilustrasi pembuatan *image pyramid* dari citra masukan dapat dilihat pada Gambar 2.6.



Gambar 2.6. *Image pyramid* (Zhang et al., 2016)

Pada setiap salinan citra yang telah diskalakan, akan digunakan kernel berukuran 12 x 12 untuk mengambil bagian dari citra yang kemudian menjadi *input* pada P-Net. Untuk setiap salinan citra, kernel 12 x 12 akan bergerak melalui setiap bagian pada citra untuk mendeteksi wajah. Kernel bergerak dari sudut kiri atas, yaitu bagian pada citra pada titik (0,0) hingga (12,12). Jika terdeteksi wajah pada bagian citra tersebut, maka P-Net akan mengembalikan koordinat kotak pembatas (*bounding box*) dari koordinat kernel. Kemudian, proses tersebut diulangi lagi pada bagian (0+2a, 0+2b) hingga (12+2a, 12+2b), menggeser kernel 12 x 12 sebanyak 2 piksel ke kanan atau ke bawah pada satu waktu. Pergeseran 2 piksel ini dikenal sebagai langkah (*stride*), atau berapa banyak piksel yang dilalui setiap saat kernel bergeser.

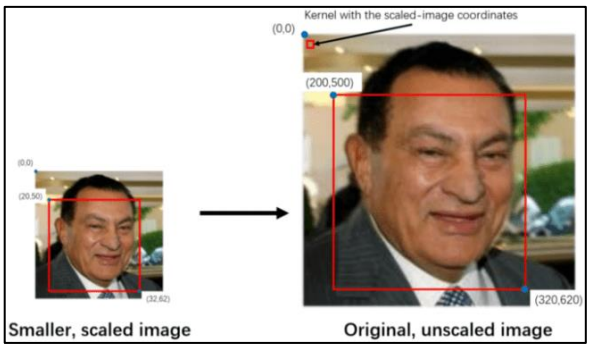
Melangkah 2 piksel membantu mengurangi kompleksitas komputasi tanpa mengorbankan akurasi secara signifikan. Hal ini disebabkan karena wajah di sebagian besar citra secara signifikan lebih besar dari dua piksel. Pada saat yang sama, komputer hanya mengeksekusi seperempat dari jumlah operasi yang ada sehingga program berjalan lebih cepat dan membutuhkan lebih sedikit memori.

Akan tetapi, dengan melakukan pergeseran 2 piksel, diperlukan perhitungan ulang pada semua indeks kernel untuk memperoleh koordinat kernel yang tepat. Misalnya jika kernel mendeteksi wajah setelah bergerak satu langkah ke kanan, indeks keluaran akan memberi tahu bahwa sudut kiri atas kernel berada di (1,0). Namun, karena langkahnya adalah 2, perlu untuk mengalikan indeks dengan 2 untuk mendapatkan koordinat yang tepat yaitu (2,0). Contoh *output* dari P-Net dapat dilihat pada Tabel 2.1 sebagai berikut.

Tabel 2.1. Contoh *output* dari P-Net

Koordinat Kernel	x_1	y_1	x_2	y_2	*Confidence*
(0, 5)	0.50	0.25	0.80	0.58	0.98
(0, 6)	0.45	0.17	0.75	0.49	0.96
(0, 7)	0.52	0.08	0.73	0.41	0.94
(0, 4)	0.42	0.34	0.67	0.66	0.92
(0, 8)	0.40	0.01	0.66	0.32	0.96

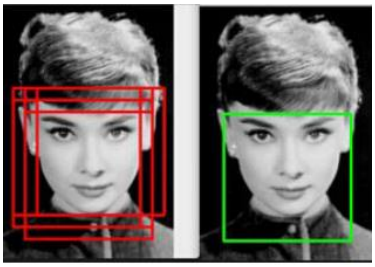
Bobot dan bias P-Net telah dilatih sehingga menghasilkan kotak pembatas yang relatif akurat untuk setiap kernel 12 x 12. Namun, beberapa kotak memiliki *confidence* yang lebih tinggi dibandingkan dengan kotak lainnya. Dengan demikian, diperlukan untuk mengeliminasi *output* dari P-Net agar diperoleh daftar nilai *confidence* untuk setiap kotak pembatas dan menghapus kotak pembatas dengan *confidence* yang lebih rendah.



Gambar 2.7. Standardisasi koordinat kernel pada citra asli (Wang, 2018)

Setelah memilih kotak dengan *confidence* yang lebih tinggi, dilakukan standardisasi sistem koordinat dengan mengubah semua koordinat pada citra asli yang tidak diskalakan. Selanjutnya, dilakukan proses *Non-Maximum Suppression* atau NMS, yaitu metode yang digunakan untuk mengeliminasi beberapa kotak pembatas yang tumpang tindih dari objek yang sama untuk memperoleh kotak

pembatas yang optimal. Pertama-tama, dipilih satu kotak pembatas berdasarkan nilai *confidence* atau skor yang tertinggi. Kemudian, dihitung nilai *Intersection over Union* (IoU) pada setiap kotak pembatas yang tumpang tindih terhadap kotak pembatas dengan *confidence* tertinggi tersebut. Dengan menentukan nilai *threshold*, semua kandidat yang memiliki nilai IoU lebih besar dari *threshold* akan dieliminasi (Zhang et al., 2020).



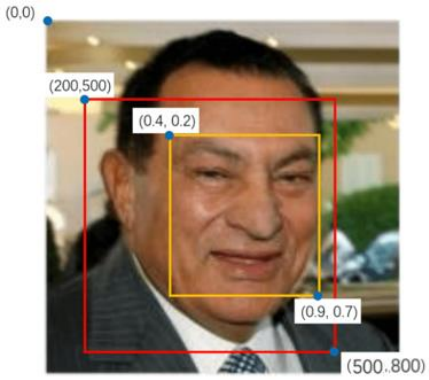
Gambar 2.8. *Non-Maximum Suppression* (Zhang et al, 2020)

Proses NMS dilakukan sekali untuk setiap citra yang diskalakan, kemudian dilakukan sekali lagi pada semua kernel yang tersisa dari setiap skala. Proses ini menghilangkan kotak pembatas yang berlebihan sehingga mempersempit pencarian ke satu kotak yang akurat per wajah.

Mengapa kotak dengan *confidence* tertinggi tidak langsung dipilih dan menghapus yang lainnya? Hanya ada satu wajah pada citra di atas. Namun, mungkin ada lebih dari satu wajah di citra lain. Jika kotak dengan *confidence* tertinggi langsung dipilih begitu saja, proses ini akan menghapus semua kotak pembatas untuk wajah lainnya.

Setelah itu, koordinat kotak pembatas dikonversi menjadi koordinat citra awal. Saat ini, koordinat dari setiap kotak pembatas adalah nilai antara 0 dan 1,

dengan (0,0) sebagai sudut kiri atas kernel 12 x 12 dan (1,1) sebagai sudut kanan bawah (lihat Tabel 2.1 di atas). Dengan mengalikan koordinat dengan lebar dan tinggi citra awal, koordinat kotak pembatas dapat dikonversi menjadi standar koordinat pada ukuran citra yang sebenarnya.



Gambar 2.9. *Bounding box* di dalam kernel 12 x 12 (Wang, 2018)

Pada Gambar 2.9, kotak merah mewakili kernel 12 x 12 yang telah diubah ukurannya kembali ke citra aslinya. Dapat dihitung lebar dan tinggi kernel yaitu 500–200 = 300 dan 800–500 = 300. Perhatikan bagaimana lebar dan tinggi tidak harus 12. Hal ini disebabkan karena panjang dan lebar yang didapatkan di sini adalah panjang dan lebar kernel ketika diskalakan ke ukuran aslinya. Setelah itu, kalikan koordinat kotak pembatas dengan 300 sehingga 0.4x300 = 120, 0.2x300 = 60, 0.9x300 = 270, 0,7x300 = 210. Terakhir, tambahkan koordinat kiri atas kernel untuk mendapatkan koordinat kotak pembatas: (200+120, 500+60) dan (200+270, 500+210) atau (320, 560) dan (470, 710).

Kotak pembatas yang dihasilkan dari tahap ini bisa saja tidak berbentuk persegi. Oleh karena itu, kotak pembatas dibentuk menjadi persegi dengan

memanjangkan sisi yang lebih pendek. Akhirnya, koordinat kotak pembatas disimpan dan diteruskan ke tahap 2.

Tahap 2: Pada beberapa kasus, sebuah citra hanya berisi sebagian wajah yang terlihat dari sisi *frame*. Dalam hal ini, *network* dapat mengembalikan kotak pembatas yang sebagian areanya keluar dari *frame*. Kasus seperti ini dapat diatasi dengan melakukan *padding*, yaitu proses mengisi area di luar *frame* dengan nilai 0.

Setelah mendapatkan kotak pembatas, ukurannya akan diubah menjadi 24 x 24 piksel dan dilakukan normalisasi ke nilai antara -1 dan 1. Saat ini, nilai piksel pada citra antara 0 hingga 255 (nilai RGB). Dengan mengurangkan setiap nilai piksel dengan setengah dari 255 (127,5) dan membaginya dengan 127,5, maka dapat dipertahankan nilainya antara -1 dan 1.

Beberapa larik citra berukuran 24 x 24 yang telah diubah ukurannya dan di normalisasi dari Tahap 1 akan dimasukkan ke dalam *Refine Network* (R-Net). *Output* dari R-Net mirip dengan P-Net, yaitu berupa koordinat kotak pembatas baru yang lebih akurat, serta nilai *confidence* dari masing-masing kotak pembatas tersebut.

Sekali lagi, kotak dengan *confidence* yang lebih rendah disingkirkan dan proses NMS dilakukan pada setiap kotak untuk mengeliminimasi kotak yang tumpang tindih. Karena koordinat kotak pembatas baru ini didasarkan pada kotak pembatas P-Net, maka diperlukan untuk mengonversinya ke koordinat standar. Setelah standardisasi koordinat, akan dibentuk kembali kotak pembatas menjadi persegi untuk diteruskan ke O-Net.

Tahap 3: Keluaran dari *Output Network* (O-Net) sedikit berbeda dengan P-Net dan R-Net. O-Net menyediakan 3 *output* yaitu koordinat kotak pembatas, koordinat 5 *landmark* wajah, dan *confidence* setiap kotak. Sekali lagi, kotak dengan nilai *confidence* yang lebih rendah dieliminasi, dan dilakukan standardisasi baik pada koordinat kotak pembatas maupun koordinat *landmark* wajah. Selanjutnya, semua kotak tersebut melalui NMS untuk terakhir kalinya. Pada titik ini, seharusnya hanya ada satu kotak pembatas untuk setiap wajah dalam citra (Wang, 2018). *Network layer* dari O-Net lebih dalam dari *network* sebelumnya. Hal ini mengakibatkan hasil pendeteksian wajah yang diperoleh dari *network* ini merupakan yang terbaik dibandingkan hasil dari *network* sebelumnya (Zhang et al., 2020).

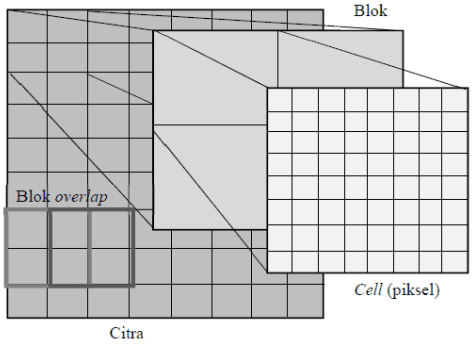
2.6 *Histogram of Oriented Gradients*

Histogram of Oriented Gradients (HOG) adalah 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 *histogram of oriented gradients*. Setiap piksel dalam *cell* berkontribusi pada saat dilakukan voting bobot untuk membangun sebuah histogram yang berorientasi pada nilai-nilai gradien yang dihitung (Pranoto et al., 2017).

Teknik ini menghitung nilai gradien dalam daerah tertentu pada suatu gambar atau citra masukan. Tiap gambar mempunyai karakteristik yang ditunjukkan oleh

distribusi gradien. Karakteristik ini diperoleh dengan membagi gambar ke dalam daerah kecil yang disebut *cell*. Dari tiap *cell* disusun sebuah histogram yang nilainya diperoleh dari nilai gradien pada *cell* tersebut. Kombinasi dari histogram ini dijadikan sebagai deskriptor yang mewakili sebuah obyek (Afifah, 2017).

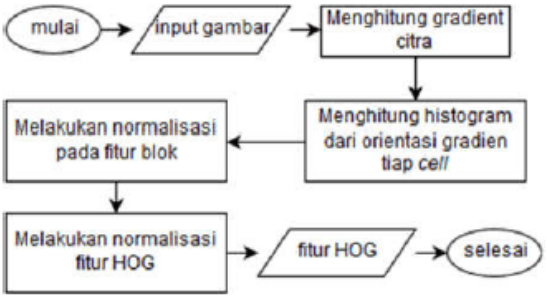
Pembuatan deskripsi fitur HOG diawali dengan menentukan ukuran citra. Citra kemudian dibagi ke dalam beberapa *cell* dengan ukuran piksel *cell* tertentu. Di dalam *cell* inilah terjadi pembuatan histogram. *Cells* ini akan terdapat pada sebuah blok, untuk lebih jelas, dapat dilihat pada Gambar 2.10. Blok ini nantinya yang akan bergerak iterasi dari kiri atas citra ke samping kanan sampai ke bawah kanan citra.



Gambar 2.10. Ilustrasi pembagian citra HOG (Afifah, 2017)

HOG bekerja menggunakan konsep pergeseran *window* dengan menghitung vektor gradien yang diperoleh pada setiap *window*. Besaran nilai dan arah dari vektor gradien dalam daerah tertentu secara keseluruhan akan menampilkan karakteristik distribusi gradien sebuah gambar. Karakteristik distribusi gradien akan menggambarkan bentuk ciri suatu objek dalam gambar sehingga dapat dilakukan pelatihan untuk mengenali suatu objek. Pada akhirnya, objek dalam suatu

gambar akan dapat dikenali dengan membandingkan tingkat kemiripan distribusi gradien yang telah dilatih terhadap distribusi gradien dalam gambar yang akan dideteksi (Rachmat, 2018a).



Gambar 2.11. Alur *Histogram of Oriented Gradients* (Randa et al., 2015)

Gambar 2.11 menunjukkan tahap-tahap untuk memperoleh fitur deskriptor HOG. Tahap awal dari metode HOG adalah menghitung nilai gradien citra. Gradien citra dapat dihitung dengan membagi citra ke dalam beberapa area kecil. Area ini dapat dikatakan sebagai *window* yang membagi n x n *grid*. Gradien citra dapat dihitung dengan Persamaan (2.2)

$$|G| = \sqrt{Ix^2 + \mathrm{Iy}^2} \qquad (2.2)$$

Di mana I adalah citra *graylevel*. I_x merupakan nilai gradien terhadap sumbu *x* dan I_y merupakan nilai gradien terhadap sumbu *y*. I_x dan I_y dapat dihitung dengan Persamaan (2.3).

$$I_x = I * D_x \,,\; I_y = I * D_y \qquad (2.3)$$

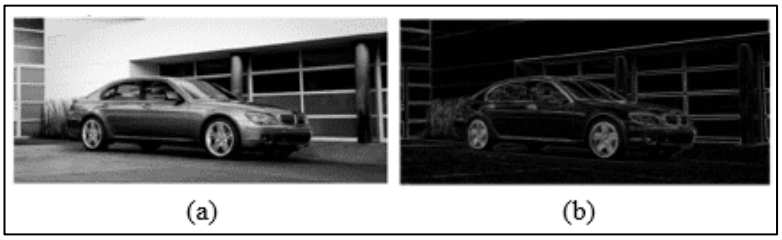
D_x adalah *mask* [−1 0 1], sedangkan D_y adalah *mask* D_x yang ditranspos.

Masing-masing dihitung dengan cara konvolusi. Kemudian gradien ditransformasi ke dalam koordinat sumbu dengan sudut di antara 0° sampai 180°

yang disebut orientasi gradien. Orientasi gradien (θ) dapat dihitung dengan Persamaan (2.4).

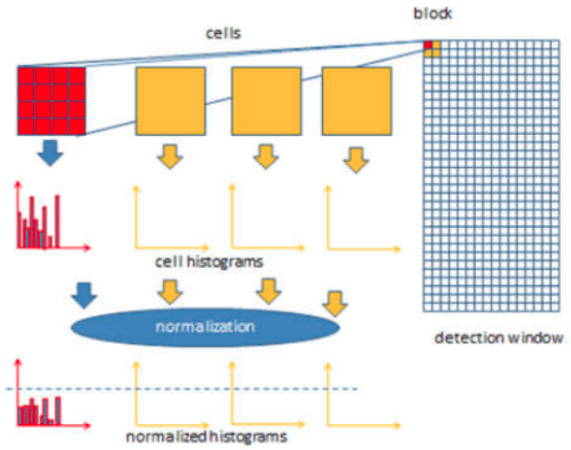
$$\theta = \arctan\left(\frac{I_x}{I_y}\right) \qquad (2.4)$$

Ilustrasi dari langkah pertama dapat dilihat pada Gambar 2.12. Bagian (a) merupakan citra awal sedangkan bagian (b) merupakan citra yang telah dihitung nilai gradiennya.



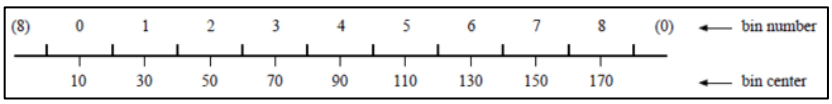
Gambar 2.12. (a) Citra awal; (b) Citra setelah penghitungan gradien (Randa, 2015)

Tahap selanjutnya adalah melakukan perhitungan histogram dari orientasi gradien tiap *cell*. Setiap piksel dalam sebuah *cell* mempunyai nilai histogram sendiri-sendiri berdasarkan nilai yang dihasilkan dalam perhitungan gradien yang kemudian dilakukan normalisasi pada setiap blok. *Cell* memiliki ukuran 8x8 piksel pada sebuah citra. Sedangkan blok memiliki ukuran 2x2 *cell*. Ilustrasi ditunjukkan pada Gambar 2.13.



Gambar 2.13. *Cell* yang menyusun sebuah blok (Randa et al., 2015)

Jumlah bin histogram yang umum digunakan dalam penerapan HOG adalah 9 bin (B=9). Penentuan nilai histogram dilakukan dengan melakukan voting terhadap masing-masing *cell* pada citra. Gambar 2.14 merupakan gambar 9 bin histogram yang digunakan pada voting untuk menentukan kontribusi nilai histogram.



Gambar 2.14. Panjang bin histogram

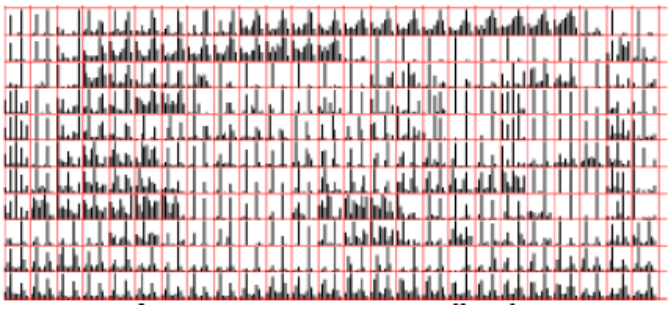
Bin yang dipakai berukuran sembilan, dengan nilai bin dari 0 sampai B-1. Kontribusi nilai histogram (v) dapat diperoleh dari Persamaan (2.5) dan Persamaan (2.6). Di mana μ merupakan besar gradien pada piksel, c merupakan nilai tengah sudut pada bin, θ adalah sudut orientasi gradien pada piksel, w adalah lebar dari nilai tengah sudut yaitu $w = \frac{180}{B}$ dan B adalah panjang bin histogram yang digunakan.

$$v_j = \mu \frac{c_{j+1} - \theta}{w} \text{ untuk bin ke- } j = (\frac{\theta}{w} - \frac{1}{2}) \; mod \; B \qquad (2.5)$$

$$v_{j+1} = \mu \frac{\theta - c_j}{w} \text{ untuk bin ke- } (j+1) \; mod \; B \qquad (2.6)$$

Gambar 2.15. Citra setelah dibagi menjadi beberapa *cell*

Ilustrasinya pembagian citra menjadi *cell* dapat dilihat pada Gambar 2.15. Untuk setiap *cell* akan dicari nilai *Ix* dan *Iy*. Nilai rentang *Ix* dan *Iy* yaitu antara -1 sampai 1. *Cell* kesatu berada pada pojok kiri atas. Setalah diketahui nilai *Ix* dan *Iy* pada citra, kemudian dihitung besar gradien dan orientasi sudut gradiennya. Nilai orientasi sudut memiliki rentang 0 sampai 180. Dari nilai gradien dan orientasi sudut gradien dapat dihitung nilai kontribusi histogramnya menggunakan Persamaan 2.5 dan Persamaan 2.6. Gambar 2.16 merupakan ilustrasi histogram pada masing-masing *cell*.



Gambar 2.16. Histogram tiap *cell* pada citra

Nilai normalisasi fitur blok didapat dari Persamaan (2.7). Fitur blok dinormalisasi untuk mengurangi efek perubahan kecerahan obyek pada satu blok. Variabel *b* merupakan nilai blok fitur dan variabel *e* merupakan bilangan positif yang bernilai kecil untuk mencegah pembagian dengan 0.

$$b = \frac{b}{\sqrt{b^2 + \varepsilon}} \tag{2.7}$$

Nilai normalisasi tiap blok digabungkan menjadi satu vektor menjadi fitur vektor HOG. Kemudian fitur vektor HOG dilakukan normalisasi. Normalisasi dilakukan melalui Persamaan (2.8). Variabel *h* merupakan nilai fitur HOG dan variabel ε merupakan bilangan positif yang bernilai kecil untuk mencegah pembagian dengan 0 (Randa, 2015).

$$h = \frac{h}{\sqrt{||h||^2 + \varepsilon}} \tag{2.8}$$

Beberapa normalisasi telah dieksplor oleh Dalal dan Triggs (2005). Didefinisikan variabel *v* sebagai vektor yang berisi semua histogram dari suatu *block*. $||v||_k$ merupakan k-norm dari *v*, dengan k ∈ 1, 2 dan ε merupakan suatu konstan dengan nilai kecil. Maka skema normalisasinya adalah sebagai berikut (Fernández, 2014).

$$L1 - norm: v = \frac{v}{||v||_2 + \varepsilon} \tag{2.9}$$

$$L1 - sqrt: v = \sqrt{\frac{v}{||v||_2 + \varepsilon}} \tag{2.10}$$

$$L2 - norm: v = \frac{v}{\sqrt{||v||_2^2 + \varepsilon^2}} \tag{2.11}$$

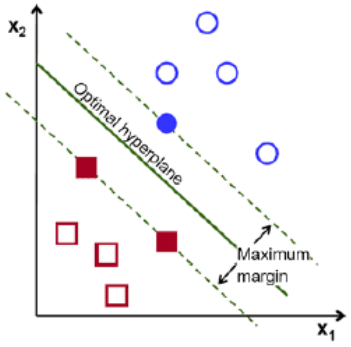
L2-Hys atau L2 Hysteresis merupakan normalisasi menggunakan metode L2 yang diikuti oleh pembatasan nilai maksimum ke 0,2 dan dinormalisasi kembali

menggunakan metode L2. Eksperimen ini menunjukkan bahwa L1-sqrt, L2-norm, dan L2-Hys memberikan hasil yang baik dan perbedaannya tidak signifikan. Tetapi L1-norm mengurangi performa sebanyak 5%.

2.7 *Support Vector Machine*

Support Vector Machine (SVM) dikembangkan oleh Boser, Guyon dan Vapnik, serta pertama kali dipresentasikan pada tahun 1992 di *Annual Workshop on Computational Learning Theory*. Konsep dasar SVM sebenarnya merupakan kombinasi harmonis dari teori-teori komputasi yang telah ada puluhan tahun sebelumnya, seperti margin *hyperplane* yang diperkenalkan oleh Aronszajn tahun 1950. Demikian juga dengan konsep-konsep pendukung yang lain. Akan tetapi hingga tahun 1992 belum pernah ada upaya merangkaikan komponen – komponen tersebut. Prinsip dasar SVM adalah linear *classifier*, dan selanjutnya dikembangkan agar dapat bekerja pada problem non-linear dengan memasukkan konsep *kernel trick* pada ruang kerja berdimensi tinggi (Harahap et al., 2018).

Support Vector Machine (SVM) merupakan pembelajaran yang mengarah ke pemrograman kuadratik dengan kendala linear. Berdasarkan minimalisasi risiko prinsip terstruktur, SVM berusaha untuk meminimalkan batas atas kesalahan generalisasi bukan kesalahan empiris, sehingga model prediksi baru efektif menghindari over-pas masalah. Selain itu, model SVM bekerja di ruang fitur berdimensi tinggi yang dibentuk oleh pemetaan non-linear dari N-dimensi vektor *input* x ke dalam ruang fitur K-dimensi (K>N) melalui penggunaan fungsi φ non-linear (x) (Li et al., 2015).



Gambar 2.17. Penentuan *hyperplane* terbaik (Prasetyo, 2012).

Hyperplane (batas keputusan) pemisah terbaik antara kedua kelas dapat ditemukan dengan mengukur *margin hyperplane* tersebut dan mencari titik maksimalnya. *Margin* adalah jarak antara *hyperplane* tersebut dengan data terdekat dari masing-masing kelas. Data yang paling dekat ini disebut *support vector*. Garis solid pada gambar di atas menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua kelas, sedangkan data lingkaran dan bujur sangkar yang dilewati garis batas *margin* (garis putus-putus) adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pelatihan SVM (Prasetyo, 2012).

Margin adalah jarak antara *hyperplane* tersebut dengan *pattern* terdekat dari masing-masing *class*. *Pattern* yang paling dekat ini disebut sebagai *support vector*. Garis solid pada Gambar 2.17 menunjukkan *hyperplane* yang terbaik, yaitu yang terletak tepat pada tengah-tengah kedua *class*, sedangkan titik merah dan kuning yang berada dalam lingkaran hitam adalah *support vector*. Usaha untuk mencari lokasi *hyperplane* ini merupakan inti dari proses pembelajaran pada SVM. Proses pembelajaran SVM adalah untuk menentukan *support vector*, kita hanya cukup

mengetahui fungsi kernel yang dipakai, dan tidak perlu mengetahui wujud dari fungsi non-linear. Berikut adalah persamaan dari fungsi SVM (Safitri, 2019).

$$f(x) = w^2\phi(x) + b \qquad (2.12)$$

Di mana :

- b adalah bias
- $x = (x_1, x_2, \ldots . x_D)^T$ adalah variabel *input*
- $w = (w_0, w_1, \ldots . w_D)^T$ adalah parameter bobot
- $\phi(x)$ adalah fungsi transformasi fitur

Dalam SVM terdapat fungsi kernel trik yang terbagi dua yaitu kernel linear dan kernel non-linear. Kernel linear digunakan ketika data yang akan diklasifikasi dapat terpisah dengan sebuah garis (*hyperplane*). Sedangkan kernel non-linear digunakan ketika data hanya dapat dipisahkan dengan garis lengkung atau sebuah bidang pada ruang dimensi tinggi. Contoh kernel non-linear adalah Polynomial, Gaussian RBF, Sigmoid, Additive (Rachmat, 2018).

Fungsi kernel yang umum digunakan pada metode SVM adalah:

1. Kernel Linear

   $K(x_i, x_j) = x_i, x_j$

2. Kernel Polynomial

   $K(x_i, x_j) = (\gamma(x_i, x_j) + r)^p \ , \qquad \gamma > 0$

3. Kernel Radial Basis Function (RBF)

   $K(x_i, x_j) = \exp\left(\gamma \left\| x_i - x_j \right\|^2\right), \qquad \gamma > 0$

4. Kernel Sigmoid

$$K(x_i, x_j) = \tanh(\gamma(x_i, x_j) + r) \ , \qquad \gamma > 0$$

Keterangan :

- (x_i, x_j) adalah vektor *input*
- r adalah variabel
- *p* adalah derajat polynomial
- γ (*gamma*) menentukan seberapa besar pengaruh yang dimiliki setiap data latih, semakin besar nilainya, semakin dekat data lain akan terpengaruh
- *c (complexity)* menentukan kompleksitas decision surface, nilai kecil akan memperhalus decision surface, sedangkan nilai tinggi akan menargetkan semua data latih diklasifikasikan dengan benar.

Pemilihan fungsi kernel yang tepat merupakan hal yang sangat penting karena akan menentukan *feature space* dimana fungsi *classifier* akan dicari. Sepanjang fungsi kernelnya sesuai (cocok), SVM akan beroperasi secara benar meskipun tidak tahu pemetaan yang digunakan (Santosa, 2007; Robandi & Prasetyo, 2008). Menurut Scholkopf dan Smola (1997), fungsi kernel gaussian RBF memiliki kelebihan yaitu secara otomatis menentukan nilai, lokasi dari *center* serta nilai pembobot dan bisa mencakup nilai rentang tak terhingga. Gaussian RBF juga efektif menghindari *overfitting* dengan memilih nilai yang tepat untuk parameter C dan γ dan RBF baik digunakan ketika tidak ada pengetahuan terdahulu. Fungsi kernel yang direkomendasikan adalah fungsi kernel RBF karena dapat memetakan hubungan tidak linear, RBF lebih robust terhadap *outlier* karena fungsi kernel RBF

berada antara selang (–∞, ∞) sedangkan fungsi kernel yang lain memiliki rentang antara -1 sampai dengan 1. (Hsu, Chang, & Lin, 2003).

2.8 *K-Folds Cross Validation*

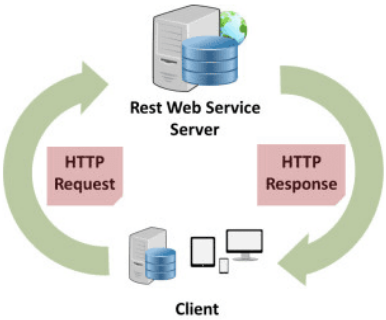
K-folds Cross Validation adalah salah satu teknik untuk validasi yang sangat populer digunakan. Metode validasi dengan *k-folds* sangat cocok digunakan untuk kasus data yang jumlah sampelnya terbatas. Untuk melakukan proses klasifikasi tentunya data dibagi ke dalam *training* dan *testing*, dan ketika data yang digunakan untuk *training* sangat sedikit kemungkinan adalah data yang digunakan kurang representatif. Dalam k-*folds cross validation*, data (D) dibagi ke dalam k *subsets* dengan jumlah yang sama. Data yang digunakan untuk *training* adalah *subsets* data k-1 yang dikombinasikan secara bersama-sama dan kemudian diaplikasikan untuk sisa satu *subsets* data sebagai hasil *testing*. Proses ini diulangi sebanyak k *subsets* dan hasil akurasi klasifikasi yaitu hasil rata-rata dari setiap data *training* dan *testing*. *k-folds* yang biasa digunakan adalah 3, 5, 10 dan 20 (Prangga, 2017).

2.9 REST

REST merupakan singkatan dari *Representational State Transfer*. Istilah REST atau RESTful pertama kali diperkenalkan oleh Roy Fielding pada disertasinya di tahun 2000. REST bukanlah sebuah standar protokol *web service*, melainkan hanya sebuah gaya arsitektur. Ide dasar dari arsitektur REST adalah bagaimana menghubungkan jalur komunikasi antar mesin atau aplikasi melalui HTTP sederhana. Sebelum adanya REST, komunikasi antar mesin atau aplikasi

dilakukan dengan menggunakan beberapa mekanisme atau protokol *middleware* yang cukup kompleks seperti DCE, CORBA, RPC, ataupun SOAP.

Arsitektur REST mampu mengeksploitasi berbagai kelebihan dari HTTP yang digunakan untuk kebutuhan *web service*. Walaupun SOAP juga dapat menggunakan protokol HTTP, namun hanya terbatas untuk kebutuhan *transport* saja. HTTP sendiri merupakan sebuah protokol standar di dunia *World Wide Web* yang berbasis *synchronous request/response*. Protokol tersebut sangat sederhana: *client* mengirimkan sebuah *request message* yang mencakup HTTP *method* yang akan di invokasi, lokasi *resource* dalam format URI (*Uniform Resource Identifier*), serta pilihan format pesan (pada dasarnya dapat berupa format apa saja seperti HTML, *plain text*, XML, JSON, ataupun data *binary*), Kemudian *server* akan mengirimkan *response* sesuai dengan spesifikasi yang diminta oleh *client*. Selama ini, yang berfungsi sebagai aplikasi *client* adalah sebuah *web browser* yang memfasilitasi komunikasi antara mesin dengan manusia. Dengan adanya REST, aplikasi *client* dapat berupa aplikasi apa saja hanya dengan memanfaatkan HTTP (Arianto et al., 2016).



Gambar 2.18. Arsitektur REST

Gambar 2.18 menunjukkan bagaimana alur pertukaran data antara *client* dan *server* pada aplikasi web dengan arsitektur REST. Suatu aksi dari pengguna yang membutuhkan pemanggilan data dari *server* akan dikemas dalam bentuk HTTP *request* yang kemudian dikirim ke REST *web service server*. *Web service* tersebut kemudian akan memproses *request* tersebut dengan menjalankan suatu fungsi yang telah diatur untuk setiap *request* yang dilakukan oleh *client*. Setelah selesai diproses, *web service* akan mengirim HTTP *response* berupa *status code* dan data yang dibutuhkan kepada *client*.

2.10 FastAPI

FastAPI adalah sebuah kerangka kerja (*framework)* Python yang dirancang untuk membangun API yang modern dan memiliki performa cepat (*high performance*). Versi minimal Python yang dibutuhkan untuk menggunakan FastAPI adalah Python versi 3.6. FastAPI dibangun menggunakan beberapa *library* Python seperti Starlette untuk bagian web dan Pydantic untuk bagian data. Fitur utama dari FastAPI antara lain (Ramírez, 2018):

- *Fast*: Memiliki performa yang sangat cepat, bersanding dengan NodeJS dan Go.
- *Fast to code*: Meningkatkan kecepatan pengembangan fitur hingga sekitar 200% hingga 300%.
- *Fewer bugs*: Mengurangi sekitar 40% kesalahan manusia (*developer*) yang menyebabkan *error*.

- *Intuitive*: Dukungan yang baik dari berbagai *text editor* sehingga waktu *debugging* dapat berjalan lebih cepat.
- *Easy*: Didesain untuk mudah digunakan dan dipelajari. Lebih sedikit waktu yang diperlukan untuk membaca dokumentasi.
- *Short*: Meminimalisir duplikasi kode. Lebih sedikit *bugs*.
- *Robust*: Dapatkan kode yang siap untuk *production*. Dokumentasi API automatis yang interaktif.
- *Standard-based*: Kompatibel dengan beberapa standar API seperti OpenAPI dan JSON Schema.

2.11 *Single Page Application*

Single Page Application (SPA) adalah aplikasi web yang memuat keseluruhan *resources* pada permintaan awal dan komponen-komponen halamannya digantikan oleh komponen lain berdasarkan interaksi yang dilakukan pengguna. Saat membandingkan SPA dengan aplikasi web tradisional, dapat ditemukan suatu analogi antara *states* pada SPA dan *web pages*. Navigasi dari *web pages* pada aplikasi web tradisional sejalan dengan navigasi *states* pada SPA. SPA terdiri dari komponen-komponen individual yang dapat digantikan atau diperbarui secara independen tanpa memuat ulang keseluruhan halaman sehingga halaman web tidak perlu dimuat ulang pada setiap aksi yang dilakukan pengguna (Jadhav et al., 2015).

Single Page Application merupakan aplikasi yang berjalan di *browser* dan tidak memerlukan pemuatan ulang halaman saat digunakan. Ini berarti bahwa

pengguna tidak berpindah halaman saat user mengirim setiap permintaan ke *server*. Menerapkan SPA memiliki dua manfaat utama yaitu mengurangi penggunaan *bandwidth* jaringan dan mempercepat penjelajahan. SPA ini didukung secara luas oleh pustaka Javascript yang mengurangi *bandwidth* jaringan. Data yang diterima dari server dalam format JSON (*JavaScript Object Notation*) dan dapat dilihat secara asinkron menggunakan Javascript, sehingga penjelajahan menjadi lebih cepat (Nasution dan Iswari, 2021).

Interaksi SPA dapat ditangani tanpa mencapai *server*, serta dapat meningkatkan kinerja dalam beberapa cara seperti waktu muat menggunakan AJAX, mudah dinavigasi ke halaman dan sebagainya. Pengguna akan lebih nyaman, karena sangat mudah untuk menavigasi atau mengarahkan ke seluruh halaman web dan filter konten yang berbeda. Penerapan teknologi SPA yang digunakan bertujuan untuk mengurangi beban kerja *server* saat terjadi permintaan data dari pengguna, dan meminimalisir penggunaan sumber daya oleh *server* (Belluano, 2018).

2.12 ReactJS

React merupakan *library* Javascript yang dikembangkan oleh Facebook untuk membuat antarmuka pengguna yang cepat dan interaktif. React biasa digunakan untuk menangani pengembangan pada SPA dan aplikasi *mobile*. ReactJS memiliki keunggulan di mana kerangka kerja ini memberikan kecepatan, *simplicity*, dan *scalability*. ReactJS dikembangkan untuk memfasilitasi pengembang dalam membuat komponen UI yang lebih interaktif, *stateful*, dan dapat digunakan

kembali. Dalam kaidah MVC (*Model View Controller*) ReactJS hanya merepresentasikan pada bagian View saja dan ini merupakan bagian terbaik dalam penyederhanaan (Nursaid et al., 2020).

ReactJS dapat memudahkan pembautan antarmuka yang interaktif. Saat merancang suatu tampilan pada aplikasi, ReactJS akan melakukan pembaruan dan menampilkan komponen yang tepat saat data pada aplikasi berubah. Penulisan sintaks dari ReactJS ditulis secara deklaratif sehingga membuat kode yang ditulis lebih mudah diprediksi dan lebih mudah untuk melakukan proses *debug.* ReactJS membangun komponen-komponen terenkapsulasi yang dapat mengatur *state* mereka sendiri yang pada tingkat lanjutnya akan memungkinkan pembuatan antarmuka yang lebih kompleks.