BAB I

PENDAHULUAN

A. Latar Belakang

Dewasa ini tidak bisa dipungkiri masyarakat Indonesia maupun dunia sedang menikmati fenomena jejaring sosial. Sebuah fenomena yang mampu menyentuh berbagai kalangan, karena keberadaannya memberikan kesempatan kepada siapapun untuk mengenal lebih banyak orang di semua belahan dunia. Tidak hanya itu, sejumlah informasi pribadi seseorang pun bisa diperoleh melalui jejaring sosial.

Seiring pesatnya perkembangan teknologi, hadirlah sistem pengenalan wajah (*face recognition*) yang juga semakin banyak dimanfaatkan pada perangkat *mobile*. Melalui sistem pengenalan wajah tersebut, identitas diri seseorang dapat diketahui dengan mudah hanya dengan memanfaatkan sistem kamera yang terdapat pada perangkat *mobile* yang diarahkan ke wajah seseorang tak dikenal. Sistem pengenalan wajah bukanlah hal yang baru dalam ranah komputasi citra atau pemrosesan wajah. Pada bidang pemrosesan wajah (*face processing*), pendeteksian wajah manusia (*face detection*) adalah salah satu tahap awal yang sangat penting dalam proses pengenalan wajah (*face recognition*). Sistem pengenalan wajah digunakan untuk membandingkan satu citra wajah masukan dengan suatu *database* wajah dan menghasilkan wajah yang paling cocok dengan citra tersebut Hanya saja

kelemahan yang sering terjadi pada penelitian pengenalan wajah yang telah ada sebelumnya adalah sistem yang berjalan tidak *real-time*, ekspresi wajah yang harus sama, serta proses komputasi yang terlalu berat.

Berawal dari proses komputasi yang berat, diperlukan terobosan baru yang mampu menyederhanakan permasalah tersebut. Salah satunya dengan memanfaatkan layanan komputasi awan (*cloud computing*). Sangatlah cocok apabila dikombinasikan antara sistem pengenalan wajah dan *cloud computing* dikarenakan keduanya memiliki relasi yang jelas, dimana *cloud computing* dapat menjadi solusi atas kendala yang dihadapi terkait proses komputasi yang terjadi pada sistem pengenalan wajah.

Augmented Reality (AR) merupakan suatu lingkungan yang memasukkan objek *virtual* kedalam lingkungan nyata. Oleh karena itu, unsur *reality* lebih diutamakan sehingga interaksi secara *real-time* pada sebuah sistem dapat memudahkan *end-user* dalam penggunaanya. Dengan demikian pemanfaatan teknologi AR pada sistem pengenalan wajah dapat memberikan informasi yang lebih menarik dan memberikan warna tersendiri dalam perkembangan teknologi.

Dengan aplikasi yang telah kami rancang dimana aplikasi tersebut dapat dengan mudah melakukan deteksi wajah manusia menggunakan kamera, oleh karena itu aplikasi yang kami rancang dengan judul Deteksi Identitas Berdasarkan Wajah

B. Rumusan Masalah

Permasalahan yang menjadi perhatian utama pada tulisan ini adalah Deteksi Identitas Berdasarkan Wajah pada perangkat *mobile* memanfaatkan layanan komputasi awan sebagai media pemrosesan, hasil pengenalan wajah tersebut ditampilkan dalam bentuk *augmented reality*.

C. Tujuan Penelitian

Tujuan dari penulisan penelitian ini adalah merancang sebuah sistem pengenalan wajah yang terdiri dari sistem pengolahan citra berbasis komputasi dan sistem menampilkan informasi dengan konsep *augmented reality* yang secara langsung. Sistem ini ditujukan untuk membantu mengenali wajah, mendapatkan informasi terkait wajah yang sesuai pada pemrosesan yang ditampilkan dalam bentuk *augmented reality.*

D. Metode Penelitian

Metode penelitian yang digunakan dalam penulisan skripsi ini antara lain :

1. Studi Literatur

   Pada tahap ini dilakukan pendalaman literatur yang berhubungan dengan pembuatan aplikasi android, dasar dari teknologi *augmented reality*, dasar sistem pengenalan wajah, penggunaan *augmented reality* pada android, dan berbagai literatur lain yang mendukung pembuatan sistem.

2. Perancangan Perangkat Lunak.

   Perancangan perangkat lunak dilakukan dan digambarkan dalam *Unified Modelling Language* (UML) sebagai bahasa standar pemodelan rancangan. Pendahuluan tahap ini dengan menyediakan *requirement* pendukung dalam pengembangan *augmented reality* pada android seperti perangkat *mobile* android, *Software Development Kit* (SDK) dan *Native Development Kit* (NDK). Pada proses pengembangan, dilakukan perancangan model tampilan *augmented reality* memanfaatkan wajah sebagai *marker* melalui pendeteksian wajah. Selanjutnya, menampilkan realitas tertambah dengan tanda berupa masker yang menutupi area wajah saat kamera android diarahkan pada wajah seseorang.

3. Integrasi Keseluruhan Perangkat Lunak.

   Berdasarkan perancangan yang telah dibuat, dilakukan integrasi dengan sistem keseluruhan yaitu pengenalan wajah. *Augmented Reality* yang diterapkan pada sistem ini memanfaatkan hasil pengenalan wajah dan ditampilkan pada layar perangkat Android.

4. Pengujian dan Analisa Perangkat Lunak.

   Pada tahap ini dilakukan pengujian terhadap sistem apakah sudah sesuai dengan rancangan awal yang telah dibuat sebelumnya atau tidak. Jika terdapat hal yang tidak sesuai akan dilakukan analisa kesalahan dan selanjutnya akan ditindak lanjuti dengan melakukan pembenaran pada sistem dengan mengulangi proses pengerjaan pada tahap tertentu sesuai dengan kesalahan yang terjadi.

5. Penulisan Skipsi

Tahap ini merupakan tahap terakhir yaitu melakukan penulisan dan detail tentang sistem yang telah dibuat.

E. Batasan Masalah

Adapun batasan masalah pada aplikasi yang akan dirancang adalah:

1. Aplikasi ini hanya menampilkan identitas seseorang dengan menggunakan kamera.
2. Aplikasi yang digunakan berbasis *Unity*

F. Sistematika Penulisan

Sistematika dari penulisan skripsi ini meliputi beberapa bab, yaitu :

BAB I PENDAHULUAN

Menjelaskan tentang latar belakang masalah, maksud dan tujuan yang akan dicapai dalam pelaksanaan skripsi ini. Penulis juga memberikan batasan-batasan masalah pada bagian akhir dan Sistematika Penulisan.

BAB II TINJAUAN PUSTAKA

Menguraikan teori yang erat hubungannya dengan cara kerja komponen yang digunakan dan penjelasan umum dalam pembuatan Aplikasi dengan *Unity*.

BAB III METODOLOGI PENELITIAN

Berisi tentang metode penelitian dengan menggunakan metode survey dimana ingin diketahui secara langsung sistem informasi, metode yang digunakan

untuk pengumpulan data adalah wawancara, pengamatan dan metode pustaka serta alat dan bahan penelitian.

BAB IV HASIL DAN PEMBAHASAN

Berisi tentang rancangan dan pembuatan aplikasi yang meliputi perangkat lunak serta cara kerja keseluruhan dan pengujian terhadap cara kerja software itu sendiri yaitu *basis path* (*prosedure* programnya).

BAB V PENUTUP

Bab ini merupakan bagian terakhir yang merangkum semua hasil yang diperoleh pada saat penyusunan tugas akhir dan saran-saran yang dapat membantu penyempurnaan pembahasan tugas akhir ini.

BAB II

TINJAUAN PUSTAKA

A. Penelitian Terdahulu

Peneliti Yunita Lusiani (2019) dalam penelitian yang berjudul *Rancang Bangun Aplikasi Wisata Bali Dengan Augmented Reality Menggunakan Metode Location Based Services Berbasis Mobile.* Dalam penelitiannya Yunita Lusiani mengimplementasikan *Augmented Reality* berupa *Location Based Services* dengan *platfrom* layar yang dapat digunakan untuk *Smartphone* berbasis operasi android objek wisata di Bali.

Fadli Raharjo (2018) dalam penelitiannya yang berjudul *Aplikasi Augmented Reality Pencarian Lokasi Wisata DKI Jakarta Berbasis Android*. Dengan teknologi *Augmented Reality* fadli raharjo bertujuan untuk memudahkan pengguna untuk mendapatkan informasi tempat-tempat wisata serta rute untuk menuju objek wisata yang dipilih yang berada di wilayah DKI Jakarta dengan menggunakan metode aplikasi pencarian lokasi berbasis *Location Based Services* dengan teknologi *Augmented Reality*.

Fajar (2019) dalam penelitiannya yang berjudul *Aplikasi Augmented Reality dengan Deteksi Objek*. Dengan teknologi *Augmented Reality* Fajar bertujuan untuk memudahkan pengguna untuk mendeteksi sebuah objek yang telah ditangkap oleh kamera pada hp dengan menggunakan aplikasi *Unity* dan *Open CV*

B. Konsep Sistem Pengenalan Wajah

Sistem pengenalan wajah merupakan konsep pengembangan ilmu yang berada dalam kategori biometrik. Biometrik adalah salah satu teknik pengenalan pola kebenaran identitas sesorang dengan menggunakan fisiologi atau perilaku manusia melalui teknologi komputer seperti pengenalan sidik jari, telapak tangan, wajah, iris dan lain-lain. Sistem pengenalan wajah menjadi salah satu diantaranya yang bertujuan untuk melakukan identifikasi wajah dengan membandingkan wajah dengan *database* wajah yang telah ada.

C. Wajah

Wajah adalah bagian terpenting dari siapa pun untuk dapat mengenali, kecuali pada kasus kembar identik. Wajah tergolong sebagai karakteristik fisik paling unik yang dimiliki setiap orang. Sementara itu manusia memiliki kemampuan bawaan untuk mengenali dan membedakan wajah yang berbeda, sedangkan teknologi komputer saat ini sedang mengejarnya untuk dapat melakukan hal serupa melalui perancangan sistem pengenalan wajah.

D. Klasifikasi Wajah

Pada sebuah sistem yang berkaitan dengan pengenalan wajah, ada beberapa hal penting yang perlu diperhatikan yakni klasifikasi wajah. Terdapat tiga jenis hasil klasifikasi wajah sebagai informasi hasil deteksi sistem:

1. Bukan Wajah.

   Klasifikasi jenis ini menyatakan bahwa sistem melakukan proses deteksi terhadap suatu objek. Proses deteksi tersebut ditemukan sebagai sebuah pola yang memberikan informasi bahwa hasil deteksi bukan sebuah wajah, melainkan hanya objek biasa.

2. Wajah yang tidak diketahui.

   Klasifikasi ini menunjukkan bahwa proses pengujian sistem memberikan informasi hasil deteksi merupakan wajah seseorang. Namun pencocokan wajah hasil deteksi tidak sesuai dengan *image* (wajah) yang terdapat pada sistem. Hasil penafsiran tersebut digolongkan sebagai wajah yang tidak diketahui.

3. Wajah yang dikenal

   Sedikit berbeda dengan sebelumnya, klasifikasi ini menemukan bahwa hasil deteksi wajah diketahui sebagai wajah yang dikenal atau mirip dengan beberapa wajah yang terdapat pada sistem.

E. Sistem Pengenalan Wajah

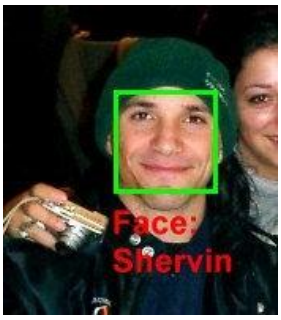
Face Recognition (pengenalan wajah) merupakan riset yang sangat digemari dalam lingkup *Computer Vision* dan *Biometrics*. Aplikasinya kian terasa dalam kehidupan sosial serta banyak dimanfaatkan dalam bidang robotika.

Face Recognition umumnya melibatkan dua tahap:

1. *Face Detection* (deteksi wajah), dimana *image* yang diperoleh diminta untuk menemukan wajah (ditampilkan di sini sebagai persegi hijau), maka

pengolahan citra membersihkan gambar wajah dan mengabaikan daerah disekitarnya.

2. *Face Recognition* (pengenalan wajah), dimana wajah yang dideteksi kemudian diproses dan dibandingkan dengan *database* wajah-wajah yang dikenal, untuk memutuskan siapa orang tersebut (ditampilkan di sini sebagai teks merah).



Gambar 2.1 Ilustrasi *Face Recognition*

Sebagai tahap awal yang sangat penting dalam proses pengenalan wajah, proses deteksi akan dihadapkan oleh tantangan-tantangan dari berbagai faktor yaitu :

1. Posisi wajah. Posisi wajah pada citra dapat bervariasi karena perbedaan posisi bisa tegak, miring, melihat kesamping atau sebagainya.
2. Komponen wajah yang tak terduga (bisa ada atau tidak ada) seperti kumis, jenggot dan kacamata.
3. Ekspresi wajah. Tampilan wajah sangat dipengaruhi oleh ekspresi seperti tersenyum, tertawa, marah, sedih dan sebagainya.

4. Citra wajah terhalang oleh objek lain misalnya pada citra yang berisi sekumpulan orang.
5. Kondisi pengambilan citra wajah yang dipengaruhi oleh lingkungan seperti intensitas cahaya, arah sumber cahaya bahkan lensa kamera.

F. Deteksi Tepi

Deteksi tepi adalah metode yang dapat mendeteksi garis tepi, yaitu garis yang memisahkan antara objek dengan latar belakang (*background*). Deteksi tepi merupakan pengolahan citra tingkat dasar yang diperlukan untuk melakukan pengolahan citra pada tingkat yang lebih tinggi. Deteksi tepi banyak digunakan dalam analisa pengolahan citra untuk berbagai macam tujuan. Berbagai macam metode deteksi tepi adalah *prewitt, Roberts, Canny,dan Sobel.* Setiap metode deteksi tepi mempunyai kekurangan dan kelebihan masing-masing. Algoritma *Sobel* memiliki kelebihan yaitu peka terhadap garis diagonal dibanding garis horizontal dan vertikal,sebalikanya algoritma *Prewitt* lebih peka terhadap garis horizontal dan vertikal dari pada garis diagonal (*Bin & Yeganeh*, 2012). Deteksi Canny mempunyai kelebihan paling optimal disbanding deteksi tepi yang lain tapi dalam proses komputasinya memakan waktu yang cukup lama. (*Chouhan & Shukla*,2011).

1. Deteksi Tepi *Roberts*

Metode pendektesian tepi dengan mencari perbedaan (*differensial*) pada arah horisontal dan perbedaan pada arah vertikal, dengan ditambahkan proses konversi biner setelah dilakukan perbedaan. Agar mendapatkan tepi-

tepi yang lebih baik, maka konversi biner dilakukan dengan meratakan distribusi warna hitam dan putih. Kernel filter yang digunakan oleh Deteksi Tepi *Roberts* adalah :

$$H = \begin{bmatrix} -1 & 1 \end{bmatrix} \quad \text{dan} \quad V = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

2. **Deteksi Tepi *Prewitt***

Metode ini adalah pengembangan metode *robert* dengan menggunakan filter HPF (*High-Pass Filter*) yang diberi satu angka nol penyangga. Metode ini mengambil prinsip dari fungsi *laplacian* yang dikenal sebagai fungsi untuk membangkitkan HPF. Sehingga tepi-tepi yang dihasilkan lebih banyak dari metode *robert*. *Kernel Filter* yang digunakan oleh Deteksi Tepi *Prewitt* adalah :

H =

-1	0	1
-1	0	1
-1	0	1

V =

-1	-1	-1
0	0	0
1	1	1

3. Deteksi Tepi *Sobel*

Sobel merupakan algoritma deteksi tepi yang menggunakan *Sobel* Operator, yaitu sepasang kernel berupa matriks berukuran 3 x 3 untuk mendeteksi tepi vertikal dan horizontal. Deteksi tepi *Sobel* merupakan pengembangan deteksi tepi *Roberts*. Kernel filter yang digunakan dalam metode *Sobel* ini adalah:

H =

-1	0	1
-2	0	2
-1	0	1

V=

-1	-2	-1
0	0	0
1	2	1

Kombinasi kedua kernel tersebut digunakan untuk menghitung jarak absolut sebuah *gradient* namun bisa juga diaplikasikan secara terpisah untuk menghitung masing-masing proses vertikal dan horizontal. Untuk menghitung jarak *gradient*, Berikut contoh penghitungan Deteksi tepi *Sobel*:

Menggunakan matrik seperti dibawah dan diambil matrik 3x3 untuk penghitungan-nya.

136	127	142	135	143
143	138	125	145	138
157	145	139	133	146
165	154	157	145	143
174	158	167	162	158

Gx=136*(-1)+127*(-2)+142*(-
1)+143*(0)+138*(0)+125*(0)+157*(1)+145*(2)+139*(1)
Gx = -136+(-254)+(-142)+0+0+0+157+290+139= 54

Gy=136*(-1)+127*(0)+142*(1)+143*(-2)+138*(0)+125*(2)+157*(-1)+145*(0)+139*(1)

Gy = -136+0+142+(-286)+0+250+(-157)+0+139= -48

Maka nilai dari G adalah

$$| \; | \quad \sqrt{\qquad\qquad}$$

$$| \; | \quad \sqrt{\qquad\qquad}$$

$$| \; | \quad \sqrt{\qquad\qquad}$$

G=72,24957

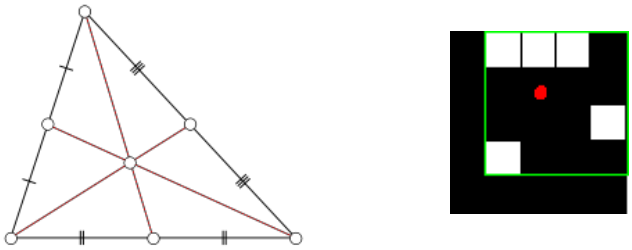
Hasil Dari Proses Deteksi Tepi dari matrik diatas adalah :

136	127	142	135	143
143	72.2	125	145	138
157	145	139	133	146
165	154	157	145	143
174	158	167	162	158

4. Centroid

Sejarah penelitian *centroid* dibilang sudah cukup lama yang dimulai dari (P.*Guldin*) menyatakan salah satu theorem dari *centroid* bahwa apabila sebuah arc atau busur ber-revolusi dan tidak berpotongan maka garis radius akan terbentuk menjadi area dari *centroid*. Setelah itu penelitian *centroid* dilanjutkan dengan mengabungkan dengan sebuah bentuk seperti didalam buku (*Altshiller-Court*) membahas bagaimana mendapatkan

nilai *centroid* didalam segitiga yang sekarang biasa disebut titik berat segitiga. Sedangkan didalam buku (*Paul*) menyatakan *centroid* adalah *centre-of- gravity* atau *center-of-mass* didalam buku itu *centroid* digunakan untuk menghitung titik pusat dari sebuah polygon yang mempunyai titik lebih daripada segitiga. Penelitian sekarang lebih tentang bagaimana memgimplementasikan *centroid* ke dalam sebuah objek. Didalam *binary image* atau gambar hitam putih *centroid* biasa disebut juga *center of gravity* yang artinya titik berat berdasarkan intensitas pixel didalam gambar.



Gambar 2.1 *Centroid* dari Segitiga *& Centroid dari Binary Image*

Centroid dari binary image berdasarkan (*George*,2001) menggunakan :

{ x, y } = {M10/M00, M01/M00 }

M00 adalah total jumlah seluruh nilai 1 didalam gambar. M10 adalah jumlah baris yang berisi nilai 1M01 adalah jumlah kolom yang berisi nilai 1

Contoh penghitungan *centroid*

0		2
0	0	0
1	0	1
1	0	0

M00 = 1 +1 +1 = 3

M = (1,0)(1,2)(2,0) M10 =1+1+2 = 4

M01 =0+0+2 = 2

{ x, y } = {4/3,2/3}

{ x, y }={1.3,0.6} Dibulatkan

{x,y}={1,1} maka *centorid* dari matrik diatas adalah :

0	0	0
1	0	1
1	0	0

Penelitian terkait *centroid* dilakukan oleh (Sidiropoulus,2011) dengan judul *Content-based binary Image retrieval using the adaptive hierarchical density histogram* (AHDH) menggunakan *centroid* sebagai pengkompresan binary *image* yang menhasilkan keakuratan 38% tertinggi kedua setelah AHDH dengan selisih hanya 2%. Kurangnya akurasi dikarenakan objek *binary image* yang digunakan hampir sama satu dengan yang lain, Sedangkan penelitian yang dilakukan (*Jae-young Hong*,2013) dalam penelitian *centroid menthod : an alternative method of determining coronalscoliosis. A comparative study versus Cobb Method in the degenerative spine*. Penelitian tersebut membandingkan metode *centroid* dengan metode cobb dalam menentukan *coronalscoliosis*. Metode *centroid* lebih unggul dalam menentukan *coronalscoliosis* daripada metode *Cobb*, sedangkan metode *cobb* lebih cocok ke *degenartive*. Hasil tersebut disebabkan *centroid* punya karakteristik untuk menentukan titik tengah suatu objek dengan

menghitung intesitas pixel didalamnya. Sedangkan (*Janusz*,2014) menggunakan *centroid triangular* untuk komputasi *Gaussian type-2 fuzzy set* mengurangi komputasi 50% dibanding menggunakan *centroid* normal.

5. Geometric mean

Geometric mean dicetuskan pertama kali oleh (*Kenney*) didalam bukunya berisi : *Geometric mean* atau rata-rata ukur adalah rata-rata dari jumlah suatu geometri atau biasa didenifisikan sebagai *nth-root* dari produk *n* angka. Contoh-nya misal dari sekumpulan data { } m $\{X_i\}_{i=1}^{N}$ *metric meannya :*

$$\left(\prod_{i=1}^{N} Xi\right)^{1/N}$$

Geometric mean dapat digunakan dalam berbagai aspek pengolahan data. Contoh beberapa dari aplikasi *geometric mean* : *aspect ratios* yang biasa digunakan untuk mengecilkan dan membesarkan sebuah video oleh (Cinema *Source*™,2001) dimana *geometric mean* digunakan untuk membandingkan layar video awal dengan layar yang dipakai, dan dibidang finacial yang dilakukan oleh (Rowley,1982) *geometric mean* digunakan untuk mengetahui rata-rata pergerakan dari sebuah index saham.

Geometric mean sering digunakan untuk mencari perbedaan dari dua benda berbeda yang setiap benda memiliki property dan jumlah yang berbeda . contoh angka 2 dan 8, maka penghitungannya seperti akar

kuadrat $\sqrt{2 \cdot 8} = 4$.Contoh lain-nya, nilai *geometric mean* dari 4, 1, dan 1/32 adalah

$$\sqrt[3]{4 \cdot 1 \cdot 1/32} = 1/2$$

Penelitian (Prakash) tentang *geometric centroid and their relative distances for off-line signature verification* menggunakan *geometric mean* sebagai indentifikasi tanda tangan seseorang dan *centroid* sebagai pembantu menghasilkan tingkat akurasi 79% dari 2250 tanda tangan dan 75 orang. *Database* yang digunakan terlalu besar karena setiap orang harus melakukan 30 tanda tangan disebabkan apabila ukuran tanda tangan berubah akan mengalami kesalahan indentifikasi. Sedangkan penelitian (*Sidram*) *a novel shape signature of geometric mean of segmented centroid distance function to track object through fourier descriptor* yang menggunakan *geometric mean* sebagai pengenalan objek yang bergerak dalam video menghasilkan 97% akurasi-nya apabila digabung dengan *centroid*. Kekurangannya adalah pengenalan objek apabila bentuk dari objek hampir sama maka akan di indentifikasi sama seperti mobil dan *taxi, taxi* mempunyai tinggi lebih daripada mobil karena mempunyai lampu diatas tetapi program tersebut akan mengenali itu sebuah mobil. Penelitian yang terbaru adalah (*Koteswar,2014*) tentang automatic *image co-segmentation using geometric mean saliency* dimana menghasilkan nilai segmentasi setiap objek sekitar 92.2%, kekurangannya adalah parameter setiap objek harus di ubah untuk menghasilkan yang terbaik.

G. *Knowledge-base method*

Metode ini menggunakan aturan dasar yang umumnya digunakan manusia untuk menentukan apa saja yang membentuk wajah. Sebagai contoh, wajah seseorang pada sebuah citra biasanya memiliki dua buah mata yang simetris, sebuah hidung, dan mulut. Relasi antar organ tersebut dijadikan sebagai sebuah tolak ukur jarak atau posisi.

Masalah yang muncul pada metode ini adalah kesulitan menerjemahkan pengetahuan manusia kedalam aturan yang dipakai. Jika aturan terlalu detail, maka akan sering gagal mendeteksi wajah untuk menyesuaikan aturan tersebut. Jika aturan terlalu umum, akan menghasilkan terlalu banyak *false positive* (salah perkiraan). Kesulitan semakin bertambah jika proses deteksi dilakukan pada posisi wajah yang bervariasi. Oleh karena itu, metode ini umumnya dapat bekerja dengan baik pada wajah *frontal* dan tegak.

*1. **Template matching method***

Metode ini akan melakukan penyimpanan pola wajah standar untuk mendeskripsikan wajah secara keseluruhan maupun bagian-bagiannya. Saat pendeteksian, akan dihitung korelasi antara citra input dengan citra pola wajah yang tersimpan sebelumnya.

2. ***Appearance-based method***

Pada metode ini, citra wajah dipelajari melalui proses *training* menggunakan satu set data pelatihan yang berisi contoh-

contoh wajah. Kemudian hasil *training* akan digunakan untuk mendeteksi wajah. Secara umum metode ini menggunakan teknik analisa statistik dan *machine learning* untuk menemukan karakteristik wajah yang relevan dengan citra wajah inputan. Proses ini dikenal dengan nama metode *eigenfaces.* Pada perancangan sistem pengenalan wajah berbasis *cloud* ini akan digunakan metode tersebut.

Dengan demikian sistem pengenalan wajah merupakan bagian akhir dari sistem deteksi dan pengenalan wajah. Pada sistem deteksi wajah dilakukan proses pengolahan citra dan deteksi wajah. Sistem deteksi bertugas mengolah citra dan menyesuaikan dengan *database* dan hanya melewatkan citra yang terdapat wajah didalamnya, serta melakukan klasifikasi wajah dan non-wajah terhadap citra masukan. Sedangkan sistem pengenalan wajah bertugas mengenali citra masukan sebagai wajah orang yang ada dalam *database* atau tidak. Citra yang merupakan wajah dalam *database* akan ditentukan sebagai wajah ke-n. Wajah ke-n tersebut akan melalui proses pengenalan yang terdiri dari dua bagian, yaitu bagian ekstraksi fitur dan klasifikasi (pengenalan). Bagian ekstraksi fitur bertujuan mereduksi data sehingga dihasilkan fitur penting yang mewakili citra wajah, pada bagian klasifikasi akan mengenali fitur wajah tersebut sebagai orang yang dikenali (sesuai *database*) atau tidak.

H. *Augmented Reality*

Menurut Bimber dan Raskar (2017), *Augmented Reality* berarti mengintegrasikan informasi sintesis kedalam lingkungan nyata. Menurut penjelasan *Haller Dkk* (2017), riset *Augmented Reality* bertujuan untuk mengembangkan teknologi yang mengiakan penggabungan secara *real-time* terhadap digital *content* yang dibuat oleh komputer dengan dunia nyata. Tidak seperti teknologi *Virtual reality* (VR) yang membawa user sepenuhnya ke dalam lingkungan sintesis, *Augmented Reality* memperbolehkan pengguna melihat objek maya tiga dimensi maupun video yang diproyeksikan terhadap dunia nyata. *Augmented Reality* pada dasarnya adalah sebuah konsep yang mencitrakan gambar tiga dimensi yang seolah nyata. Proses ini bisa diperinci menjadi beberapa proses dan komponen.

Untuk mencitrakan gambar tiga dimensi tersebut, sistem *Augmented Reality* terlebih dahulu harus melakukan penglihatan atau vision terhadap lingkungan yang padanya akan dicitrakan objek virtual. Kemudian, dilakukanlah proses tracking terhadap objek spesifik yang menetukan letak citraan objek virtual tersebut. Kemudian, objek tersebut akan dikenali, atau dianalisis. Setelah dikenali dan dianalisis posisi dan oriientasinya, maka komputer akan melakukan proses pencitraan objek tersebut, dan akan tampak pada perangkat *display*. Komponen penting yang harus ada adalah:

1. Perangkat tampilan (*Display*)
2. Alat *tracking* (pencarian)

3. Peralatan input
4. Perangkat komputer

Perlengkapan tampilan digunakan untuk menampilkan informasi gambar atau objek tiga dimensi yang dicitrakan terhadap dunia nyata tempat user melihat. Perlengkapan tampilan terbagi menjadi tiga jenis, yakni *Head Mounted Display*, Handheld *Display*, dan Spatial *Display*. *Head Mounted Display* adalah perlengkapan tampilan yang dikenakan di kepala *user* dan digunakan sebagai 'kecamata' untuk melihat dunia nyata, yang telah digabungkan dengan objek virtual yang telah diregistrasikan dalam sistem, *Handheld Display* adalah perlengkapan ringkas yang dibawa-bawa ke mana saja dan dapat dimuat di tangan, contohnya adalah *smartphone* dan android *phone. Spatial Display* adalah sistem pencitraan yang menggunakan proyektor digital untuk mempetakan informasi grafis pada objek fisik. Hal yang paling membedakan *Spatial Display* adalah bahwa pencitraannya tidak terasiosiasi dengan setiap individu *user*, namun secara berkelompok.

Tracking biasanya dilakukan dengan teknologi-teknologi menangkap gambar, misalnya dengan kamera digital, sensor optis lainnya, GPS, kompas dan lain sebagainya. Selain itu, alat tracking yang sekarang meningkat popularitanya adalah *webcam*, karena praktis, kecil, mudah dibawa dan diatur untuk dijalankan. Peralatan input hingga sekarang ini masih banyak menjadi objek penelitian. Hingga saat ini, alat yang digunakan mencakup alat '*pinch glove'*, tongkat bertombol atau peralatan handheld seperti smartphone. Perankat komputer, terutama dengn CPU yang kuat dan jumalah RAM yang cukup besar untuk

memproses gambar yang ditangkap. Sistem yang digunakan untuk mobilitas biasanya menggunakan laptop yang dilengkapi dengan *webcam*, sementara untuk yang bersifat diam menggunakan *workstation* dengan kartu grafis yang kuat.

Implementasi perhitungan pada teknologi *Augmented Reality* proses ini dilakukan perhitungan untuk mendapatkan koordinat wisata sehingga bisa ditampilkan dalam layar *handphone*

a. Perhitungan sudut *azimuth*

Dalam sistem navigasi terdapat istilah azimuth yaitu sudut antara utara manetis (nol derajat) dengan titik/ sasaran yang kita tuju relatif terhadap sudut utara. Sudut *azimuth* pada masing-masing titik dihitung berdasarkan sudut arctangent antara perbedaan lattitude dengan lokasi pengguna dibagi dengan perbedaan longitude. Setelah mendapatkan sudut *azimuth* pada setiap titik langkah berikutnya adalah menemukan lokasi horizintal pada layar untuk menampilkan titik-titik lokasi wisata. Sudut *azimuth* di-*update* terus menerus sehingga bisa mewakili posisi imajiner di depan pengguna saat pengguna melakukan rotsi atau memutar layar *handphone* kekiri atau kekanan.

b. Perhitungan sudut inklinasi

Sudut inklinasi merupakan sudut vertikal yang dibentuk antara titik tempat pengguna menggunakan aplikasi dengan titik ketinggian suatu lokasi dapat diketahui secara akurat. Setelah didapatkan ketinggian masing-masing titik langkah berikutya adalah menetukan sudut vertikal antara 2 titik tersebut dengan rumus trigonometri. Sensor akselerometer melakukan pembacaan terhadap orientasi *handphone* yang dipegang pengguna. Jika pengguna

memegang dengan posisi layar menghadap depan dan rata dengan tanah maka ikon wisata akan ditampilkan sejajar terhadap horizon. Namun jika pengguna memegang *device* pada posisi 45 derajat kebawah maka ikon wisata muncul pada layar bagian atas. Sumbu- X dalam representasi dunia nyata adalah garis horizontal, sumbu-Y merupakan garis vertikal dan sumbu-z mengarah kearah depan layar sedangkan area di belakang layar memiliki nilai Z negatif.

I. *Smartphone*

Ponsel cerdas (bahasa inggris: *smartphone*) adalah telepon genggam yang mempunyai kemampuan dengan penggunaan dan fungsi yang menyerupai komputer. Belum ada standar pabrik yang menentukan arti ponsel cerdas. Bagi beberapa orang, ponsel cerdas merupakan telepon yang bekerja menggunakan seluruh perangkat lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. Bagi yang lainnya, ponsel cerdas hanyalah merupakan sebuah telepon yang menyajikan fitur canggih seperti surel (surat elektronik), internet dan kemampuan membaca buku *electronik* (*e-book*) atau terdapat papan ketik (baik sebagaimana jadi maupun dihubung keluar) dan penyambung VGA. Dengan kata lain, ponsel cerdas merupakan komputer kecil yang mempunyai kemampuan sebuah telepon.

Singkatnya, *smartphone* adalah perangkat yang memungkinkan anda melakukan panggilan telepon, sekaligus memiliki fitur yang dimasa lalu hanya bisa ditemukan pada personal *digital assistant* (PDA) atau komputer- seperti kemampuan untuk mengirim dan menerima *e-mail* dan editing dokumen.

Ponsel terutama digunakan untuk menelpon, sementara PDA digunakan sebagai semacam asistensi digital pribadi atau digital organizer. PDA dapat menyimpan info kontak dan agenda harian serta bisa disinkronisasi dengan komputer. Perkembangan ini lantas mendorong terciptanya apa yang kini dikenal sebagai *smartphone*.

J. Bahasa Pemrograman C#

C# (baca: C *sharp* atau *see sharp*) sering dianggap sebagai bahasa penerus C++ atau versi canggih dari C++, karena ada anggapan bahwa tanda # adalah perpaduan dari 4 buah tanda tambah yang disusun sedemikian rupa sehingga membentuk tanda pagar. Akan tetapi, terlepas dari benar tidaknya anggapan tersebut, C# adalah sebuah bahasa pemrograman yang sangat menjanjikan. C# adalah sebuah bahasa pemrograman yang beriorentasi pada objek yang dikembangkan oleh *Microsoft* dan menjadi salah satu bahasa pemrograman yang mendukung. *Net programming* melalui *Visual Studio*.

C# didasarkan pada bahasa pemrograman C++, C# juga memiliki kemiripan dengan beberapa bahasa pemrograman seperti *Visual Basic, Java, Delphi*, dan tentu saja C++. C# memiliki kemudahan *syntax* (cara penulisan) seperti *Visual Basic*, dan tentu saja ketangguhan seperti *Java* dan C++. Kemiripan-kemiripan ini tentunya memudahkan programmer dari berbagai latar belakang bahasa pemrogrman tidak perlu waktu yang lama untuk menguasainya, karena bagaimanapun juga C# lebih sederhana dibandingkan bahasa-bahasa pemrograman seperti C++ dan *Java*.

C# didesain oleh program *designer dari microsoft, Anders Hajlsberg*. Sebelum bekerja pada *Microsoft*, Anders bekerja di *Borland*, tempat dia menulis *Pascal compiler*. Sebelum mengembangkan C# *Anders* pernah mengembangkan J++ untuk *Microsoft*. Setelah itu Anders mengembangkan C# dan *Common Language Runtime* yang merupakan mesin *Virtual dan Runtime library* yang merupakan salah satu tiang utama teknologi. *Net. Anders* mengetahui berbagai macam kekurangan pada bahasa C++, *Delphi, Java, dan Smaltalk*, karena itu Anders menciptakan bahasa C# yang lebih tangguh. Hal ini juga menjelaskan mengapa C# memiliki kemiripan dengan beberapa bahasa tersebut.

C# bergantung pada CLR yang juga merupakan sumber library bagi program. Semua program C# memerlukan CLR (berarti juga memerlukan .*Net Framework*) untuk dapat dijalankan. Sama halnya dengan Visual Basic 6 yang memerlukan *runtime library* tertentu untuk dapat dijalankan. Bahasa C# dapat digunakan untuk menciptakan aplikasi *windows, console, class* yang dapat digunakan kembali, dan aplikasi *web*.

Ada beberapa alasan kuat yang mendasari pmilihan bahasa C# untuk mengembangkan aplikasi-aplikasi, yaitu:

a. C# benar-benar beriorantasi objek

   C# adalah bahasa yang benar-benar objek *oriented*. Ini dapat dilihat dari kemampuan C# dalam pembentukan objek, *class*, melakukan *encapsulation, inheritance, dan polymorphsm* dengan mudah.

b. C# sangat sederhana

Bahasa C# bersifat sederhana karena didasarkan pada bahasa C dan C++ bahkan bahasa Java. Tetapi C# lebih sederhana dari bahasa-bahasa tersebut karena C# dibuat dengan menghilangkan kelemahan-kelemahan dari bahasa-bahasa yang mendasarinya.

c. Mampu membuat berbagai aplikasi

Dengan C# kita dapat membuat berbagai macam aplikasi, mulai dari aplikasi *console*, pengolahan kata, *form web*, dan lain-lain.

d. Efisien

Bahasa C# merupakan bahasa yang mengandalkan library yang sangat lengkap, karena itu bahasa C# hanya memiliki sedikit *keywords*. Jadi para pengembang dapat mengingatnya dan memahami kegunaanya dengan baik.

e. C# bersifat modular

Bahasa C# sangat modular, tiap *class* disimpan dalam *namespace* yang dapat dimanfaatkan kembali oleh program lain yang membutuhkannya.

K. Android (SDK)

Android SDK (*Software Development Kit*) adalah tools API (*Aplication Programming Interface*) yang diperlukan untuk memulai pengembangan suatu aplikasi pada *platform android* menggunakan bahasa pemrograman java

Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang direkase oleh *google*. Saat ini disediakan Android SDK sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada *platfrom* aplikasi netral, Android memberi anda

kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan aplikasi bawaan *Handphone/smartphone*. Beberapa fitur Android adalah :

a. *Framewor*k aplikasi yang mendukung penggantian komponen dan *reusable*
b. *Dalvik Virtual Machine* dioptimalkan untuk perangkat *mobile*
c. *Integrated browser* berdasarkan *engine open source Webkit*
d. *Grafis* yang dioptimalkan dan didukung oleh *libraries grafis* 2D, *grafis* 3D berdasarkan spsifikasi *opengl ES* 1,0
e. SQLite untuk penyimpanan data/*database*
f. Media *suport* yang mendukung audio, video, dan gambar
g. *Bluetooth, EDGE,3G,WiFi (tergantung hardware*)
h. Kamera, GPS,kompas dan *accelerometer* (tergantung hardware)

Lingkungan *Development* yang lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging*, profil dan kinerja memori dan *plugin* untuk IDE *eclipse.*

L. JDK (*Java Development Kit*)

Java Development Kit merupakan perangkat lunak yang digunakan untuk melakukan proses kompilasi dari kode java menjadi *bytecode* yang dapat dimengerti dan dapat dijalankan oleh *Java Runtime Environment*.

Java Development Kit wajib terinstall pada komputer yang akan melakukan proses pembuatan aplikasi berbasis java. Namun *Java Development Kit* tidak wajib terinstall dikomputer yang akan menjalankan aplikasi yang dibangun menggunakan Java.

Seperti namanya, JDK adalah semacam kotak peralatan (kit) yang digunakan untuk *Development.* JDK ini berguna saat anda menulis *code* program. Seperti halnya JRE, JDK juga memiliki JVM di dalamnya.

a. Java yaitu program untuk meng-*compile* kode sumber

b. Java, yaitu program untuk meluncurkan aplikasi.

M. *Unity*

Unity ini adalah sebuah aplikasi berbasis *Multi Platform, Munti Platform* adalah aplikasi yang dapat mempublish ke banyak format tipe *file*, contohnya : *exe*, apk,dan lain-lain. *Unity* perlu lisensi untuk dapat dipublish ke *platform* tertentu. Tetapi *Unity* menyediakan untuk *free user* dan bisa di *publish* dalam bentuk Standalone (.*exe*) dan web. Untuk saat ini *Unity* sedang mengembangkan berbasis AR (*Augment Reality*). *Unity* ini tidak hanya untuk membuat game 3D saja, namun 2D juga.

Unity Technologies dibangun pada tahun 2004 oleh *David Helgason (CEO), Nicholas Francis (CCO), dan Joachim Ante (CTO) di Copenhagen*, Denmark setelah game pertama mereka *GooBall*, gagal lagi dalam meraih sukses. Ketiganya menyadari nilai sebuah *engine* dan *tool* dalam sebuah pengembangan game dan berencana untuk membuat sebuah engine yang dapat digunakan oleh semua dengan harga terjangkau. *Unity Technologies* mendapat bantuan dana dari *Sequoia Capital, WestSummit Capital, and iGlobe Partners*.

Kesuksesan *Unity* terletak pada fokus mereka untuk memenuhi kebutuhan *indie developer* yang tidak dapat membangun game engine mereka sendiri atau membeli lisensi *game engine* yang terlalu mahal. Fokus perusahaan ini adalah

"*Democratize game development*" atau diterjemahkan sebagai "Demokrasi Pembangunan Game" dan membuat sebuah pembangunan game baik 2D maupun 3D bisa dicapai oleh banyak orang. Oleh siapa pun. Pada tahun 2008, *Unity* melihat kebangkitan *iPhone* dan menjadi *game engine* pertama yang melakukan dukungan penuh pada *platform* tersebut. *Unity* sekarang di gunakan oleh 53.1% *developers* (termasuk *mobile game developer*) dengan ratusan game yang dirilis baik untuk iOS maupun Android. Pada tahun 2009, *Unity* mulai meluncurkan produk mereka secara gratis. Jumlah *developer* yang mendaftar melonjak drastis sejak pengumuman tersebut. Pada April 2012, *Unity* mencapai popularitas yang sangat tinggi dengan lebih dari 1 juta *developer*.

Unity sangat mampu melihat berbagai peluang dan perubahan. Hal inilah yang menjadikannya sebagai *game engine* "termurah" yang paling banyak digunakan oleh seluruh orang di dunia. Ingat, *Unity* bisa digunakan untuk perorangan dan tidak selalu harus digunakan oleh sebuah *studio game* yang berjumlah ratusan orang. Tampaknya, demokrasi yang diusung sebagai slogan *Unity* memang benar adanya. Apalagi semenjak dirilisnya *Unity* dengan lisensi free.

N. *Marker*

Marker adalah object dunia nyata yang sudah di inisialisasi sebelumnya, yang nantinya akan di kenali oleh *system* sebagai wadah untuk memunculkan *object* atau area interaksi. Dalam pembuatan marker dalam hal ini markerless diperlukan sebuah file gambar.JPG yang nantinya akan di *upload* ke *Vuforia*,

marker yang telah di-*upload* akan dinilai kualitasnya oleh *system*. Berikut adalah metode dan alur pengenalan pola gambar (Mario Fernando, 2013).

1. Metode Pengenalan Pola Gambar

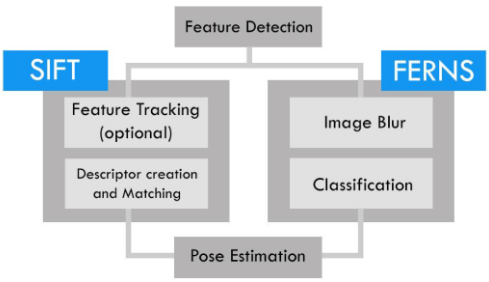
   Qualcomm sebagai salah satu pengembang *Augmented Reality* melakukan proses pendeteksian marker menggunakan pengenalan pola gambar. Metode yang digunakan dalam QCAR adalah *Natural Features Tracking* dengan metode *FAST Corner Detection* yaitu pendeteksian dengan mencari titik-titik (*interest point*) atau sudut-sudut (*corner*) pada suatu gambar. Istilah corner dan *interest point* sering digunakan secara bergantian. Pertama-tama dilakukan pendeteksian tepi (*edge*), keudian dilakukan analisa tepi untuk mendapatkan pendeteksian sudut (*corner*) secara tepat. *Algoritma* ini kemudian dikembangkan, sehingga deteksi tepi secara *eksplisit* tidak lagi giperlukan. Misalnya mendeteksi kelengkungan dalam gradient gambar. Pada saat itu juga ternyata bagian-bagian yang tidak berbentuk sudut (*corner*) terdeteksi juga sebagai bagian dari gambar, misalnya titik-titik kecil pada latar belakang gelap mungkin terdeteksi. Titik – titik ini yang disebut interest point namun istilah corner tetap digunakan.

2. Natural Feature *Tracking and Detection*

   Ponsel adalah *platform* kinerja rendah dengan sumber daya yang sangat terbatas dibandingkn dengan komputer. Akibatnya, keterbatasan ponsel dalam setiap aspek harus diperhitungkan ketika mengembangkan sebuah teknologi AR. Banyak teknologi AR yang dirancang untuk ponsel, menggunakan penanda berbasis pelacakan untuk menimbulkan estimasi dan ekstraksi.

Pendekatan ini bekerja dengan cukup baik dan menyederhanakan proses pelacakan, sehingga bagi pengguna akhir, lingkungan harus dipersiapkan terlebih dahulu. Hal ini membuat prosedur pengembang menjadi rumit tetapi membawa fleksibilitas sehingga setiap benda dengan tekstur yang cukup rinci dapat dengan mudah dilacak.

Dalam metode ini informasi yang diperlukan untuk tujuan pelacakan dapat diperoleh dengan cara *optical-flow* berbasis pencocokan *template* atau korespondensi fitur. *Optical flow atau aliran optic* adalah pola gerakan jelas suatu benda, permukaan, dan tepi dalam adegan visual yang disebabkan oleh gerakan reatif antara pengamat (mata atau kamera) dan adegan. Korespondensi fitur bekerja lebih baik dan lebih efektif daripada pencocokan template karena mereka bergantung pada pencocokan fitur *local*. Mengingat korespondensi tersebut, *pose* secara kasar dapat dihitung dengan estimasi yang kuat yang membuatnya cukup *sensitive terhadap oklusi persial, blur, refleksi*, perubahan skala, kemiringan, perubahan iluminasi atau kesalahan pencocokan. Salah satu unsur diterapkan pendekatan pelacakan fitur alami didasarkan pada versi modifikasi dari fitur SIFT dan FERN *descriptor*. SIFT sanagat baik dalam mengekstrak tetapi *prosesor* intensif bekerja karena komputasi, sementara FERN menggunakan klasifikasi fitur yang cepat tetapi membutuhkan kapasitas memori yang besar. Dalam hal ini pelaksanaan SIFT dan FERN telah terintegrasi, tetapi dengan signifikan modifikasi untuk membuat sebuah *system* pelacakan cocok untuk ponsel.



Gambar 2.2. Alur SIFT dan FERNS

O. *Vuforia* SDK (*Sofware Development Kit*)

Vuforia SDK adalah *sofware development Kit* berbasis AR yang menggunakan layar perangkat *mobile* sebagai "Lensa ajaib" atau kaca untuk melihat kedalam dunia *augmented Reality* dimana dunia nyata dan virtual muncul berdampingan. Aplikasi ini membuat *preview* kamera secara langsung pada layer *smartphone* untuk mewakili pandangan dari dunia fisik. *Objeck* 3 dimensi akan nampak secara langsung di layar *smartphone*, sehingga akan terlihat *objeck* 3 dimensi berada didalam dunia nyata.(*Febriyan M.Fajar,2014)*

Vuforia SDK mendukung berbagai 2D dan 3D jenis sasaran termasuk *markerless Image Target, konfigurasi 3D Multi-Target,* dan bentuk beralamat *Fidusia Marker* dikenal sebagai Bingkai *Marker* . Fitur tambahan dari SDK termasuk lokal *Occlusion Deteksi* menggunakan *Virtual Buttons*, pemilihan target gambar runtime, dan kemampuan untuk membuat dan mengkonfigurasi ulang sasaran set pemrograman saat *runtime*. *Vuforia* menyediakan Antarmuka *Application Programming (API) di C++, Java, Objective-C , dan* bahasa *Net* . Melalui perpanjangan untuk mesin permainan *Unity.* Dengan cara ini , SDK mendukung pengembangan asli untuk *iOS* dan Android sementara juga

memungkinkan pengembangan aplikasi AR di *Unity* yang mudah dibawa ke kedua *platform* . Aplikasi AR yang dikembangkan menggunakan *Vuforia* karena itu kompatibel dengan berbagai perangkat *mobile* termasuk iPhone (4/4S), iPad, dan ponsel *Android dan tablet yang menjalankan OS Android* versi 2.2 atau yang lebih besar dan ARMv6 atau 7 prosesor dengan FPU (*Floating Point Unit*) kemampuan pemrosesan.

Vuforia SDK berisi banyak fitur menarik, berhubungan dengan AR dunia:

1. **QR code mengakui:** Perangkat lunak ini dapat memindai dan mengenali QR code.
2. **Gambar mengenali:** *Vuforia* SDK dapat memindai dan mengenali *Image*s, ada beberapa keterbatasan kemampuan memindai nya, misalnya, sudut atau bentuk melengkung dapat lebih sulit untuk mengenali, dengan cara yang sama warna yang dibuat menggunakan nuansa yang sama dari warna yang sama.Namun *software* ini benar-benar kuat.
3. ***Quadrilaterals*** **Kecil objek** : Perangkat lunak ini juga mampu mengenali benda kecil, menggunakan wajah mereka sebagai pembuat.
4. **Kecil objek *silinder:*** Salah satu rilis terbaru dari perangkat lunak memperkenalkan kemungkinan menggunakan benda *silinder* kecil sebagai penanda. Saat ini aku tidak bereksperimen namun dengan fitur ini dan aku tidak yakin bagaimana kuat dan dapat diandalkan itu, tapi saya akan mencoba untuk yakin.
5. **Kata pengakuan :** *Vuforia* SDK juga mampu mengenali kata *Inggris* tertentu. Saya menguji fitur ini banyak, dan itu sangat menarik, ada beberapa

batasan tentang kaligrafi, tetapi jika kata tersebut dicetak atau digital di layar komputer *software* tidak menunjukkan masalah.

6. ***Cloud* penyimpanan :** Perangkat lunak ini juga memungkinkan penggunaan data awan disimpan. Fitur ini bisa sangat berguna bagi orang-orang yang tidak memiliki masalah kinerja pada ponsel mereka.
7. **Multi-target :** *Vuforia* dapat memindai dan mengenali lebih dari satu target sekaligus, biasanya ada batasan sampai 5 target sekaligus, tapi *software* ini sangat lincah dan itu tergantung pada jenis perangkat keras yang digunakan.
8. **Tombol virtual :** perangkat lunak memiliki *build*-in maya manajemen tombol. Fitur ini bisa sangat menarik untuk menambahkan lebih banyak interaksi antara pemain dan spidol.

P. UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) menurut Adi nugroho (2010:6), "*Unified Modelling Language* adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berpradigma beriorentasi objek",

Metode *Unified Modelling Language* (UML) menggunakan tiga bangunan dasar untuk mendeskripsikan sistem atau perangkat lunak yang akan dikembangkan, yaitu :

a. Sesuatu (*things*)

Ada empat tings dalam *Unified Modelling Language* (UML):

1) *Structural things,* bagian yang relatif statis dapat berupa elemen yang bersifat fisik maupun konseptual.

2) *Behavorial things*, bagian dinamis biasanya merupakan kata kerja dari model UML yang mencerminkan perilaku sepanjang waktu
3) *Grouping things*, bagian yang pengorganisasian dalam UML. Dalam penggambaran model UML yang rumit diperlukan penggambaran paket yang menyederhanakan model. Paket-paket ini kemudian dapat didekomposisi lebih lanjut. Paket berguna bagi pengelompokan sesuatu, misalnya model-model serta subsistem.
4) *An notational things*, merupakan bagian yang memperjelas model UML. Dapat berisi komentar yang menjelaskan fungsi serta ciri-ciri tiap elemen dalam model UML.

b. Relasi (*relationship*)

Ada empat *relationship* (hubungan) dalam *Unified Modelling Language* (UML):

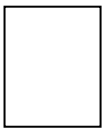
1) Ketergantungan (*dependency*) adalah hubungan dimana perubahan yang terjadi pada suatu elemen independent akan mempengeruhi elemen yang bergantung padanya.
2) Asosiasi adalah apa dan bagaimana yang menghubungkan antara objek satu dengan yang lainnya. Suatu bentuk asosiasi adalah agregasi yang menampilkan hubungan suatu objek dengan bagian-bagiannya.
3) Generalisasi adalah hubungan dimana objek anak berbagai perilaku dan struktur data dari objek yang ada diatasnya (objek induk). Arah dari objek induk ke objek anak dinamakan spesialisai sedangkan arah sebaliknya dinamakan generalisasi.

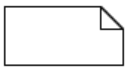
4) Realisasi adalah operasi yang benar-benar dilakukan oleh suatu objek.

Diagram *Use Case* menyajikan intraksi antara *Use Case* dan aktor, dimana aktor dapat berupa orang, peralatan, atau sistem lain yang berintraksi dengan sistem yang sedang di bangun.

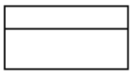
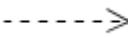
Adapun simbol-simbol *Use Case* diagram antara lain :

Tabel 2.1. Simbol *Use Case* Diagram

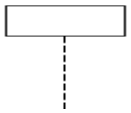
NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *Use Case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *Use Case* target memperluas perilaku dari *Use Case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor

NO	GAMBAR	NAMA	KETERANGAN
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		*Note*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

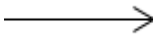
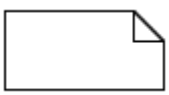
Tabel 2.2. Simbol *Class* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu actor
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya

Tabel 2.3. Simbol *Sequence* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi

Tabel 2.4. Simbol *StateChart Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*State*	Nilai atribut dan nilai link pada suatu waktu tertentu, yang dimiliki oleh suatu objek.
2		*Initial Pseudo State*	Bagaimana objek dibentuk atau diawali
3		*Final State*	Bagaimana objek dibentuk dan dihancurkan
4		*Transition*	Sebuah kejadian yang memicu sebuah state objek dengan cara memperbaharui satu atau lebih nilai atributnya
5		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
6		*Node*	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

Tabel 2.5. Simbol *Actifity* Diagram

NO	GAMBAR	NAMA	KETERANGAN
1		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Initial Node*	Bagaimana objek dibentuk atau diawali.
4		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

Q. *OpenCV*

OpenCV Open Source Computer Vision adalah *open source library* yang berisi lebih dari 500 algoritma teroptimasi untuk analisa citra atau video.Sejak diperkenalkan pada tahun 1999, *OpenCV* sebagian besar telah diadopsi sebagai alat pengembangan utama oleh komunitas peneliti dan pengembang dalam bidang komputer visi. *OpenCV* pada awalnya dikembangkan di Intel oleh tim yang dipimpin oleh Gary Bradski sebagai inisiatif untuk memajukan penelitian dalam visi dan mempromosikan pengembangan aplikasi yang kaya dan berbasis visi-CPU-intensif. Setelah *OpenCV* versi Beta diluncurkan, versi 1.0 diluncurkan pada tahun 2006. Rilis besar kedua terjadi pada tahun 2009 dengan diluncurkannya *OpenCV* 2 yang menawarkan banyak perubahan penting.*OpenCV* didesain untuk digunakan bersama dengan *Intel Image Processing Library IPL* dan memperluas

fungsionalitas terhadap citra dan pola analisis. Oleh karena itu, *OpenCV* berbagi format iplImage citra yang sama dengan IPL. Pengaplikasian *OpenCV* dapat digunakan untuk interaksi antara manusia dan *computer*, misalnya wajah dari manusia dideteksi oleh kamera *webcam*, lalu di proses oleh *computer*, untuk melakukan aksi tertentu seperti mengikuti mengenal wajah orang tersebut. Kesemuanya itu membutuhkan *OpenCV* sebagai program utama antara *webcam* dan perangkatnya yaitu *computer* maupun *smartphone*. *Library* ini terdiri dari fungsi-fungsi *computer vision* dan *API Aplication Programming Interface* untuk *image processing high* level maupun low level dan sebagai optimisasi aplikasi *realtime*. *OpenCV* sangat disarankan untuk programmer yang akan berkutat pada bidang *computer vision*, karena *library* ini mampu menciptakan aplikasi yang handal, kuat dibidang digital *vision*, dan mempunyai kemampuan yang mirip dengan cara pengolahan pada manusia.

R. *Single face vs Multiple face*

Single face merupakan citra wajah yang mana proses pengenalan wajah dilakukan secara otomatis terhadap objek wajah dengan mendeteksi objek hanya satu wajah (Yang, 2002). Penerapan *single face* saat ini banyak digunakan dalam absensi pengenalan wajah. Adapun keuntungan dari penerapan *single face* dalam sistem absensi pengenalan wajah yaitu sistem hanya fokus terhadap satu objek yang terdeteksi dan hanya mengolah satu citra image yang merupakan titik acuan pengenalan. Sebaliknya kekurangan dari sistem *single face* adalah kurangnya

efisiensi waktu terhadap banyak objek, sistem hanya bisa mendeteksi satu objek wajah dalam jangkauan kamera meskipun ada banyak objek .

Sedangkan *multiple face* merupakan citra wajah dimana sistem dapat mengenali banyak wajah secara otomatis. Dengan diterapkannya sistem ini banyak keuntungan yang dapat membuat sistem absensi pengenalan wajah menjadi lebih efisien, karena dapat mendeteksi dan mengenali objek wajah dalam satu kali input wajah (Rojas, 2015). Penerapan *multiple face* sangat berguna untuk absensi yang memiliki member yang cukup banyak, maka dari itu sistem *multiple face* ini dapat mendeteksi banyak objek muka yang berada dalam jangkauan kamera.

S. *Face Detection* menggunakan *Haar Cascade Classifier*

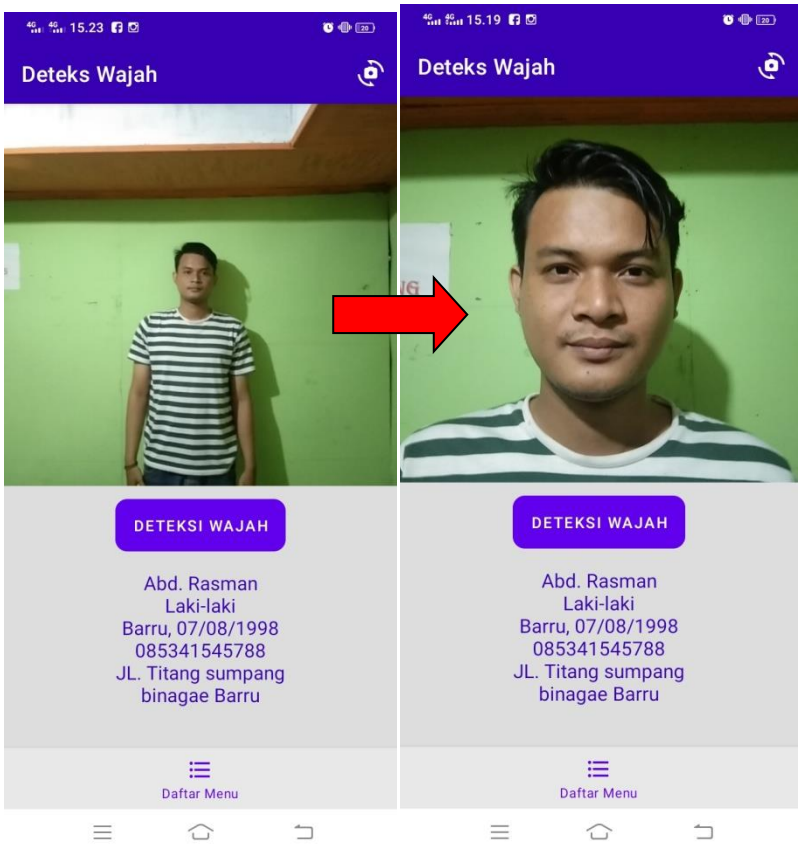
Face Detection adalah teknologi komputer yang digunakan dalam berbagai aplikasi yang mengidentifikasi wajah manusia dalam gambar digital. Deteksi wajah juga merupakan salah satu tahap proses yang sangat penting di dalam sistem pengenalan wajah yang digunakan untuk sistem biometrik.

Deteksi wajah juga dapat digunakan untuk pencarian dan pengindeksan citra yang di dalamnya terdapat wajah manusia dalam berbagai ukuran, posisi, dan latar belakang. Bagaimana sistem deteksi wajah ini memproses gambar dari obyek bergerak. Pemrosesan gambar ini bertujuan untuk mencari wajah dari gambar obyek bergerak yang telah di-*capture*, kemudian gambar tersebut diolah dengan memisahkan gambar dengan latar belakangnya, sehingga hanya bagian yang dianggap kulit yang ditampilkan sedangkan bagian yang bukan kulit akan

dihitamkan. Pemisahan gambar dengan latar belakang ini bertujuan untuk memudahkan proses pencarian wajah.

Deteksi objek wajah menggunakan *Haar Cascade classifier* yang merupakan sebuah fungsi pendeteksian objek wajah yang dimiliki oleh aplikasi *OpenCV*. *Face detector* akan malakukan proses pengujian tiap lokasi *image* dan mengklasifikasi sebagai wajah dan bukan wajah. Klasifikasi wajah ini memakai suatu ketetapan skala, misalnya 100×100 *pixel*. Jika wajah pada objek *image* lebih besar atau lebih kecil dari *pixel* tersebut, *classifier* terus menerus jalan beberapa kali, untuk mencari wajah pada gambar tersebut.

Classifier memakai data yang disimpan pada file XML untuk memutuskan bagaimana mengklasifikasi setiap lokasi *image*. *OpenCV* memakai 4 data XML untuk deteksi wajah depan dan wajah *profile*. Termasuk juga 3 file XML untuk bukan wajah, satu untuk mendeteksi badan secara penuh, satu untuk badan bagian atas, dan satu untuk badan bagian bawah. Kita harus memberitahukan (mendeklarasikan) letak dari *classifier* yang dipakai. Salah satunya bernama "*haarcascade_frontalface_default.xml*". Dari fungsi tersebut menghasilkan sebuah nilai *eigen* yang menjadi acuan pengenalan wajah dan menghasilkan pendeteksian objek wajah seperti pada gambar berikut.



Gambar 2.3. *Face Detection*

T. Multi Face Recognition

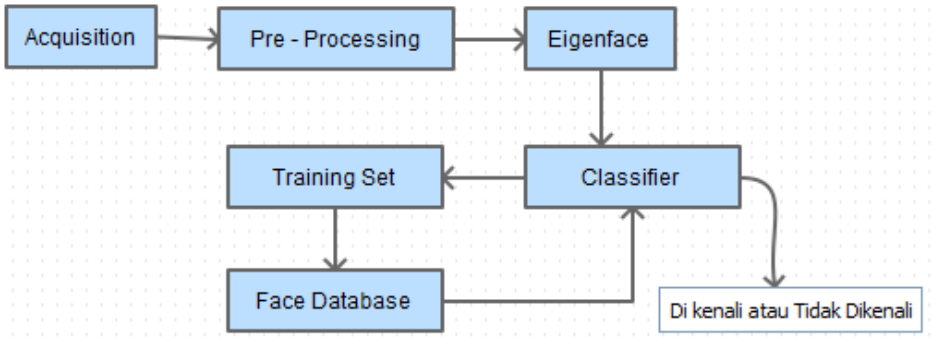
Face recognition adalah salah satu tehnik *biometric* yang memungkinkan komputer atau mesin *authentik* untuk mengenal wajah manusia. Salah satu aplikasi dari pengenalan wajah adalah pengenalan banyak wajah, yaitu pengenalan wajah dari suatu citra yang terdiri dari banyak wajah. Untuk membedakan beberapa wajah manusia dalam suatu citra sangat sulit bagi sistem pengenalan wajah. Selain itu juga terdapat beberapa masalah Pada sistem pengenalan wajah otomatis, yaitu ekspresi wajah, iluminasi atau pencahayaan dan

jarak. Sistem pengenalan banyak wajah merupakan salah satu solusi dari permasalahan ini.

Face Recognition atau pengenalan wajah adalah suatu sistem atau metode yang berorientasi pada wajah. Pengenalan ini dapat dibagi menjadi dua bagian yaitu : “dikenali” dan “tidak dikenali”, setelah dilakukan dengan pola yang sebelumnya disimpan di dalam database. Sistem ini juga harus bisa mengenali objek bukan wajah .

Secara umum sistem pengenalan citra wajah dibagi menjadi dua jenis, yaitu sistem *feature-based* dan sistem *image-based.* Pada sistem pertama digunakan ciri yang dieksekusi dari komponen citra wajah seperti mata, hidung, mulut, dan lain-lain yang kemudian dimodelkan secara geometris hubungan antara ciri-ciri tersebut. Sedangkan pada sistem kedua menggunakan informasi mentah dari pixel citra yang kemudian direpresentasikan dalam metode tertentu, misalnya pada *Principal Component Analysis* (PCA) atau Transfomasi *Wavelet* yang digunakan untuk klasifikasi indentitas citra.

Multi face recognition mengambil karakteristik alami yang ada pada tiap-tiap wajah untuk dikenali. Ada tiga tahap untuk melakukan *face recognition,* yaitu deteksi wajah, ekstraksi ciri dan klasifikasi. Deteksi wajah adalah suatu langkah dalam *face recognition* untuk menemukan posisi wajah dari sebuah citra yang akan di ekstraksi selanjutnya. Ekstraksi ciri adalah langkah untuk menentukan karakteristik alami dari suatu wajah yang selanjutnya akan diklasifikasi atau dikenali.



Gambar 2.4. Diagram Blok Pengenalan wajah

Digram blok diatas merupakan tahap sistem dalam melakukan pengenalan wajah. Perolehan (*acquisition*) untuk mendapatkan input gambar (*face image*) selanjutnya *Pre*_processing pengolahan citra *image* sebagai tahap awal pemrosesan objek wajah menghasilkan *normalized face image*, dan selanjutnya akan di lakukan proses *feature extractor* untuk extraksi ciri pada setiap *image* yang telah masuk untuk mendapatkan ciri yang membedakan setiap objek wajah. *Training sets* melakukan perhitungan terhadap objek wajah terhadap database yang sudah ada(*face database*). Selanjutnya dilakukan *classifier* dimana proses ini melakukan tahap yang paling penting yaitu menjalankan apakah objek wajah tersebut dianggap dikenali atau tidak.

U. Metode Eigenface

Eigenface berasal dari prefiks bahasa Jerrnan “eigen”, yang berarti “sendiri/individual”. Metode *Eigenace* dianggap sebagai teknologi pengenalan wajah otomatis pertama yang pernah diciptakan. Teori ini dikembangkan oleh Turk dan Petland.

Teori ini dikembangkan dengan membagi sebuah citra wajah menjadi data set fitur karakteristik yang disebut *eigenface*. *Eigenface* adalah salah satu algoritma pengenalan pola wajah yg berdasarkan pada *Principle Component Analysis* (PCA) yg dikembangkan di MIT. Eigenface merupakan kumpulan dari eigenvector yang dipakai untuk masalah *computer vision* pada pengenalan wajah manusia.

Fitur karakteristik merupakan komponen utama (*principal component*) dari *training* set awal dari citra wajah. Penelitian yang dilakukan oleh Carey dan Diamond menunjukkan bahwa fitur wajah sebuah individu dan hubungan langsung antar fitur tersebut tidak dapat menyamai kemampuan manusia dalam memperhatikan dan mengenal wajah. *Eigenface* adalah sekumpulan *standardize face ingredient* yang diambil dari analisis statistik dari banyak gambar wajah (Layman dalam Al Fatta, Hanif, 2009).

Untuk menghasilkan *eigenface*, sekumpulan citra digital dari wajah manusia diambil pada kondisi pencahayaan yang sama kemudian dinormalisasikan dan diproses pada resolusi yang sama (misal m x n), kemudian citra tadi diperlakukan sebagai vektor dimensi m x n dimana komponennya diambil dari nilai piksel citra.

Metode *eigenface* menggunakan *eigenvectors* dan *eigenvalues* untuk mempresentasikan *image* wajah. *Eigenvector* merupakan sekumpulan dari fitur yang mencirikan variasi antara *image* wajah. Setiap lokasi *image* wajah memberikan kontribusi yang lebih atau kurang untuk setiap *eigenvector*, sehingga tampilan dari *eigenvector* seperti wajah yang samar. Dalam mengenali objek wajah,

hasil deteksi wajah kemudian dibandingkan dengan wajah yang tersimpan di *database/template*.

V. Konsep Perhitungan Deteksi Wajah

1. Menghitung jarak dari gambar tersebut dibandingkan dengan gambar-gambar yang ada di dalam *database*. (menggunakan perhitungan *Euclidean Distance*)
2. Memilih sebuah gambar dari database yang mendekati wajah yang ada di dalam gambar tersebut.
3. Jika jarak yang telah diukur tersebut hasilnya mendekati nilai dari threshold maka gambar tersebut dikenali oleh sistem, tetapi bila nilai yang dihasilkan memiliki perbedaan yang jauh maka gambar tersebut termasuk dalam gambar yang tidak dikenali oleh sistem karena sistem hanya mengenali gambar yang ada di dalam database.

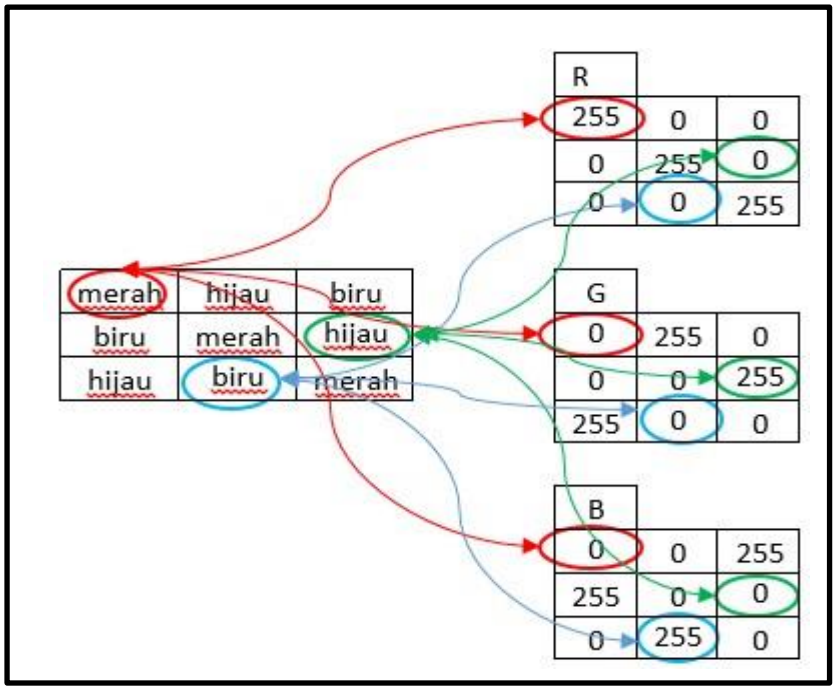


Gambar 2.5 Objek wajah dikenali

RGB adalah suatu komponen penyusun gambar yang mana terdiri atas 3 warna yaitu merah (*Red*) R, hijau (*Green*) G, dan biru (*Blue*) B, yang menjadi patokan secara *universal*. Dari ketiga warna tersebut dikombinasikan menjadikan warna – warna pada setiap pixel sehingga menjadi gambar. Setiap piksel pada warna membuthkan citra data sebesar 3 byte. Setiap warna dasar memiliki intensitas unuk menentukan seberapa besar tingkatan warna tersebut dengan nilai nol (0) dan nilai maksimum 255 (8 bit). Dalam RGB digunkan kode – kode untuk menentukan warna. Seperti gambar berikut :



Gambar 2.6 Representasi Warna Pada RGB



Gambar 2.7 Komponen Gambar (RGB)

Grayscale adalah warna – warna dalam *pixel* yang berada dalam rentang gradasi warna hitam dan putih. Citra yang ditampilkan dari *grayscale* adalah citra yang terdiri atas warna abu – abu, berorientasi pada warna hitam pada gradasi warna dengan intensitas lemah dan warna putih pada gradasi dengan intensitas warna kuat.

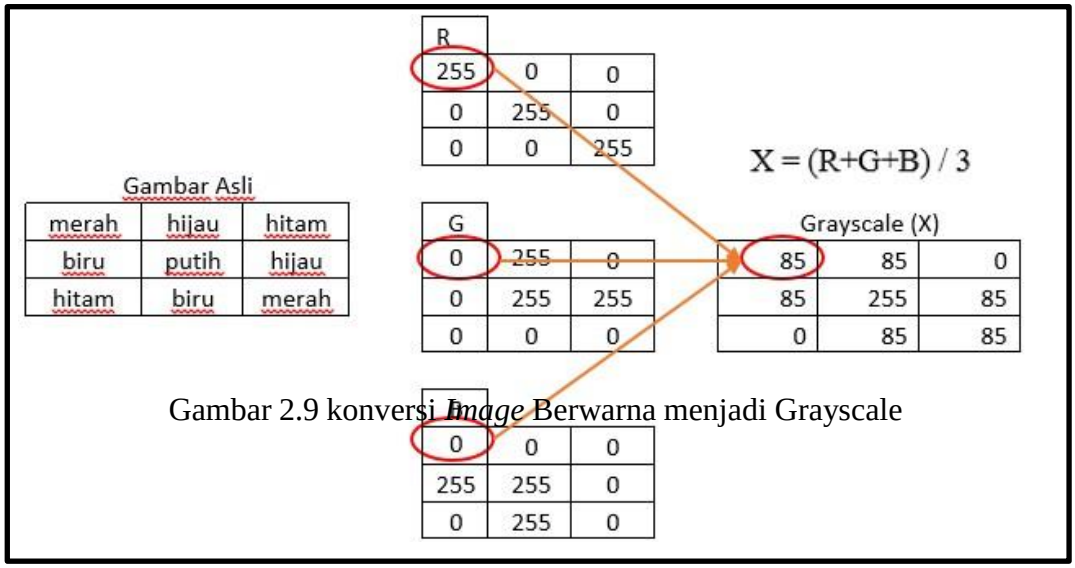
Citra *grayscale* memiliki format 8 bit pada setiap piksel, yang memiliki intensitas tingkatann warna sebanyak 256 tingkatan. Untuk mengubah citra berwarna dalam suatu *image* R,G, dan B menjadi *image* grayscale, maka dapat dikonversi dengan mengambil nilai rata – rata dari nilai R, G, dan B sehingga dapat dituliskan dalam rumus sebagai berikut :

$$X = (R+G+B) / 3$$



Gambar 2.8 Visualisasi 256 tingkat keabu – abuan

Dengen rumus tersebut maka dapat di implementasikan bahawa seperti contoh pada gambar 2.12. proses gambar berwarna diubah menjadi *grayscale* sebagai berikut :



Gambar 2.9 konversi *Image* Berwarna menjadi Grayscale

Dari proses tersebut selanjutnya akan di proses menggunakan fungsi *library "haarcascade_Classifier.xml"* dalam hal ini klasifikasi pada *frontal face* yaitu "*haarcascade_frontalface_default.xml."* sebagai pendeteksi objek wajah tampak

depan lurus. Pada penggunaan *library* ini menghasilkan sebuah nilai yang menjadi acuan terhadap pengenalan wajah.

a. Euclidean Distance

Euclidean Distance merupakan perhitungan yang paling sering digunaakan untuk menhitung jarak antara 2 vektor. Cara perhitungan *Euclidean Distance* adalah dengan menghitung akar dari kuadrat perbedaan 2 vektor (jarak antara 2 titik).

Rumus *Euclidean Distance* sebagai berikut :

$$d_{ab} = \sqrt{\sum_{k=1}^{n} (X_{ak} - X_{bk})^2}$$

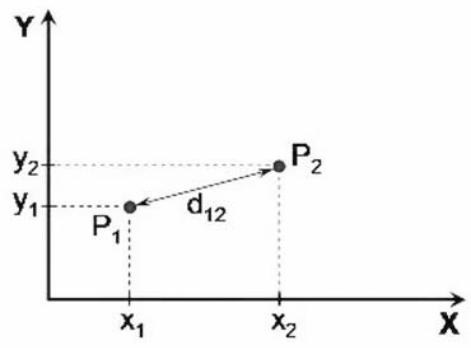
Keterangan :

d_{ab} : *Euclidean Distance*

X : nilai vektor

ak, bk : vektor yang dioperasikan

contoh : *image* diketahui mempunyai vektor P_1 = (213), Y = (124,4).



Gambar 2.10 Jarak Euclidean

Penyelesaian : P_1 = (213), Y = (124,4).

Euclidean Distance d_{12} = √(213 – 124)^2 + (0 – 4)^2

= √(89)^2 + 0

= √7921

= 89

Sehingga jarak *euclidean* dari kedua titik P1 dan P2 adalah 89.

BAB III

METODE PENELITIAN

A. Jenis Penelitian

Jenis penelitian yang digunakan adalah penelitian deskriptif dimana memberikan gambaran mengenai apa yang sesungguhnya terjadi. Dalam pembuatan Skripsi ini digunakan metode deskripsif yang menggambarkan fakta-fakta dan informasi secara sistematis, faktual dan akurat.

Penelitian ini dilakukan melalui internet yang dapat memberikan sumber data dan pengetahuan mengenai sistem yang diteliti, kemudian mencocokkan dengan kemungkinan yang terjadi dalam usaha penyelesaian masalah.

B. Waktu dan Tempat Penelitian

Adapun waktu dan tempat penelitian yang dilaksanakan pada bulan April tahun 2021, dimana proses pengambilan data wajah yang akan dimasukkan dalam aplikasi nantinya yaitu Masyarakat Umum dan mahasiswa Teknik Informatika Fakultas Teknik Universitas Muhammadiyah Parepare

C. Metode Pengumpulan Data

Untuk memperoleh data-data yang dibutuhkan dalam rangka melakukan penelitian, maka penulis mengumpulkan data melalui beberapa cara yaitu :

1. Analisis Data

   Menganalisa data-data yang sebelumnya telah dikumpulkan.

2. Perancangan Program

   Sebagai pedoman dalam penulisan program atau kode-kode agar berjalan sesuai rencana.

3. Uji Coba Program

   Pengujian program dilakukan untuk memastikan bahwa program yang dibuat dapat berjalan dengan baik.

4. Evaluasi

   Sistem yang telah selesai dibangun perlu adanya evaluasi untuk menemukan kelemahan yang terdapat pada program yang telah dibangun tadi, yang nantinya bisa digunakan sebagai acuan untuk memperbaiki program sehingga lebih sempurna.

D. Jenis Data

Data yang digunakan dalam penelitian ini berupa data-data yang telah dikumpulkan melalui Penelitian Pustaka (*Library Research*). Adapun jenis data primer dan data sekunder yang relevan dengan masalah yang akan dibahas.

a. Data primer

   Data Primer adalah data yang berasal atau data yang diperoleh langsung dari sumber data dan pengetahuan.

b. Data Sekunder

   Data sekunder adalah data yang diperoleh tidak secara langsung dari objek penelitian. Peneliti mendapatkan data yang sudah jadi dari internet, website dan jurnal.

E. Tahapan Peneletian

Tahapan penelitian yang dimaksud dalam penelitian ini ada beberapa tahapan yaitu persiapan penelitian, pengumpulan data, analisis perancangan, pengujian dan implementasi. Adapun Uraian dari tahapan tersebut adalah sebagai berikut :

1. Persiapan Penelitian

   Pada tahapan ini peneliti melakukan persiapan penelitian. Persiapan penelitian yang dimaksud adalah menyiapkan buku-buku, artikel-artikel tentang topik penelitian serta software yang digunakan selama penelitian.

2. Pengumpulan Data

   Pada tahap ini peneliti melakukan observasi dengan peninjauan, pencatatan dan pengamatan langsung di tempat penelitian.

3. Analisis

   Pada tahap analisis, peneliti melakukan analisa terhadap sistem yang di terapkan sekarang berdasarkan kemudian merumuskan masalah yang menjadi pokok penelitian sehingga dapat dibuat alternatif pemecahan masalah.

4. Perancangan

   Peneliti kemudian merancang aplikasi yang ingin dibuat berdasarkan alternatif pemecahan masalah.

5. Pengujian

   Setelah melakukan perancangan, peneliti kemudian menguji hasil perancangan yang telah dibuat. Jika hasil perancangan terdapat kekurangan atau kelemahan maka kembali ke tahap analisis.

6. Implementasi

Setelah pada perancangan tidak terdapat kekurangan maka aplikasi siap untuk di gunakan oleh user.

F. Metode Pengujian Aplikasi

Beberapa *test-case* harus dilaksanakan dengan beberapa perbedaan strategi, query, atau jalur navigasi yang mewakili penggunaan sistem yang *tipical*, kritis atau abnormal. Isu kunci pada pengembangan sistem adalah pemilihan sekelompok *test-case* yang cocok, sekecil dan secepat mungkin, untuk meyakinkan perilaku sistem secara detail. Pengujian harus mencakup *unit testing*, yang mengecek validasi dari prosedur dan fungsi-fungsi secara independen dari komponen sistem yang lain. Kemudian modul testing harus menyusul dilakukan untuk mengetahui apakah penggabungan beberapa unit dalam satu modul sudah berjalan dengan baik, termasuk eksekusi dari beberapa modul yang saling berelasi, apakah sudah berjalan sesuai karakteristik sistem yang diinginkan.

Jika struktur kendali antar modul sudah terbukti bagus, maka pengujian yang tak kalah pentingnya adalah pengujian unit. Pengujian unit digunakan untuk menguji setiap modul untuk menjamin setiap modul menjalankan fungsinya dengan baik. Ada 2 metode untuk melakukan unit testing, yaitu :

1. White Box Testing

Uji coba *white box testing* merupakan metode perancangan *testcase* yang mengunakan struktural untuk mendapatkan *testcase*, test ini digunakan untuk meramal cara kerja perangkat lunak secara rinci kepada *logic path* (jalur logika), perangkat lunak di tes dengan kondisi dan perulangan secara fisik.

Contoh pengujian white box testing ini merupakan peringatan ketika user menginputkan password user yang salah, untuk kesalahan semacam ini akan memberikan suatu informasi kepada user mengenai kesalahan yang di lakukan.

2. Black Box

Berdasarkan hasil uji coba yang dilakukan, seluruh navigasi dan tombol fasilitas program lainnya serta proses yang di jalankan tidak terjadi kesalahan, tetapi aplikasi mempunyai aturan-aturan yang sudah di tetapkan dan harus di ikuti karena apabila di hiraukan maka sistem akan menolak perintah yang tidak sesuai seperti kesalahan ketika user belum menginput data yang harusnya di input sesuai ketentuan sistem yang di jalankan dan sistem memberikan informasi kepada user karena data yang ingin diproses belum lengkap atau tidak memenuhi ketentuan untuk proses selanjutnya.

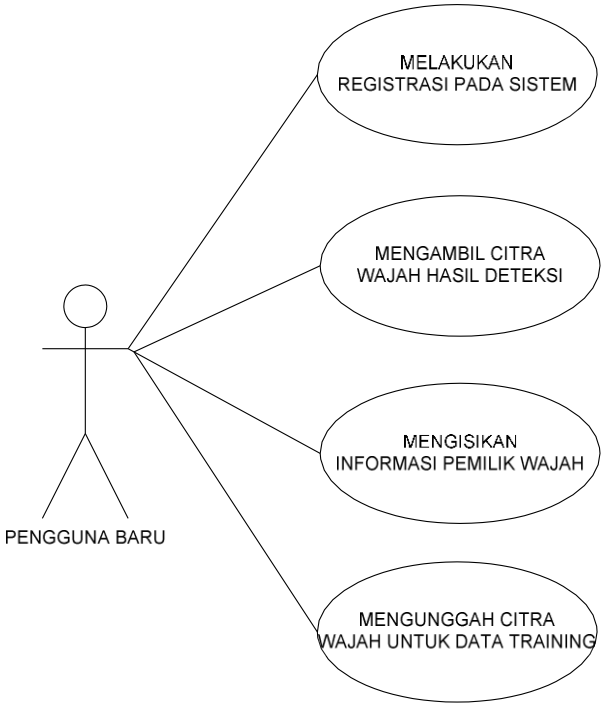
G. Metode Pengujian

Metode pengujian yang dilakukan pada penelitian ini menggunakan blackbox yang dipakai untuk mengetahui kinerja aplikasi pada saat melakukan proses *Tour Guide* dan mengetahui berfungsinya semua aplikasi yang telah dibuat, *blackbox* dalam metode pengujian perangkat lunak yang tes fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja.

BAB IV

HASIL DAN PEMBAHASAN

A. Rancangan Sistem Yang Diusulkan

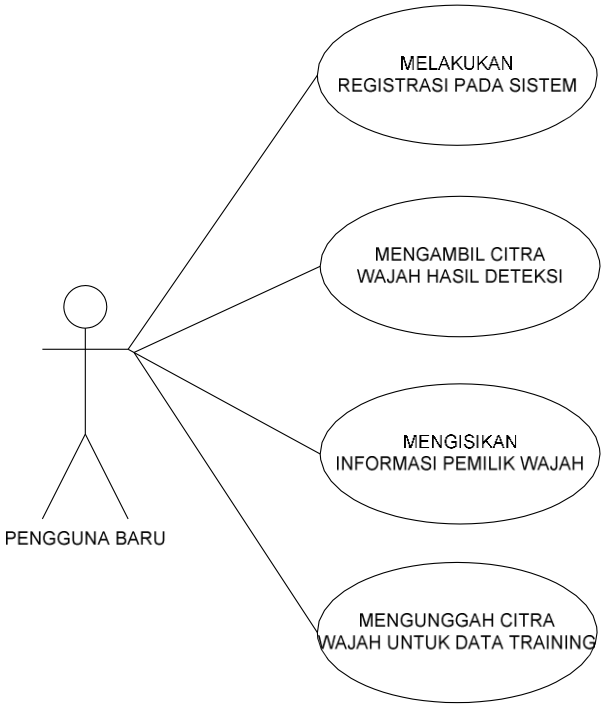


Gambar 4.11. Use Case Diagram

Gambar 4.11 merupakan diagram *use case* dari sistem pengenalan wajah, yaitu diagram *use case* yang menunjukkan interaksi antara pengguna baru dengan sistem. Pertama, pengguna harus memilih fitur registrasi pada sistem. Selanjutnya akan masuk pada tampilan fitur *training*, pengguna diminta untuk mengambil citra wajah depan dirinya dari berbagai ekspresi. Kemudian citra wajah yang sudah diambil harus diberikan label berupa in*form*asi data diri. Sebagai tahapan akhir, citra wajah yang telah memiliki label in*form*asi.

Use Case Diagram menjelaskan fungsi yang dimiliki oleh sistem dari segi fungsionalitas yang dilakukan oleh pengguna system. *Use Case Diagram* dari sistem ini terbagi menjadi dua jenis yaitu *Use Case Diagram* untuk registrasi awal pengguna sebagai tahap awal untuk proses *training* data wajah pengguna aplikasi dan *Use Case Diagram* untuk pengguna terdaftar.

1. *Use Case Diagram*



Gambar 4.12 *Use Case Diagram* Sistem Yang Diusulkan

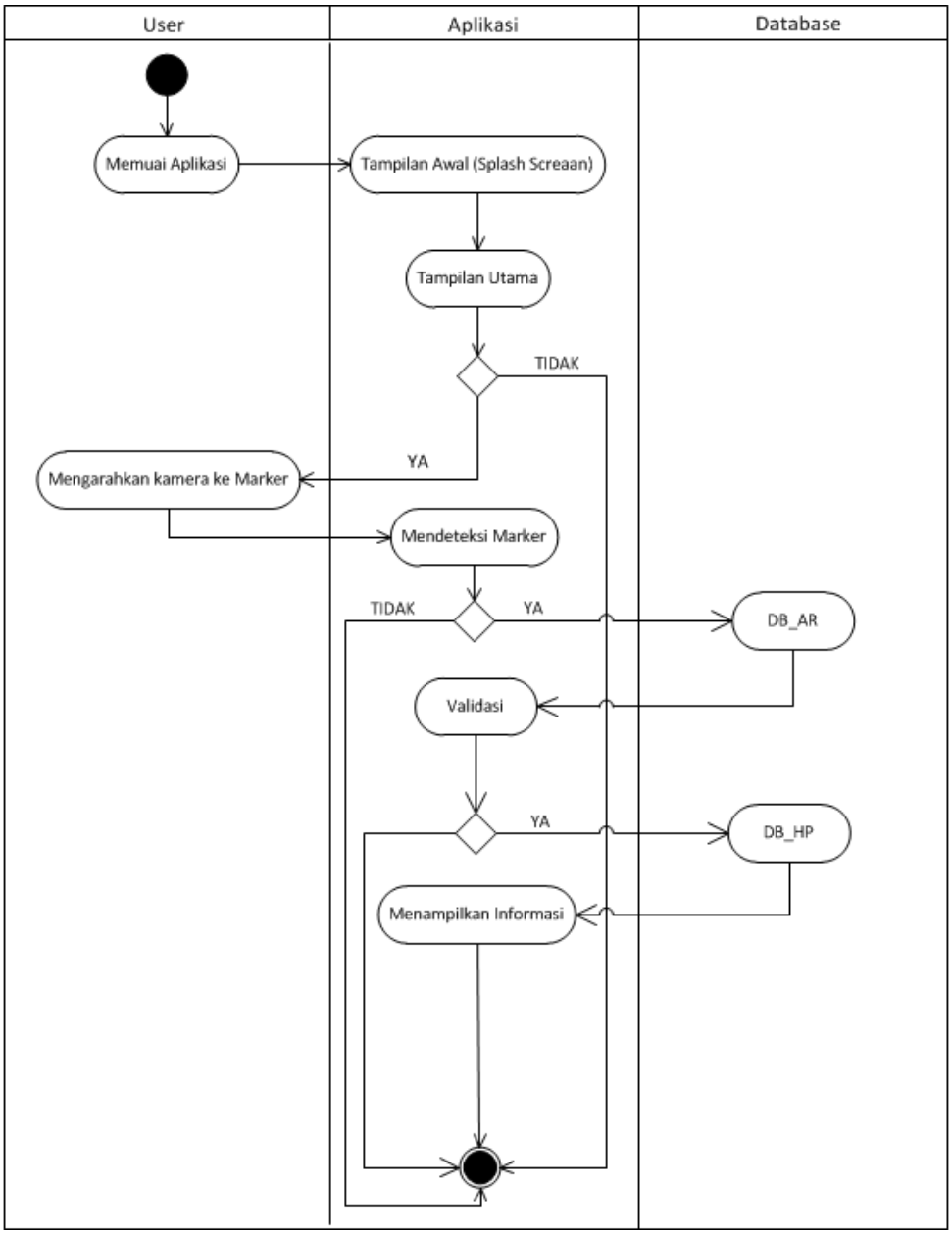
a. *Actor* Pengguna

Tabel 4.6 Tabel *Actor* Pengguna

Nama *Use Case*	Deskripsi *Use Case*
Registrasi Sistem	*Use Case* ini menjelaskan bahwa *actor* pengguna memulai aplikasi *augmented reality* yang digunakan untuk menampilkan registrasi sistem.
Tampilan Awal	*Use Case* ini menjelaskan bahwa setelah pengguna memulai aplikasi, akan tampil tampilan awal aplikasi.
Tampilan Utama	*Use Case* ini menjelaskan tentang tampilan utama yang digunakan untuk pendeteksian *wajah*.
Deteksi *Wajah*	*Use Case* ini menjelaskan bahwa setelah *AR* kamera aktif maka pengguna akan mengarahkan kamera ke *wajah* untuk menginisialisasi dan identifikasi *wajah* dapat terdeteksi.

2. *Activity Diagram*

a. *Activity diagram* aplikasi



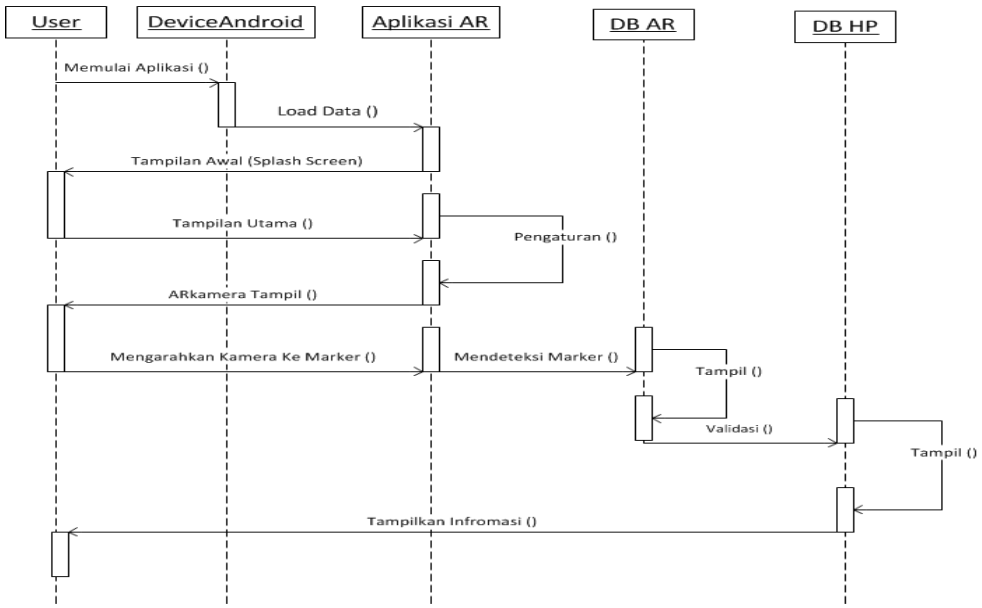
Gambar 4.13 *Activity Diagram* Sistem yang Diusulkan

Aktifitas diagram diatas menjelaskan bahwa aplikasi yang digunakan atau dioperasikan oleh *user* akan menampilkan tampilan awal (*splash screen*) aplikasi, dan selanjutnya secara otomatis akan pindah pada tampilan utama aplikasi, aplikasi yang

pertama digunakan setelah melalui tahap ini akan melakukan persamaan *key* yang ada pada database *vuforia*, setelah aplikasi melakukan persamaan maka aplikasi akan melanjutkan tampilan yang ada pada tampilan utama dan siap digunakan oleh pengguna untuk diarahkan ke *wajah*, pada tahap ini untuk *wajah* dapat terdeteksi, aplikasi akan melakukan persamaan gambar pada database *vuforia*, jika terdapat persamaan gambar maka selanjutnya sebelum aplikasi menampilkan in*form*asi, terlebih dahulu aplikasi akan melakukan persamaan *ID* yang telah di inisialisasi pada *wajah* untuk memanggil data yang ada pada database.

3. *Sequence Diagram*

a. *Sequence Diagram* Aplikasi

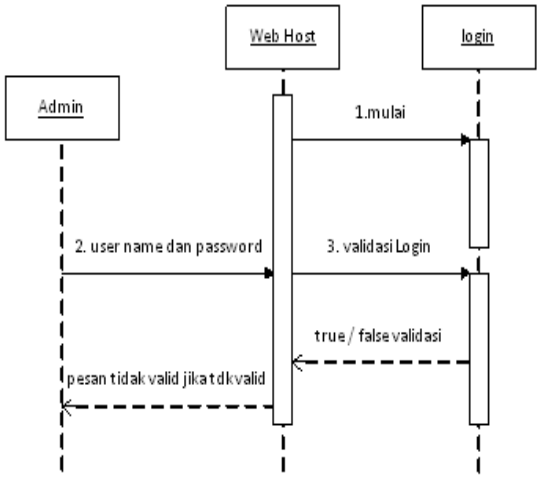


Gambar 4.14 *Sequence Diagram* Sistem yang diusulkan

Diagram *sequence* diatas menjelaskan bahwa ketika *user* memulai aplikasi maka *device* android yang digunakan akan meload data untuk selanjutnya memulai aplikasi dan menampilkan tampilan awal aplikasi (*splash screen*), selanjutnya *user* akan diarakan pada tampilan utama, pada tampilan utama terdapat pengaturan yang dapat digunakan oleh *user* dan selanjutnya akan

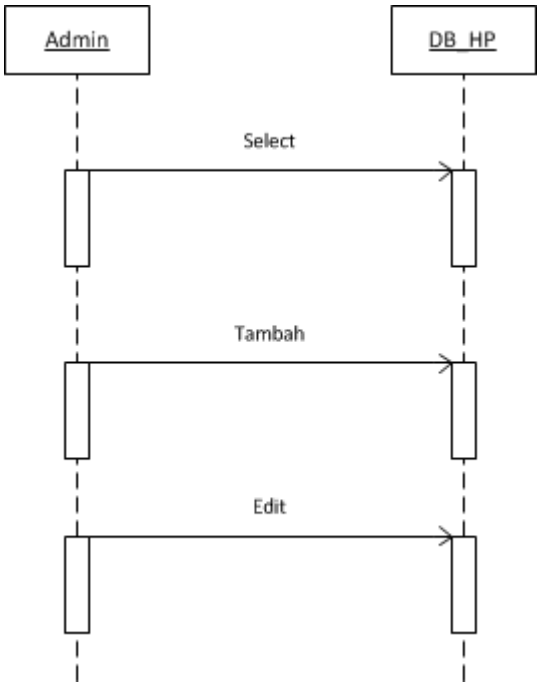
diarahkan pada *AR* kamera untuk melakukan proses pendeteksian *wajah* untuk selanjutnya melakukan pencocokan dengan database, jika sesuai dengan database aplikasi akan kembali melakukan persamaan ID pada *wajah* dengan ID yang ada pada database *mysql*, jika sesuai dengan database maka in*form*asi dari objek tersebut akan tampil.

b. Sequence Diagram Login



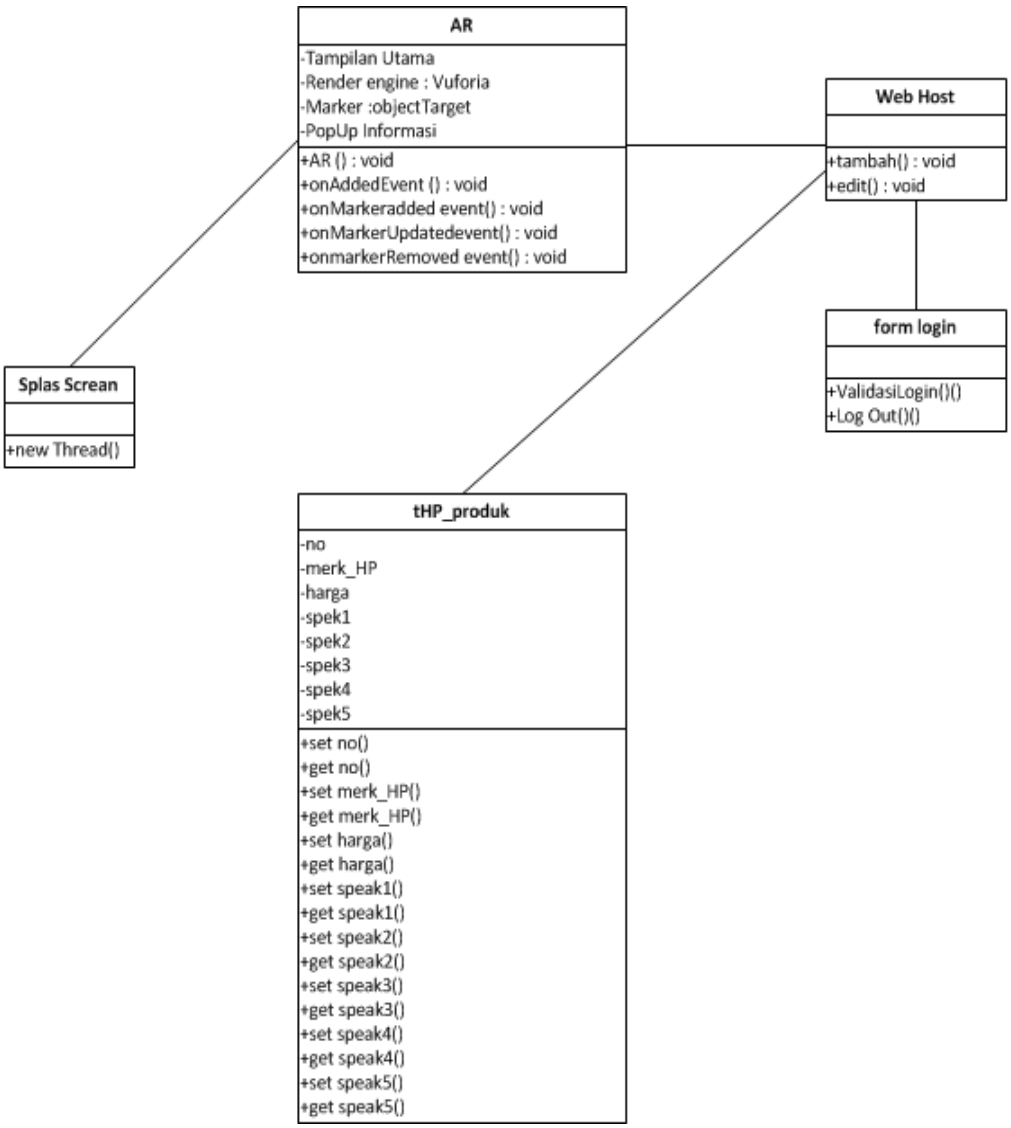
Gambar 4.15 *Sequence Diagram Login*

c. Sequence Diagram Vuforia



Gambar 4.16 *Sequence Diagram vuforia*

4. *Diagram Class*



Gambar 4.17 *Class Diagram*

Class diagram diatas menjelaskan keterkaitan antara setiap *form* dan fungsi-fungsi yang ada di dalamnya, ketika aplikasi dijalankan pertama kali akan muncul *splash screen* sebelum masuk ke tampilan utama, pada menu utama ini adalah inti dari aplikasi yang didalamnya terdapat proses pendeteksian *wajah* dilakukan, setelah proses pendeteksian *wajah*.

B. Detail Aplikasi

1. Prinsip Kerja Aplikasi

   Aplikasi *Augmented Reality* untuk deteksi wajah, dimana data yang diambil berada pada database yang telah daftar dan kemudian ditampilkan oleh aplikasi pada layar *smartphone.*

2. Fitur-fitur Aplikasi

   Adapun fitur-fitur yang terdapat pada aplikasi yaitu

   a. Lampu *flash*

      Lampu *flash* yang digunakan menggunakan lampu *flash* bawaan dari *smartphone* yang kemudian akan diaktifkan oleh pengguna ketika pencahayaan pada lokasi tersebut kurang memadai yang dapat mengganggu proses pendeteksian *wajah*.

   b. *Front* Kamera

      Front kamera yang digunakan dapat difungsikan ketika kamera belakang bermasalah atau pada waktu-waktu tertentu.

   c. Manual *focus*

      Manual *focus* digunakan ketika kamera sulit untuk mendeteksi *wajah* dikarenakan *focus* pada kamera *smartphone* kurang.

   d. Penginputan data

      Terdapat penginputan data, yaitu

      a) Penginputan tambah data

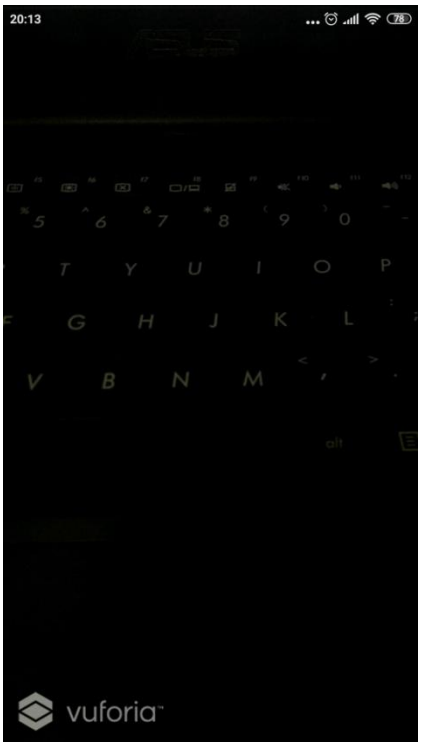
         Penginputan tambah data yaitu ketika wajah pengguna ingin ditambahkan maka data wajah tersebut akan tersimpan kedatabase..

C. Rancangan *Input* dan *Output*

1. Rancangan *Output*

Rancangan *output* dari aplikasi yang dibuat adalah sebagai berikut:

a. Tampilan Utama



Gambar 4.18 Tampilan Utama

Gambar diatas merupakan tampilan utama yang berfungsi untuk menampilkan in*form*asi setelah *wajah* terdeteksi, pada tampilan ini terdapat pengaturan kamera, yaitu aktif lampu *flash*, *front kamera*, dan *manual focus*.

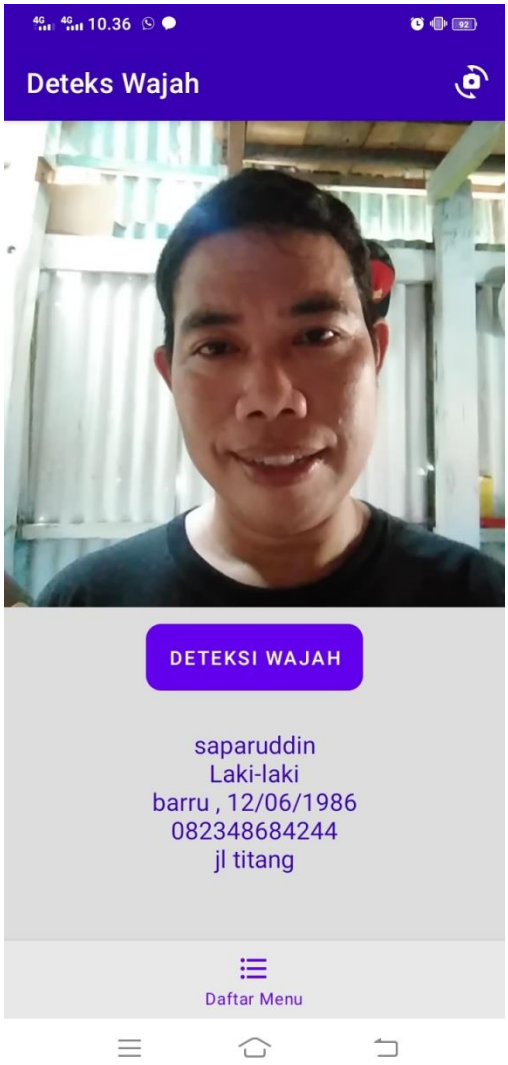
b. Tampilan deteksi wajah pertama



Gambar 4.19 Tampilan deteksi wajah pertama

Gambar diatas merupakan tampilan wajah pertama yang akan di deteksi oleh aplikasi

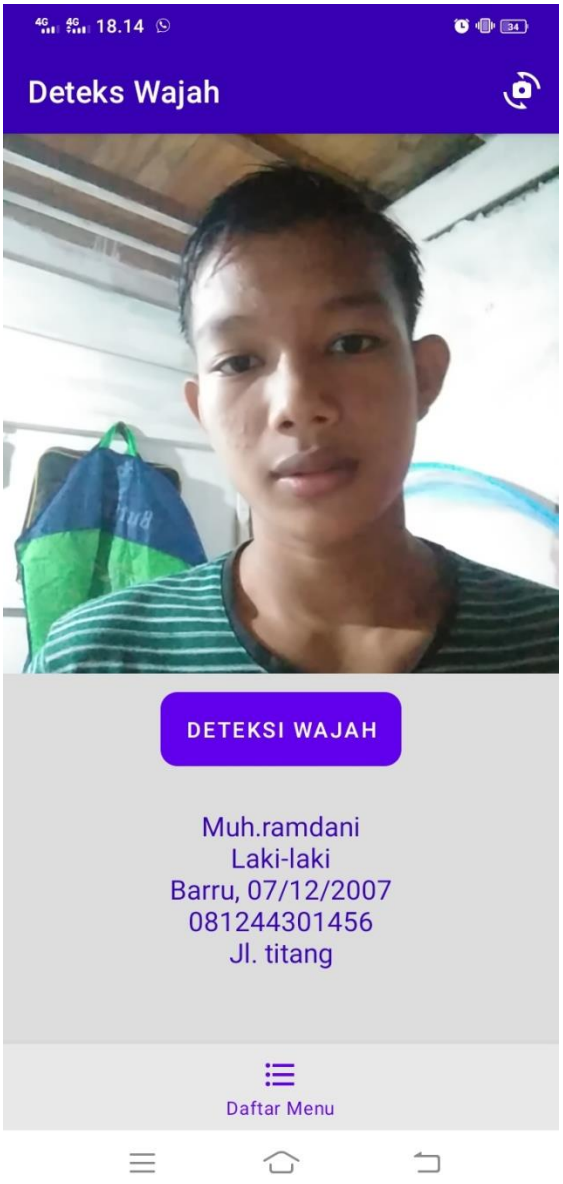
c. Tampilan deteksi wajah kedua



Gambar 4.20 Tampilan wajah kedua

Gambar diatas merupakan hasil dari wajah kedua yang telah terdeteksi.

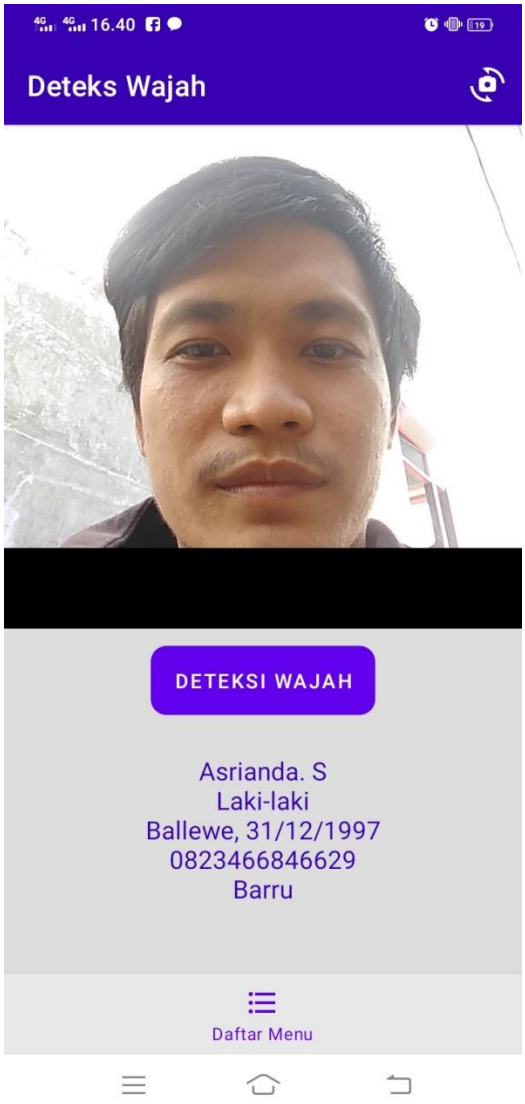
d. Tampilan deteksi wajah ketiga



Gambar 4.21 Tampilan wajah ketiga

Gambar diatas merupakan tampilan wajah ke tiga yang akan di deteksi oleh aplikasi

e. Tampilan deteksi wajah keempat



Gambar 4.22 Tampilan deteksi wajah keempat

Gambar diatas merupakan hasil dari deteksi wajah keempat yang telah terdeteksi oleh aplikasi

D. PENGUJIAN SISTEM

1. Implementasi

Pada tahap ini *design* sistem yang telah penulis rancang diterapkan ke dalam bahasa pemrograman yang kemudian dilakukan pengujian sistem, bahasa pemrograman yang penulis gunakan untuk membuat aplikasi ini adalah bahasa pemrograman *C#* (*Sharp*).

1. Kebutuhan *Hardware*

   Spesifikasi minimum *hardware* pengguna aplikasi ini adalah sebagai berikut :

   Tabel 4.7 Spesifikasi *Hardware Smartphone*

Jenis	Spesifikasi
Smartphone	Semua Merk
Processor	*Snapdragon 439 octa-core 1,95 GHz*
RAM	3 GB
Memori	32 GB
Camera	8 MP + 13MP
Display	*Touchscreen* 46.35 *inci*

2. Kebutuhan *Software*

   Spesifikasi minimum *software* pengguna aplikasi ini yaitu sebagai berikut :

   Tabel 4.8 Spesifikasi *Software*

Jenis	Spesifikasi
Sistem Operasi	*Android* Ver. 4.4.2 (*Kitkat*)
Unity Pro	*2020*

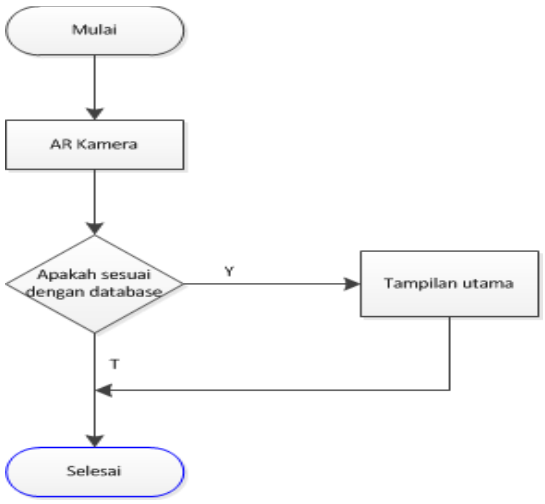
2. Pengujian Aplikasi

Pengujian aplikasi dilakukan dengan menggunakan dua metode pengujian yaitu Pengujian *Black Box (White Box Testing)* dan Pengujian *White Box (Black Box Testing).*

1. Pengujian *White Box* berfokus pada efektifitas aplikasi yang dirancang

   a. white box aplikasi

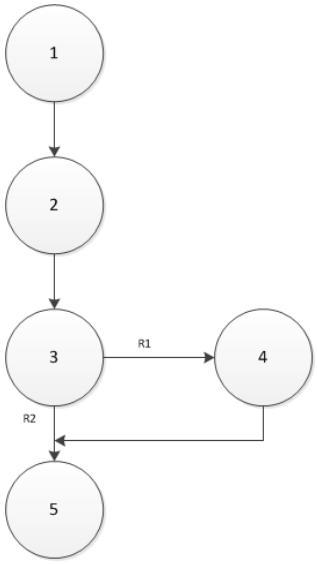
      1. Flowchart Aplikasi



Gambar 4.23 *Flowchart* Aplikasi

1) *Flow Graph* Aplikasi

Dari *flowchart* aplikasi yang digunakan untuk pengujian perangkat lunak, maka ditentukan *flowgraph* sebagai berikut



Gambar 4.24 *Flowgraph* Aplikasi

2) Proses Pehitungan aplikasi

Dari gambar *flowgraph* aplikasi diatas dapat dilakukan proses perhitungan sebagai berikut :

a) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

Dengan rumus : V (G) = E – N + 2

N(*node*) = 5

E(*edge*) = 5

P (*predikat node*) = 1

Penyelesaian : V(G) = E- N + 2

= 5 – 5 + 2

= 2

Predikat *Node* (N) = P + 1

= 1 + 1

= 2

b) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 2

c) *Independent Path* pada *flowgraph* diatas adalah :

Path 1 = 1 - 2 - 3 - 4 - 5

Path 2 = 1 – 2 – 3 – 5

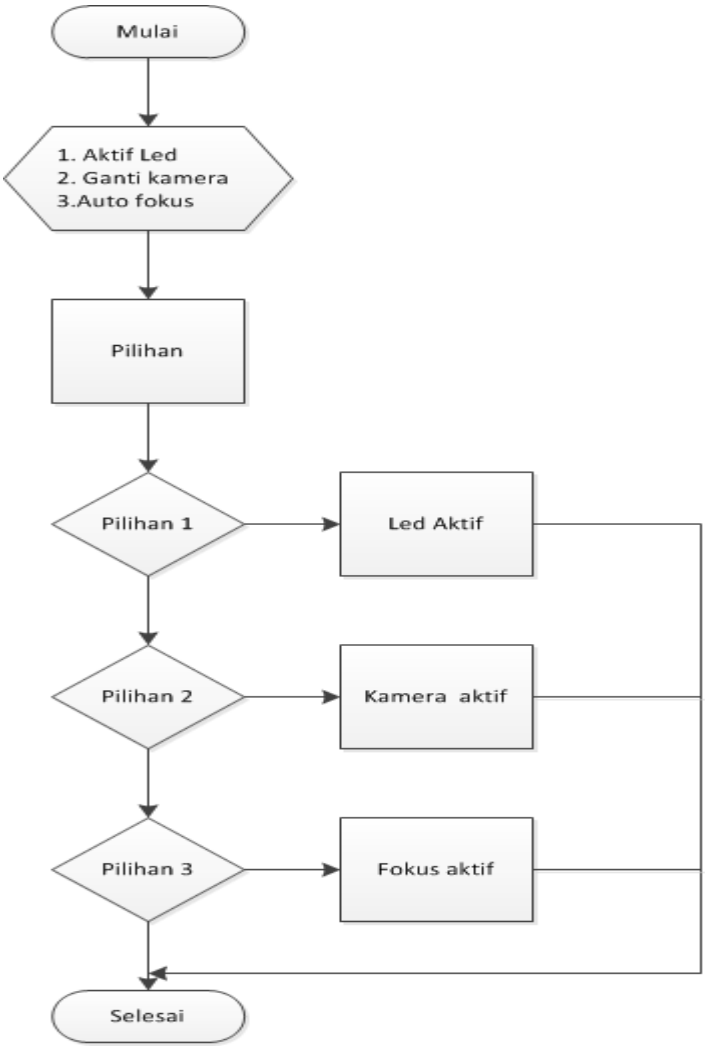
d) Grafik matriks aplikasi

Tabel 5.9 Grafik Matriks Aplikasi

	1	2	3	4	5	E-1
1		1				1-1=0
2			1			1-1=0
3				1	1	2-1=1
4					1	1-1=0
5						0
SUM(E+1)						1+1=2

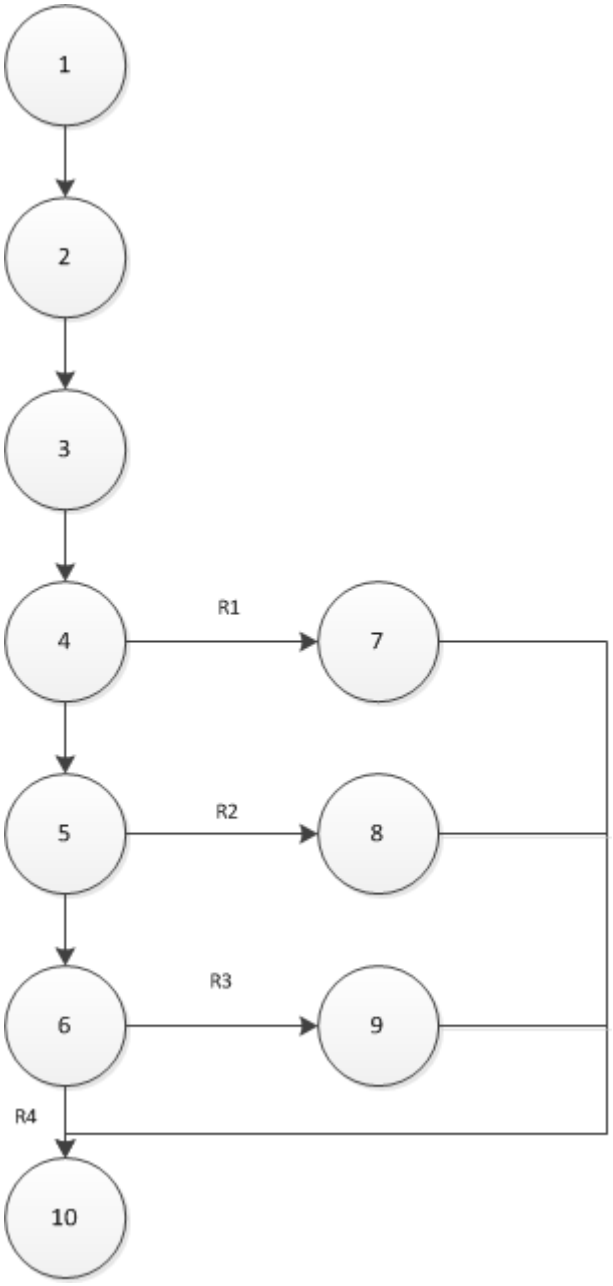
b. *White box* menjalankan aplikasi

1) *Flowchart* menjalankan aplikasi



Gambar 4.25 *Flowchart* menjalankan aplikasi

2) *Flowgraph* menjalankan aplikasi



Gambar 4.26 *Flowgraph* menjalankan aplikasi

3) Proses perhitungan menjalankan aplikasi

Dari gambar *flowgraph* diatas dapat dilakukan proses perhitungan sebagai berikut :

a) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

Dengan Rumus : V (G) = E – N + 2

N(*node*) = 10

E(*edge*) = 12

P (*predikat node*) = 3

Penyelesaian : V(G) = E – N + 2

= 12- 10 + 2

= 4

Predikat Node (N) = P + 1

= 3 + 1

= 4

b) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 4

c) *Independent Path* pada *flowgraph* diatas adalah :

Path 1 = 1 – 2 – 3 – 4 – 7 - 10

Path 2 = 1 – 2 – 3 – 4 – 5 – 8 - 10

Path 3 = 1 – 2 – 3 – 4 – 5 – 6– 9 - 10

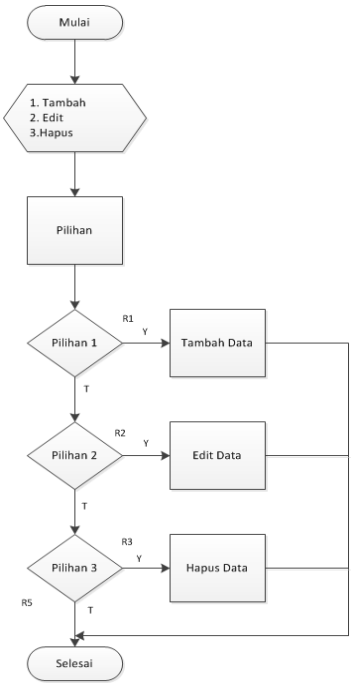
Path 4 = 1 – 2 – 3 – 4 – 5 –6 - 10

d) Grafik Matriks Menjalankan Aplikasi

Tabel 5.10 Grafik matriks menjalankan aplikasi

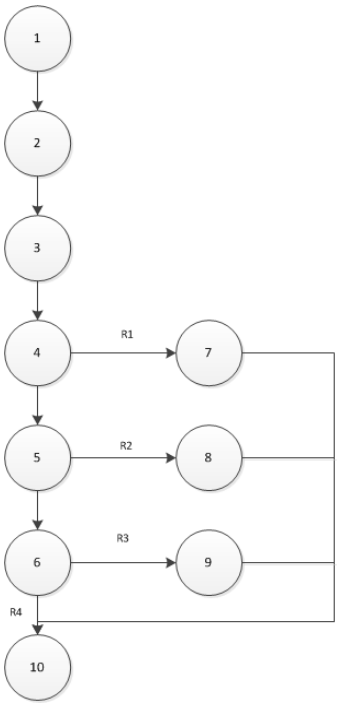
	1	2	3	4	5	6	7	8	9	10	E-1
1		1									1-1=0
2			1								1-1=0
3				1							1-1=0
4					1		1				2-1=1
5						1		1			2-1=1
6									1	1	2-1=1
7										1	1-1=0
8										1	1-1=0
9										1	1-1=0
10											0
SUM(E+1)											3+1=4

1. Flowchart *vuforia* data



Gambar 4.27 *Flowchart vuforia* data

1) *Flowgraph vuforia* data



Gambar 4 .28 *Flowgraph vuforia* data

2) Proses perhitungan web data

Dari gambar *flowgraph* diatas dapat dilakukan proses perhitungan sebagai berikut :

3) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

Dengan Rumus : V (G) = E – N + 2

N(*node*) = 10

E(*edge*) = 12

P (*predikat node*) = 3

Penyelesaian : V(G) = E – N + 2

= 12- 10 + 2 = 4

Predikat *Node* (N) = P + 1

= 3 + 1

= 4

4) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 4

5) *Independent Path* pada *flowgraph* diatas adalah :

Path 1 = 1 – 2 – 3 – 4 – 7 - 10

Path 2 = 1 – 2 – 3 – 4 – 5 – 8 - 10

Path 3 = 1 – 2 – 3 – 4 – 5 – 6– 9 - 10

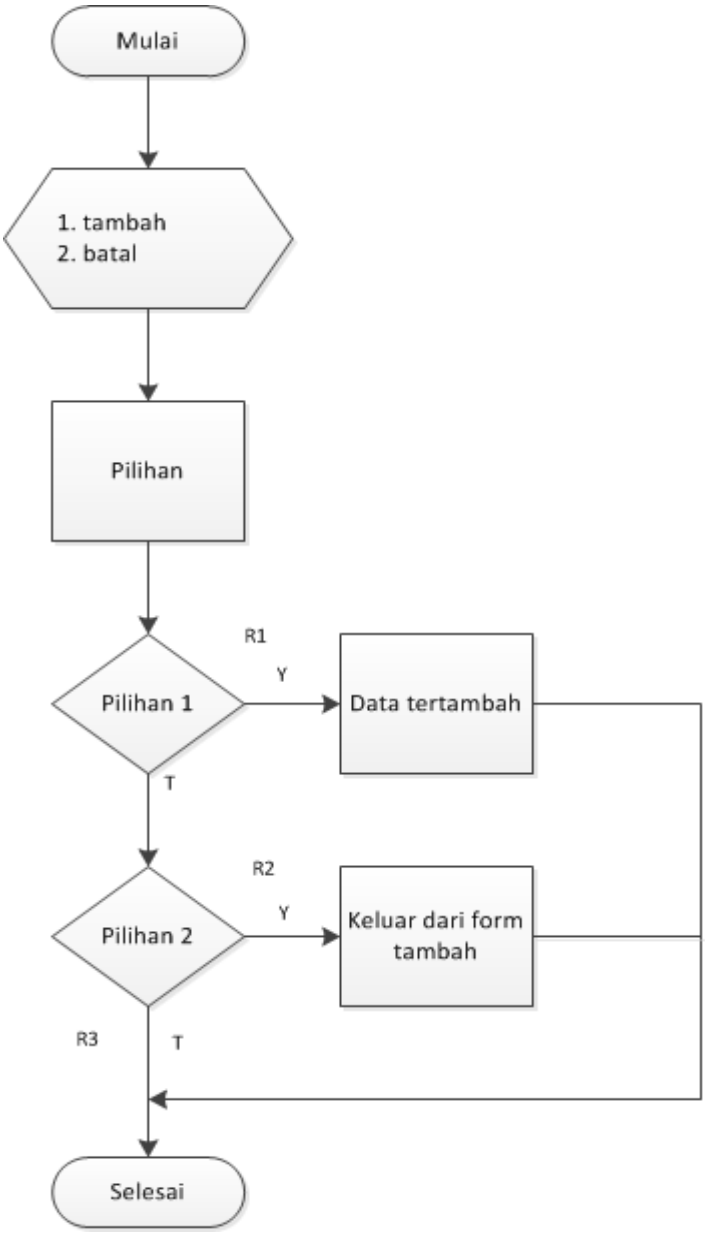
Path 4 = 1 – 2 – 3 – 4 – 5 –6 - 10

6) Grafik Matriks *vuforia*

Tabel 4.11 Grafik Matriks *vuforia*

	1	2	3	4	5	6	7	8	9	10	E-1
1		1									1-1=0
2			1								1-1=0
3				1							1-1=0
4					1		1				2-1=1
5						1		1			2-1=1
6									1	1	2-1=1
7										1	1-1=0
8										1	1-1=0
9										1	1-1=0
10											0
SUM(E+1)											3+1=4

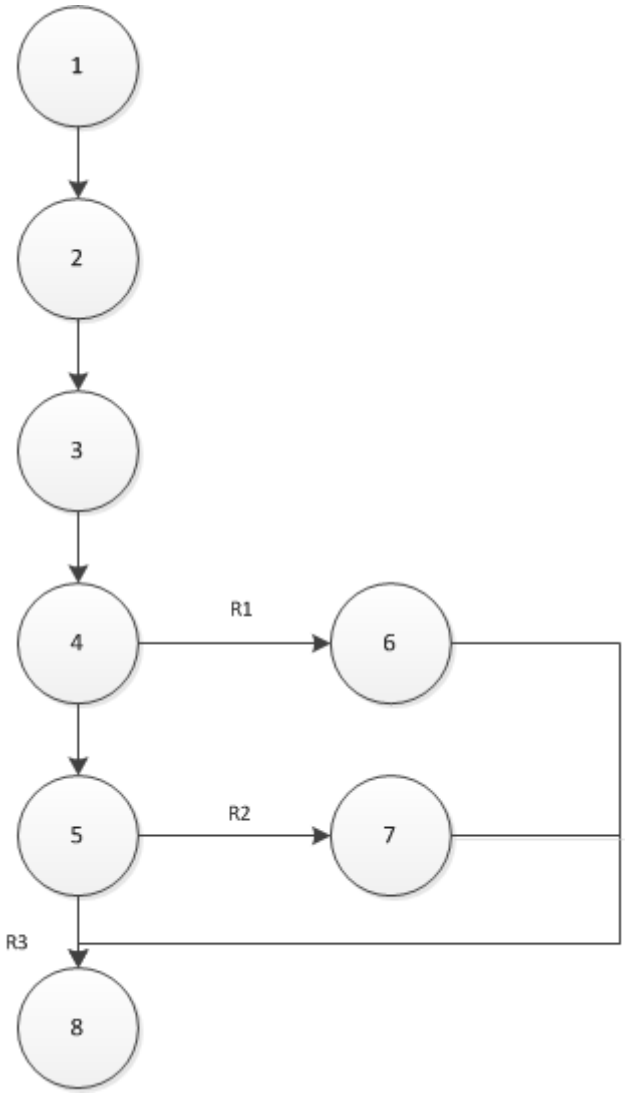
c. *White box vuforia* berdasarkan *form* tambah



Gambar 4.29 *Form* tambah data

1) *Flowchart form* tambah

2) *Flowgraph form* tambah data



Gambar 4.30 *Flowgraph form* tambah

3) Proses perhitungan *form* tambah

Dari gambar *flowgraph* diatas dapat dilakukan proses perhitungan sebagai berikut :

a) Menghitung C*yclomatic Complexcity* V(G) dari *Egde* dan *Node*:

Dengan Rumus : V (G) = E – N + 2

N(*node*) = 8

E(*edge*) = 9

P (*predikat node*) = 2

Penyelesaian : V(G) = E – N + 2

= 9 – 8 + 2

= 3

Predikat Node (N) = P + 1

= 2 + 1

= 3

b) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 3

c) *Independent path* pada *flowgraph* diatas adalah :

Path 1 = 1 – 2 – 3 – 4 – 6 – 8

Path 2 = 1 – 2 – 3 – 4 – 5 – 7 – 8

Path 3 = 1 – 2 – 3 – 4 – 5 – 8

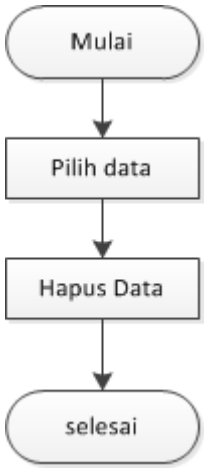
d) Grafik Matriks *Form* tambah

Tabel 4.12 matriks *form* tambah

	1	2	3	4	5	6	7	8	9	10	E-1
1		1									1-1=0
2			1								1-1=0
3				1							1-1=0
4					1	1					2-1=1
5							1	1			2-1=1
6								1			1-1=0
7								1			1-1=0
8											0
SUM(E+1)											2+1=3

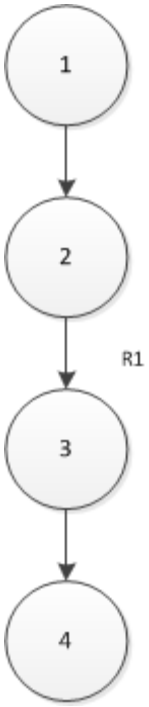
d. *White box* hapus data

1) *Flowchart* hapus data



Gambar 4.31 Flowchart Hapus Data

2) *Flowgraph* hapus data



Gambar 4.32 *Flowgraph* Hapus Data

3) Proses Perhitungan Hapus Data

Dari gambar *flowgraph* diatas dapat dilakukan perhitungan sebagai berikut :

a) Menghitung *cyclomatic complexcity* V(G) dari Edge dan Node:

Dengan Rumus : V (G) = E – N + 2

N(*node*) = 4

E(*edge*) = 3

P (*predikat node*) = 0

Penyelesaian :

V(G) = E – N + 2

= 3 – 4 + 2

= 1

Predikat Node (N) = P + 1

= 0 + 1

= 1

b) Berdasarkan perhitungan C*yclomatic Complexcity* dari *Flowgraph* diatas memiliki *Region* = 1

c) *Independent Path* pada *flowgraph* diatas adalah :

Path 1 = 1 – 2 – 3 – 4

d) Grafik Matriks Hapus Data

Tabel 5.14 Grafik Matriks Hapus Data

	1	2	3	4	E-1
1		1			1-1=0
2			1		1-1=0
3				1	1-1=0
4					0
SUM(E+1)					0+1=1

2. Pengujian *Black Box*

   a. *Black Box Popup* In*form*asi dengan jaringan yang tidak stabil

Test Factor	Hasil	Keterangan
Jika mendeteksi *wajah* dengan kondisi yang baik	✓	Berhasil, Karena tampil deteksi wajah pertama.
ScreenShoot		



Gambar 4.24 *Black Box* deteksi wajah pertama

b. Uji akurasi dari pendeteksian *wajah* pada *sample* yang digunakan adalah sebagai berikut :

Tabel 4.15 Uji Akurasi

Jarak	Wajah pertama	Wajah kedua	Wajah ketiga	Wajah keempat	Ket
30 cm	✓	✓	✓	✓	✓ = Tampil X = Tidak Tampil
60 cm	✓	✓	✓	✓	
75 cm	✓	✓	✓	✓	
1 m	✓	✓	✓	✓	
2.5 m	✓	✓	✓	✓	
3.5 m	X	X	X	X	
Derajat Kemiringan	Wajah pertama	Wajah kedua	Wajah ketiga	Wajah keempat	
15º	✓	✓	✓	✓	
30º	✓	✓	✓	✓	
45º	✓	✓	✓	✓	
60º	✓	✓	✓	✓	
75º	X	X	X	X	
90º	X	X	X	X	

Tabel 4.15 Deteksi Jarak

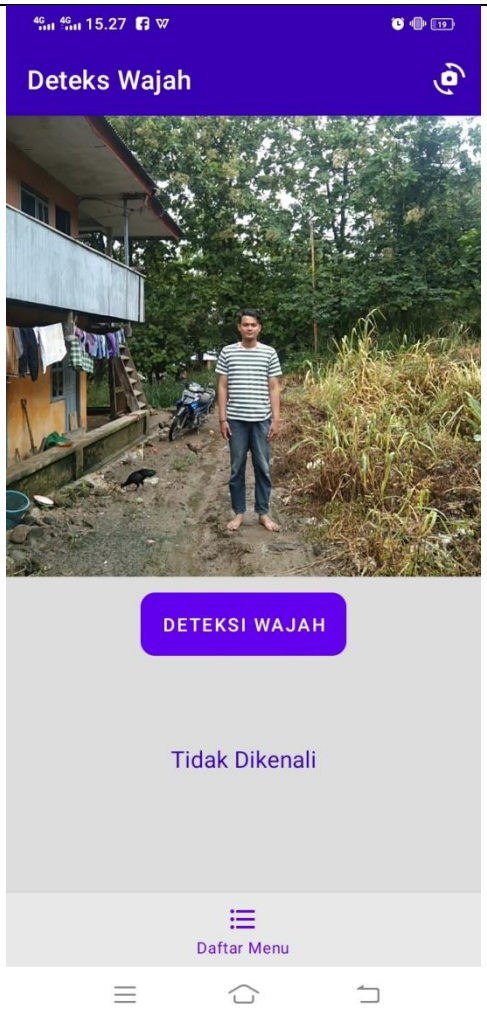
JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
45 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

60 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
75 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

90 CM			Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1 M			Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

1,5 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
2 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

2,5 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

3,5 M		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel 4.16 Deteksi Kemiringan

KEMIRINGAN	GAMBAR	KET
15^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
30^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

45^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

60^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
75^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

90^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

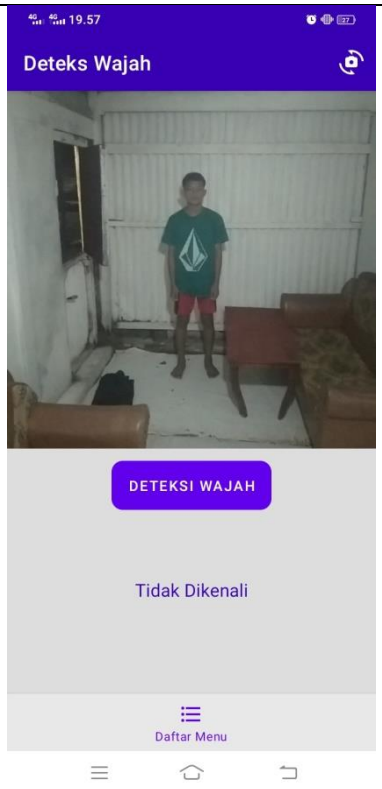
Tabel. 4.17 Deteksi Jarak Pengujian ke 1

JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

1,5 M			Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel. 4.18 Deteksi Jarak Pengujian ke 2

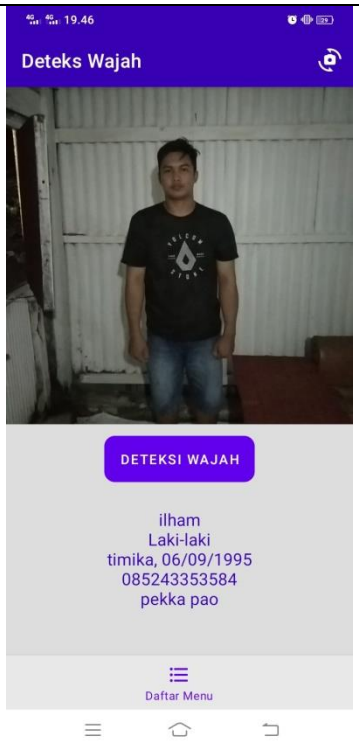
JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1,5 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

2 M			Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel. 4.19 Deteksi Jarak Pengujian ke 3

JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1M		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

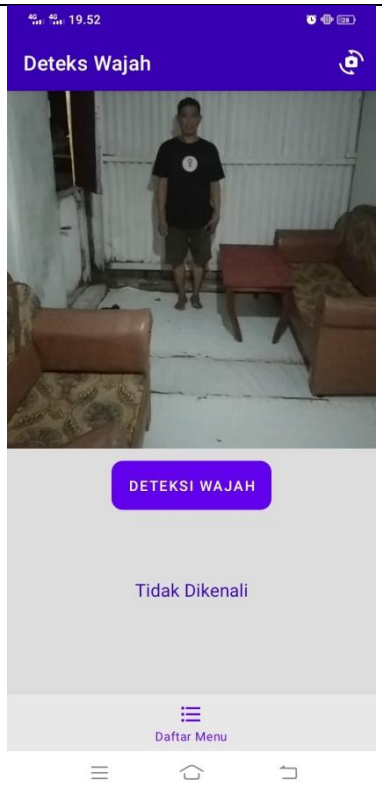
Tabel. 4.20 Deteksi Jarak Pengujian ke 4

JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

1,5 M			Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

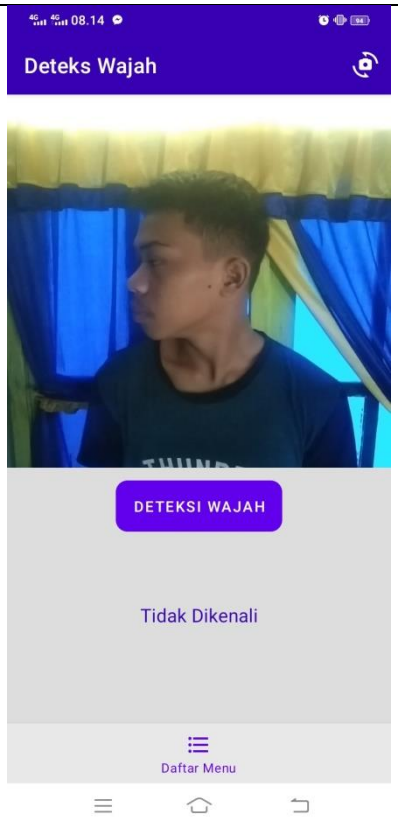
Tabel. 4.21 Deteksi Jarak Pengujian ke 5

JARAK	GAMBAR	KET
30 CM		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
1,5 M		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

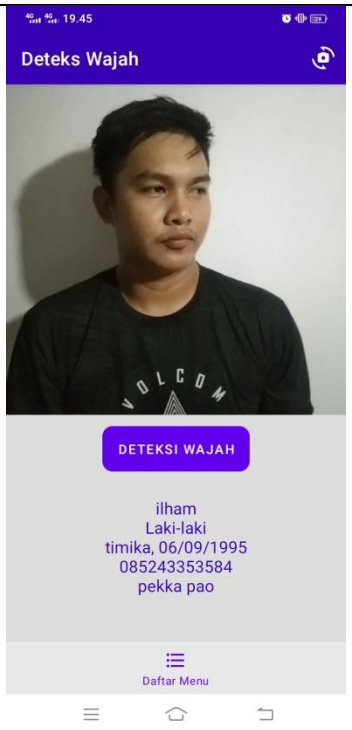
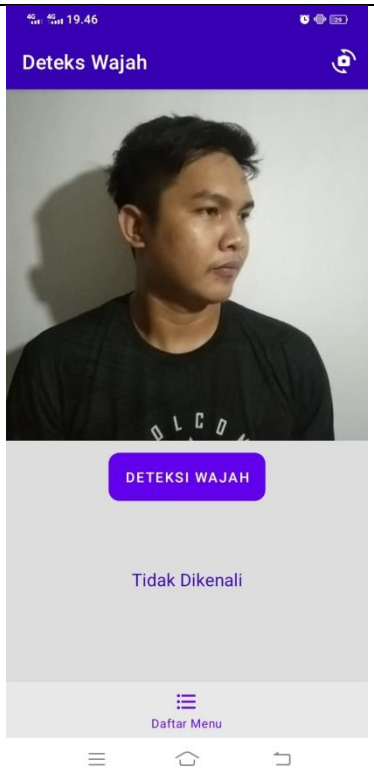
2 M			Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel 4.22 Deteksi Kemiringan Pengujian ke 1

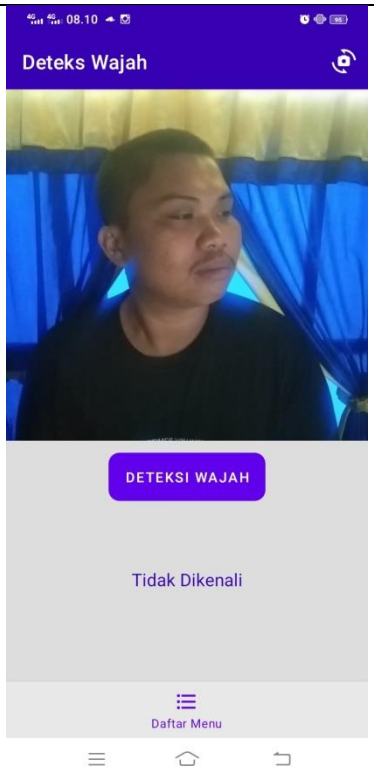
KEMIRINGAN	GAMBAR	KET
15^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
60^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

90^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

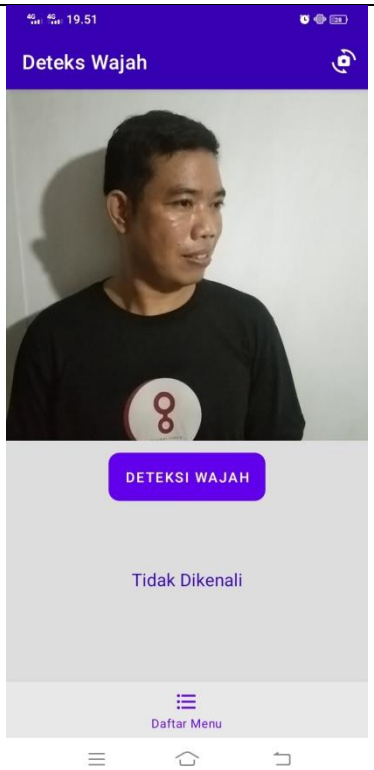
Tabel 4.23 Deteksi Kemiringan Pengujian ke 2

KEMIRINGAN	GAMBAR	KET
30^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
75^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

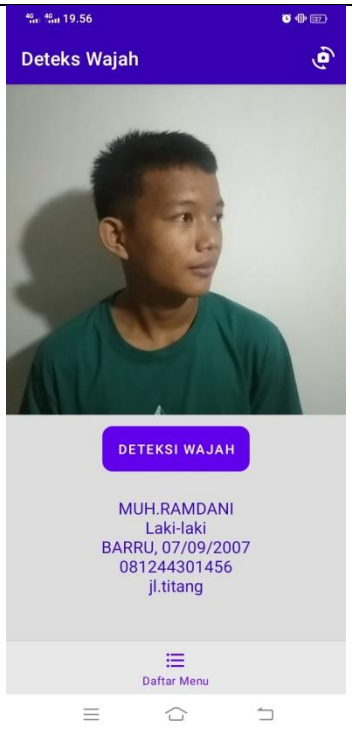
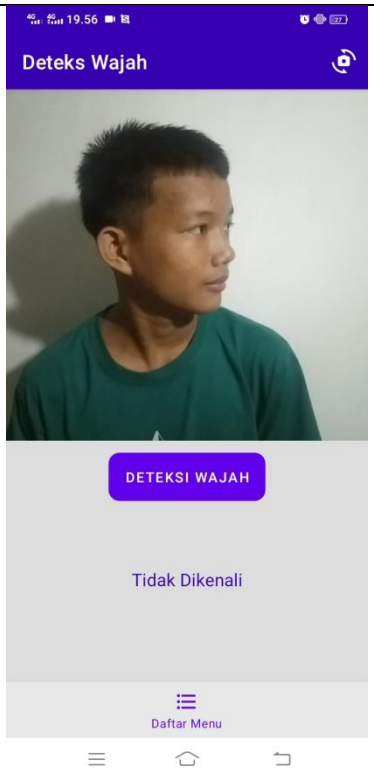
Tabel 4.24 Deteksi Kemiringan Pengujian ke 3

KEMIRINGAN	GAMBAR	KET
15^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
75^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

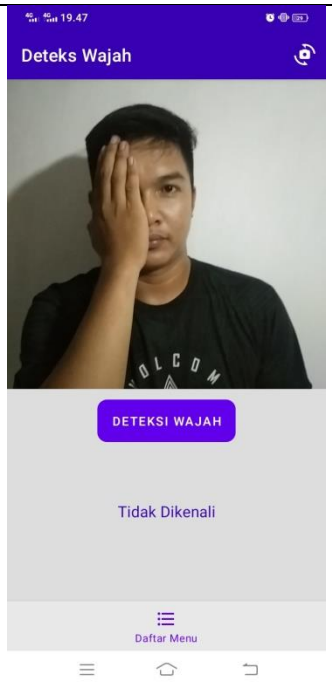
Tabel 4.25 Deteksi Kemiringan Pengujian ke 4

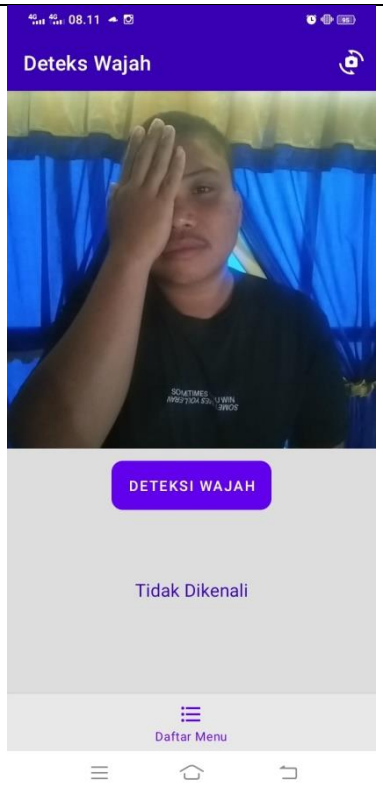
KEMIRINGAN	GAMBAR	KET
30^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
60^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel 4.26 Deteksi Kemiringan Pengujian ke 5

KEMIRINGAN	GAMBAR	KET
60^0		Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
90^0		Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

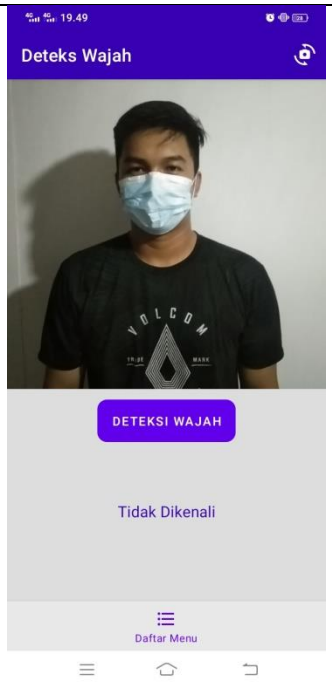
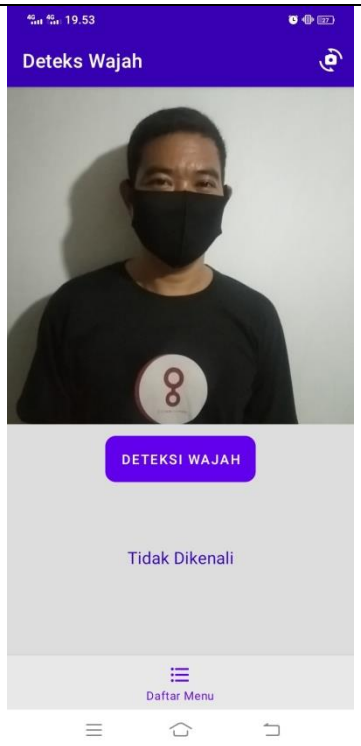
Tabel 4.27 Deteksi wajah separuh ditutup

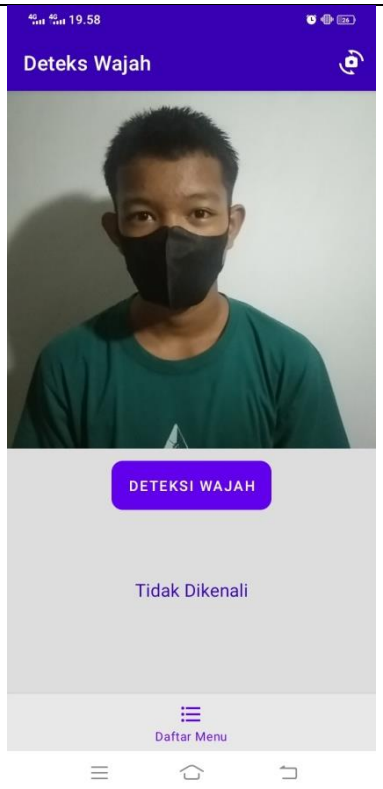
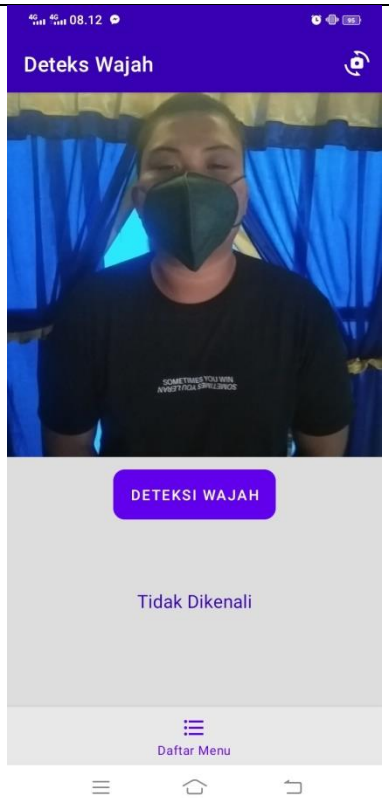
GAMBAR	KET
	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
	Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

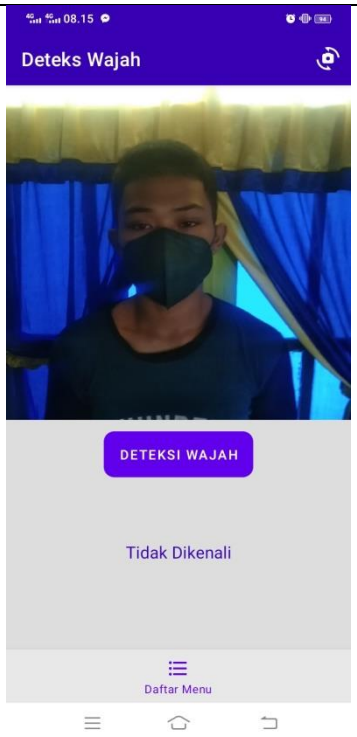
	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
	Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

	Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

Tabel 4.28 Deteksi wajah memakai masker

GAMBAR	KET
	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp
	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

	Tidak Terdeteksi Kamera Depan = 8 MP Kamera Belakang = 13mp+2mp

BAB V

KESIMPULAN DAN SARAN

A. KESIMPULAN

Dari hasil implementasi program dan analisa hasil uji coba dapat disimpulkan bahwa :

a. Titik tengah didapat dengan cara *centroid* dapat berjalan maksimal dengan bantuan perbaikan citra dikarenakan titik tengah tepat maka jarak yang dihasilkan oleh wajah pun juga tepat sehingga pada saat penghitungan *geometric mean* wajah menghasilkan nilai yang berbeda. Citra yang memiliki selisih *geometric mean atau Error Value* terendah dikatakan sebagai *output* terbaik dari hasil pengenalan wajah.
b. Mengenai aplikasi deteksi wajah menghasilkan pada hasil teratas menunjukan angka 80% dari segi jarak ,untuk hasil kemiringan menghasilkan 75%, pada hasil menutup wajah separuh yaitu 50% dan terakhir dari hasil menggunakan masker 0% , data uji yang diambil dari data pengujian aplikasi dan sudah melakukan daftar wajah dan tersimpan di *database*.

B. SARAN

Terdapat banyak kekurangan dalam penelitian ini, berikut beberapa saran sebagai bahan pengembangan penelitian selanjutnya yaitu :

a. Pengambilan citra tidak di-atur pencahayaannya sehingga tidak akan berpengaruh ketika pendeteksian tepi.
b. Perlu pengoptimalan untuk menginputkan citra *database.*
c. Perlu diadakan penelitian lagi jika citra yang digunakan tidak *frontal* dan bentuk wajah tidak sesuai dengan ukuran kotak fitur
d. Perlu diadakan penelitian lanjutan untuk mendapatkan tingkat akurasi yang lebih baik.

DAFTAR PUSTAKA

Adi, n. (2011). *Perancangan dan implementasi sistem basis data.* yokyakarta: Andi.

Card, & shneiderman, m. (2017). *Devinisi visualisasi.* surabaya: institut bisnis informatika.(online)*(https://id.m.wikipedia.org/wiki/Visualisasi)*

Daulay, M. S. (2018). *Mengenal Hadware dan Sofware dan Pengelolaan Instansi Komputer.* Yokyakarta: Andi.

Eko, P. (2017). Data mining konsep dan aplikasi menggunakan matlab. Yokyakarta: Andi.

Fernando, M. (2018). Strategi pengembangan kawasan sebagai sebuah tujuan wisata. Tesis PS. Megister kajian Pariwisata. bandung: universitas gajah mada.

Fernando, M. (2016). *Membuat aplikasi android augmented Reality menggunakan Vuforia SDK dan Unity. Jebres.* solo: AROnline.

Harita. (2021). *Pengertian Augmented Reality.http://www.haritsthinkso/2021/12augmented-reality-adalah-teknologi-yang.html.* diakses tanggal 17 Maret 2021

Munir, R. (2019). Algoritma dan pemrograman dalam bahasa pascal dan C# edisi ke-3. bandung: Informatika.

Rosikhan, Y. M., & Aristiawan. *Unity 3D- Game Engine.* (online)*(http://www.hermantolle.com/class/docs/unity-3d-game-engine)* (diakses pada 17 Maret 2021)

Sholiq. (2006). Pemodelan sistem informasi beriorentasi objek dengan UML. yokyakarta: Graha ilmu.

Soetomo. (2019). Dasar pengertian pariwisata. (online)(https://mangkutak.wordpress.com(2021/01/05)

Suyono, A. (2019). *belajar pemrograman C#.* semarang: Andi.

Yogiyanto. (2017). *Pengenalan Komputer.* yogyakarta: andi.