BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1 Artificial Intelligence (AI)

Para ahli mendefinisikan AI secara berbeda-beda tergantung pada sudut pandang mereka masing-masing.Ada yang fokus pada logika berpikir manusia saja, tetapi ada juga yang mendefinisikan AI secara lebih luas pada tingkah laku manusia.

(Russel & Norvig, 2010: 3-5) mengelompokkan definisi AI ke dalam empat kategori, yaitu:

1. *Thinking Humanly : The cognitive modeling approach*

   Pendekatan ini dilakukan dengan dua cara. Cara pertama adalah melalui intropeksi dengan mencoba menangkap pemikiran-pemikiran kita sendiri pada saat kita berpikir. Tetapi, seorang psikolog barat mengatakan "*how do you know that you understand*?". Bagaimana anda tahu bahwa anda mengerti? Karena pada saat kita menyadari pemikiran anda, ternyata pemikiran tersebut sudah lewat digantikan kesadaran anda.Sehingga definisi ini terkesan mengada-ada dan tidak mungkin dilakukan. Cara yang kedua adalah melalui eksperimen – eksperimen psikologi.

2. *Acting Humanly : The Turing Test Approach*

   Pada tahun 1950, Alan Turing merancang suatu ujian bagi komputer berinteligensia untuk menguji apakah komputer tersebut mampu mengelabuhi seorang manusia yang menginterogasinya melalui *teletype* (komunikasi berbasis teks jarak jauh). Jika *interrogator* tidak dapat membedakan yang diinterogasinya adalah manusia atau komputer, maka komputer berinteligensia tersebut lolos dari *Turing Test*. Komputer tersebut perlu memiliki kemampuan: *Natural Language Processing, Knowledge Representation, Automated Reasoning, Machine Learning, Computer Vision, Robotics*. *Turing test* sengaja menghindari interaksi fisik antara *interrogator* dan komputer karena simulasi fisik manusia tidak memerlukan intelegensia.

3. *Thinking Rationally : The "Laws of Thought" Approach*

   Terdapat dua masalah dalam pendekatan ini. Masalah pertama adalah tidak mudah untuk membuat pengetahuan informal dan menyatakan pengetahuan tersebut ke dalam *formalterm* yang diperlukan oleh notasi logika, khususnya

ketika pengetahuan tersebut memiliki kepastian kurang dari 100%.Masalah kedua adalah terdapat perbedaan besar antara dapat memecahkan masalah dalam prinsip dan memecahkannya dalam dunia nyata.

4. *Acting Rationally : The Rational Agent Approach*

   Membuat inferensi yang logis merupakan bagian dari suatu *rational agent*. Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan menalar secara logis, maka bisa didapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Jika mencapai tujuan, maka *agent* dapat melakukan aksi berdasarkan kesimpulan tersebut.

Russell dan Norvig dalam bukunya yang berjudul Artificial Intelligence (AI) mengatakan bahwa saat ini AI mencakup variasi bidang ilmu yang cukup luas, mulai dari yang umum (learning dan perception) hingga yang khusus, seperti permainan catur, pembuktian teorema matematika, bagaimana berkendara menggunakan mobil pada jalan yang padat, membangun robot cerdas dan juga yang bermanfaat untuk banyak orang adalah AI dalam fungsinya sebagai pendiagnosa penyakit.

Secara garis besar bidang ilmu yang dipelajari dalam domain AI dikelompokan sebagai berikut :

1. *Expert System*

   Bidang ilmu ini mempelajari bagaimana membangun sistem atau komputer yang memiliki keahlian untuk memecahkan masalah dengan meniru atau mengadopsi keahlian yang dimiliki oleh pakar. Dengan sistem ini, permasalahan yang seharusnya hanya bisa diselesaikan oleh para pakar/ahli, dapat diselesaikan oleh orang biasa/awam. Sedangkan, untuk para ahli, sistem pakar juga akan membantu aktivitas mereka sebagai asisten yang seolah olah sudah mempunyai banyak pengalaman.

2. *Natural Language Processing (NLP)*

   NLP mempelajari bagaimana bahasa alami itu diolah sedemikian hingga user dapat berkomunikasi dengan komputer. Konsentrasi ilmu ini adalah interaksi antara komputer dengan bahasa natural yang digunakan manusia, yakni bagaimana komputer melakukan ekstraksi informasi dari input yang berupa natural language dan atau menghasilkan output yang juga berupa natural language.

3. *Robotics and Sensory System*

   Robot mampu melakukan beberapa task dengan berinteraksi dengan physical world.Untuk melakukan hal tersebut, robot diperlengkapi dengan effector seperti lengan, roda, kaki, dll.Kemudian, robot juga diperlengkapi dengan sensor, yang memampukan mereka untuk menerima dan bereaksi terhadap *environment* mereka.

   Bidang ilmu inilah yang mempelajari bagaimana merancang robot yang mampu membantu manusia, bahkan yang nantinya bisa menggantikan fungsi manusia.

4. *Computer System*

   Cabang ilmu ini erat kaitannya dengan pembangunan arti/makna dari image ke obyek secara fisik. Yang dibutuhkan didalamnya adalah metode-metode untuk memperoleh, melakukan proses, menganalisa dan memahami image. Apabila cabang ilmu ini dikombinasikan dengan *Artificial Intelligence* secara umum akan mampu menghasilkan sebuah visual *intelligence system*.

5. *Game Playing*

   Game biasanya memiliki karakter yang dikontrol oleh user, dan karakter lawan yang dikontrol oleh game itu sendiri. Dimana kita harus merancang aturan-aturan yang nantinya akan dikerjakan oleh karakter lawan. Game akan menjadi menarik apabila karakter lawan (*non-player)* bereaksi dengan baik terhadap apa yang dilakukan oleh *player*. Hal ini akan memancing penasaran user dan membuat game menarik untuk dimainkan. Tujuan intinya adalah membuat non-player memiliki strategi yang cerdas untuk mengalahkan *player*.Di field ini, ilmu AI dibutuhkan, yaitu untuk merancang dan menghasilkan game yang fun dan menarik untuk dimainkan.

2.1.2 *Computer Vision*

Menurut (Szeliski, 2011: 5) *Computer Vision* merupakan ilmu yang bertujuan untuk mendiskripsikan dunia dalam bentuk satu atau lebih citra dan melakukan rekonstruksi properti-properti yang ada seperti bentuk, pencahayaan, dan distribusi warna.

Computer Vision banyak digunakan dalam berbagai aplikasi salah satu bidangnya adalah *Face Recognition*. *Face Recognition* digunakan untuk pengenalan

wajah dan meningkatkan focus terhadap kamera serta pencarian gambar yang lebih relevan. *Computer vision* merupakan metode untuk memperoleh, mengolah, menganalisis, dan mendefinisikan gambar, seperti tinggi dimensi data dari dunia nyata untuk menghasilkan informasi numerik atau simbolis.

2.1.3 Interaksi Manusia dengan Komputer

Menurut (Shneiderman & Plaisant, 2010: 88) terdapat delapan aturan emas (*Eight Golden Rules*) dalam merancang *user interface* pada aplikasi, sehingga aplikasi yang dibangun sesuai dengan standar yang telah ditetapkan. Delapan aturan tersebut adalah

1. Konsisten

   Aturan ini memberikan informasi agar aplikasi yang dikembangkan memiliki aturan yang pasti dan tidak berubah-ubah di dalam suatu aplikasi. Misalnya aksi yang dilakukan dalam setiap proses yang mirip harus memiliki urutan yang sama, penggunaan jenis huruf, tata letak, warna dan penulisan juga diperhatikan di dalam aturan ini.
2. Memenuhi kebutuhan universal

   Aturan ini memberikan informasi agar aplikasi yang dikembangkan harus memperhatikan kebutuhan pengguna yang bervariasi. Variasi ini dapat berupa penggunaan simbol gambar yang sudah digunakan secara umum, sehingga pengguna tidak mengalami kebingungan dalam menggunakan simbol yang ada.
3. Memberikan umpan balik yang informatif

   Aturan ini memberikan informasi agar aplikasi yang dikembangkan harus mempunyai umpan balik dari sistem, sehingga penguna mengetahui informasi bahwa aplikasi telah berjalan dengan baik.
4. Memberikan dialog untuk keadaan akhir

   Setiap rangkaian aksi harus diakhiri dengan memberikan dialog pada aplikasi. Umpan balik ini akan memberikan informasi kepada pengguna bahwa aksi yang dilakukan sudah pada tahap yang terakhir.
5. Mencegah kesalahan

   Aturan ini mencegah adanya kesalahan masukan yang berasal dari pengguna. Pencegahan ini biasanya aplikasi akan memberikan informasi kepada pengguna bagian yang salah dengan menggunakan simbol atau keterangan tertentu misalnya memberikan warna merah pada tulisan.

6. Memberikan pembalikan aksi yang mudah

   Aksi yang telah dilakukan, harus dapat dikembalikan. Fitur ini menghilangkan kecemasan pengguna karena pengguna dapat membalikkan aksi yang dilakukan jika aksi tersebut menyebabkan kesalahan.

7. Mendukung pusat kendali internal

   Aplikasi yang menarik biasanya memberikan kebebasan kepada pengguna dalam mengatur aplikasi yang digunakan seperti pengaturan suara, warna dan sebagainya. Kebebasan ini tentu saja masih dalam ruang lingkup yang mampu dikerjakan oleh aplikasi tersebut dan sesuai dengan kebutuhan.

8. Mengurangi beban ingatan jangka pendek

   Manusia memiliki kapasitas yang terbatas dalam mengingat. *User interface* yang dirancang harus menghindari rancangan dimana pengguna harus mengingat informasi yang bersangkutan. Informasi tersebut dapat berupa simbol dan gambar yang digunakan pada aplikasi.

2.1.4 *Unified Modelling Language*

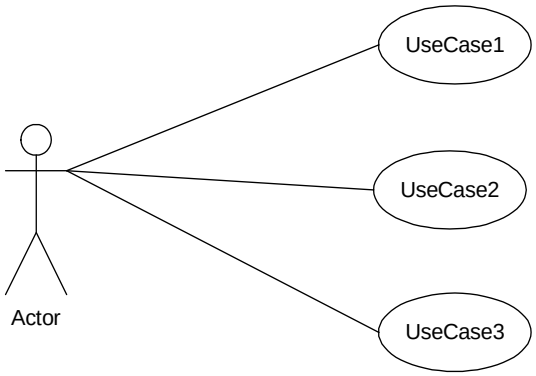
Unified Modeling Language (UML) merupakan bahasa standar untuk perancangan perangkat lunak.UML dapat digunakan untuk mendeskripsikan atau menggambarkan perangkat lunak (S.Pressman, 2010: 841).

UML memiliki banyak struktur dan diagram dalam pemodelannya. Pada pembuatan aplikasi ini akan digunakan beberapa diagram seperti *Use Case Diagram*, *Activity Diagram, Class Diagram,* dan *Sequence Diagram.*

2.1.4.1 Use Case Diagram

Use Case *Diagram* adalah diagram yang menggambarkan interaksi antara sistem, eksternal sistem, dan pengguna. Diagram ini menyediakan informasi mengenai siapa saja yang akan menggunakan sistem tersebut dan bagaimana cara untuk menggunakannya.

Komponen-komponen dalam *Use Case Diagram* adalah sebagai berikut :



Gambar 2.1*Use Case Diagram*

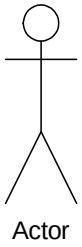
a) Use Case

Use Case menggambarkan apa yang dapat dilakukan oleh *actor* terhadap sistem, baik secara otomatis maupun manual.

Gambar 2.2 Contoh*Use Case*

b) Actor

Actor adalah *user* yang akan berinteraksi dalam sistem dengan melakukan *use case* untuk bertukar informasi. *Actor* digambarkan dalam bentuk *stick figure* dengan label peran *actor* tersebut dalam sistem.



Gambar 2.3*Actor*

2.1.4.2 Use Case Narrative

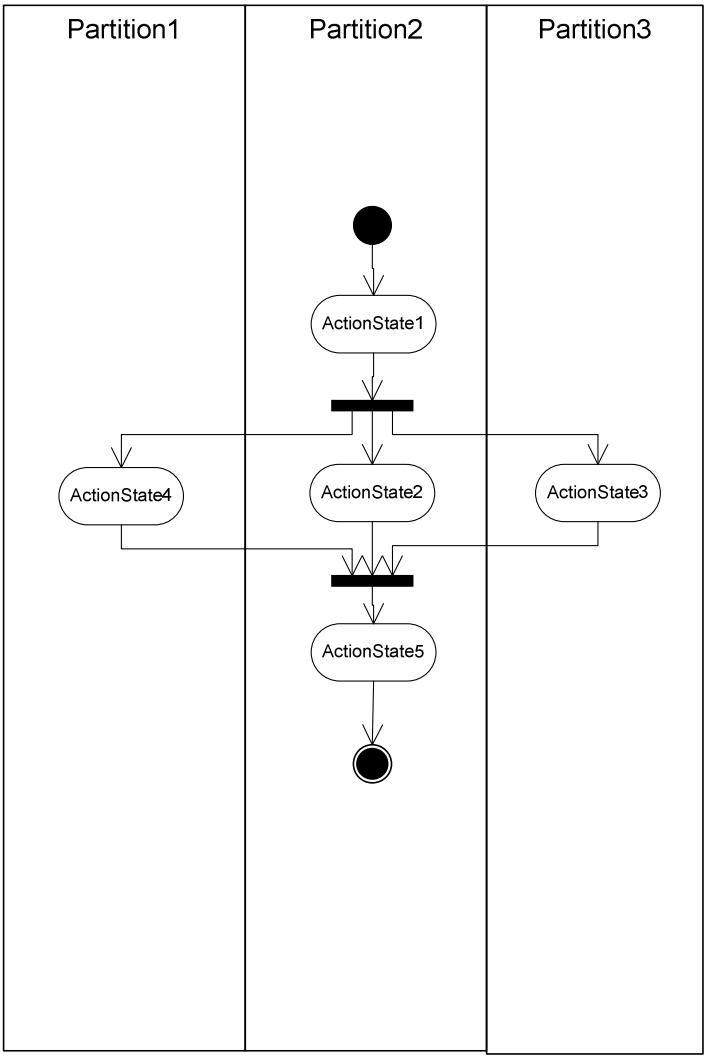
Use Case Narrative merupakan deskripsi mengenai urutan-urutan proses dari setiap interaksi yang berguna untuk mempercepat pemahaman tentang sistem(Whitten & Bentley, 2007: 256 – 260).

Tabel 2.1Contoh *Use Case Narrative*

Nama *Use Case*	*Backup Data*	
Actor	*Admin*	
Deskripsi	*Use Case* ini mendeskripsikan tentang proses *Backup data* dan sistem akan melakukan proses *Backup data.*	
Precondition	*Actor* telah membuka aplikasi.	
Flow of Event	***Actor Action***	***System Response***
	Step 1. *Actor* menekan tombol *backup data.*	Step 2. Sistem akan melakukan proses *Backup data.*
Postcondition	*Actor* melakukan proses *Backup data.*	

2.1.4.3 Activity Diagram

*Activity Diagram*adalah diagram yang menggambarkan perilaku dinamis dari suatu atau bagian sistem melalui proses kontrol dari berbagaitindakan yang dilakukan oleh sistem. *Activity Diagram* memiliki kemiripan dengan *flowchart*, tetapi dapat memiliki proses yang konkuren. *Activity Diagram* juga dapat diberi *swimlanes* untuk menyatakan *participant*yang menjalankan *action* terkait (Pressman, 2010:853-855). Berikut ini adalah contoh *Activity Diagram* dengan *swimlanes*:



Gambar 2.4 Contoh *Activity Diagram*.

Berikut ini adalahpenjelasan istilah dalam *Activity Diagram*:

1. *Initial Node*

   Initial node digunakan untuk menggambarkan titik awal proses dalam *activity diagram*. *Initial node* digambarkan sebagai lingkaran hitam (Pressman, 2010:853).

Gambar 2.5 *Initial node*.

2. *Action Node*

   Action node digunakan untuk menggambarkan proses yang dilakukan oleh sistem dalam *activity diagram*. *Action node* digambarkan sebagai *rounded rectangle* (Pressman, 2010:853).

Gambar 2.6*Action node*.

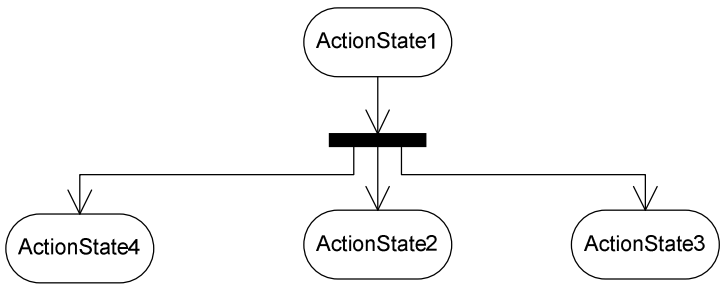
3. *Control Flow*

   Control flow digunakaan untuk menggambarkan alir dari suatu elemen ke elemen lainnya dalam *activity diagram*. *Control flow* digambarkan sebagai garis panah (Pressman, 2010:853).

Gambar 2.7 *Control flow*

4. *Fork*

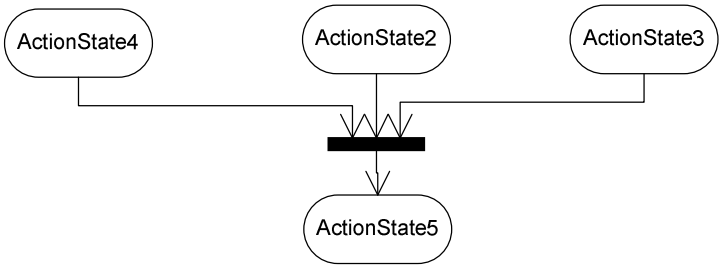
   Fork digunakan untuk menggambarkan pemisahan suatu proses menjadi dua atau lebih proses yang konkuren. *Fork* digambarkan sebagai persegi panjang hitam horizontal dengan satu panah *input* dan dua atau lebih panah *output* (Pressman, 2010:853).



Gambar 2.8 *Fork*.

5. *Join*

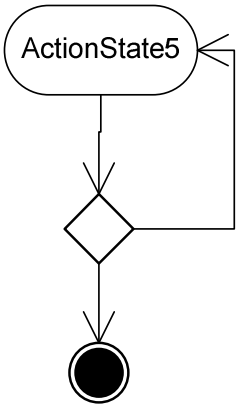
   Join digunakan untuk menggambarkan sinkronisasi proses yang konkuren. *Join* digambarkan sebagai persegi panjang hitam horizontal dengan banyak panah *input* dan satu panah *output* (Pressman, 2010:854).



Gambar 2.9 *Join*.

6. *Decision*

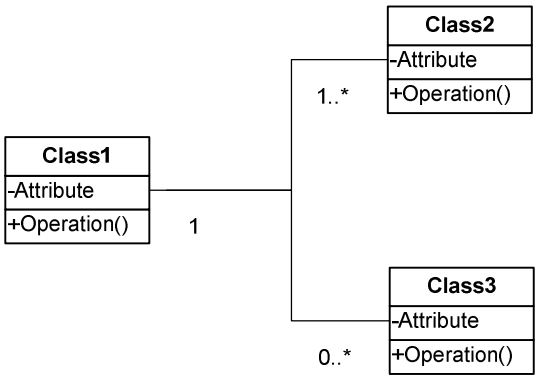
   Decision digunakan untuk menggambarkan kondisi seleksi dalam *control flow*.*Decision* digambarkan sebagai wajik dengan satu panah *input* dan dua atau lebih panah *output*. Setiap panah *output*akan diberi keterangan (Pressman, 2010:854-856).



Gambar 2.10*Decision*

2.1.4.5 Class Diagram

Class Diagram merupakan diagram yang dapat memberikan pandangan struktural dari sistem. *Class Diagram* digunakan untuk memodelkan kelas-kelas yang berisikan atribut, operasi dan hubungan relasi antar kelas yang terdapat dalam sistem (S.Pressman, 2010: 842).

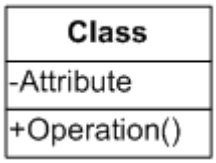


Gambar 2.11 Contoh*Class Diagram*

Berikut ini adalah istilah yang terdapat dalam *ClassDiagram* :

a. *Class*

Class merupakan *template* atau *blueprint* yang berisikan atribut dan operasi yang menggambarkan kumpulan objek yang sama. Attribut merupakan gambaran data dari suatu kelas. Sedangkan operasi digunakan untuk mengakses attribut yang terdapat dalam kelas.



Gambar 2.12 Contoh*Class*

b. *Visibility*

Visibility digunakan untuk menggambarkan informasi hak akses dari suatu atribut dan operasi yang terdapat dalam kelas (S.Pressman, 2010: 843).

Tabel 2.2 Deskripsi *Visibility*

Visibility	**Simbol**	**Keterangan**
Private	-	Atribut dan operasinya hanya dapat diakses oleh kelas yang mendefinisikannya.
Public	+	Atribut dan operasinya dapat diakses oleh kelas lainnya.
Protected	#	Atribut dan operasinya hanya dapat diakses oleh kelas yang mendefinisikan dan turunannya.

c. *Association* dan *Multiplicity*

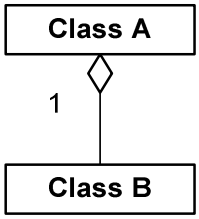
Association digunakan untuk mewakili hubungan antara kelas dan menyatakan apa yang perlu diketahui dari suatu *instance* terhadap lainnya. Sedangkan *Multiplicity* digunakan untuk menyatakan jumlah *instance* dari suatu *class* yang dihubungkan ke *class* lainnya (S.Pressman, 2010: .844 - 845).

Tabel 2.3 Deskripsi *Multiplicity*

Multiplicity	**Simbol**	**Keterangan**
No more than one	1	*Instance* yang dihubungkan dapat mempunyai tepat satu data.
Zero or one	0…1	*Instance* yang dihubungkan dapat mempunyai tepat satu data atau tidak sama sekali.
Many	*	*Instance* yang dihubungkan dapat mempunyai banyak data.
Zero or many	0…*	*Instance* yang dihubungkan dapat mempunyai banyak data atau tidak sama sekali.
One or many	1…*	*Instance* yang dihubungkan dapat mempunyai satu atau banyak data.

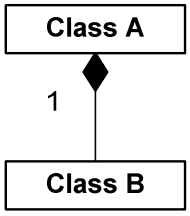
d. *Aggregation* dan *Composition*

Aggregation digunakan untuk menghubungkan antara dua *class* yang menyatakan bagian dari suatu relasi. Artinya, *class* ini (B) akan tetap ada walupun *class* induknya (A) tidak ada (S.Pressman, 2010: 845).



Gambar 2.13 Contoh *Aggregation*

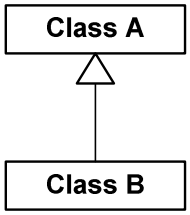
Sedangkan *Composition* digunakan untuk menghubungkan antara dua *class* yang menyatakan bagian kuat dari suatu relasi. Artinya, *class* ini (B) akan ada jika *class* induknya (A) ada (S.Pressman, 2010: 845).



Gambar 2.14 Contoh*Composition*

e. *Generalization*

Generalization digunakan untuk menggambarkan hubungan turunan antar *class* (*inheritance*). Artinya, *class* turunannya akan mewarisi sifat yang dimiliki oleh *class*induknya (S.Pressman, 2010: 843).

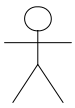


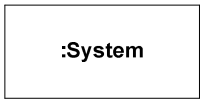
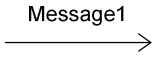
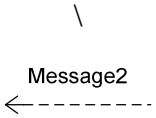
Gambar 2.15 Contoh*Generalization*

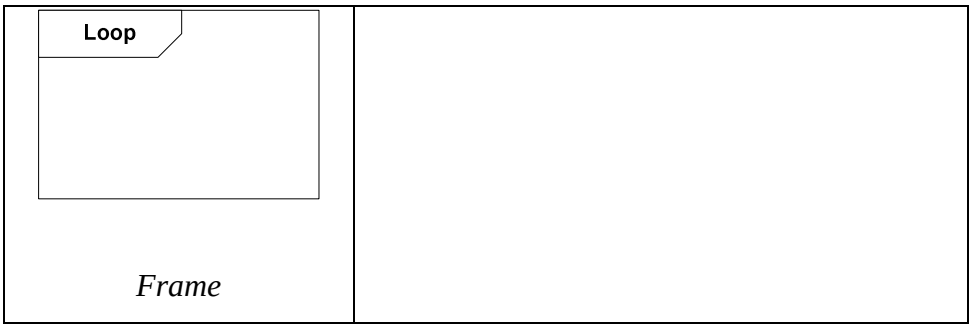
2.1.4.6 Sequence Diagram

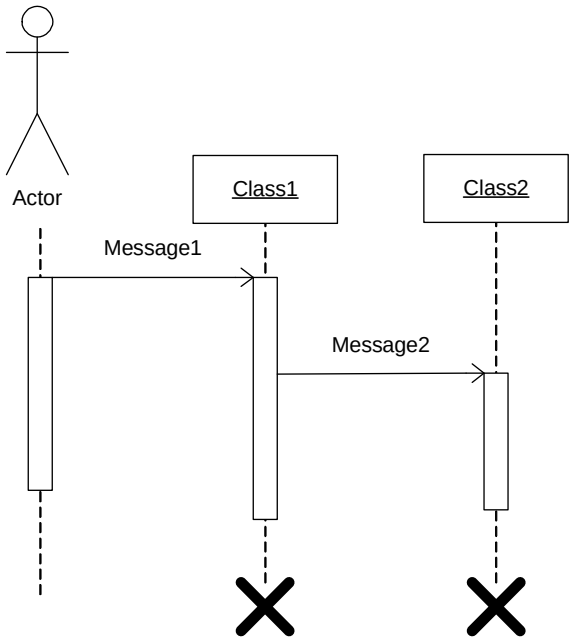
Sequence Diagram merupakan diagram yang dapat menunjukan komunikasi dinamis antara objek dengan sistem. *Sequence Diagram* digunakan untuk mendeskripsikan interaksi yang terjadi dalam sebuah *use case* ke dalam urutan waktu yang digambarkan ke dalam bentuk diagram (S.Pressman, 2010: 848).

Tabel 2.4*Notasi Sequence Diagram*

Notasi	Keterangan
 *Actor*	Notasi ini menggambarkan *user* yang berinteraksi dengan system

 *System*	Notasi ini menggambarkan kelas-kelas yang ada pada *class diagram*
 Lifelines	Notasi ini menggambarkan hidupnya objek dalam sebuah *sequence*
 *Activation Bars*	Notasi ini menggambarkan periode waktu saat proses aktif dalam interaksi
 *Input Message*	Notasi ini menggambarkan pesan masuk yang dikirimkan yaitu berupa *behavior.*
 *Output Message*	Notasi ini menggambarkan pesan yang dikirimkan sebagai balasan pesan masuk yaitu berupa *attribute.*
	Notasi ini menggambarkan sebuah pesan yang memanggil pesan dari hidupnya objek yang sama
	Notasi ini menggambarkan area pada sistem yang mengalami perulangan (*loops),* seleksi (*alternate fragments),* atau kondisi opsional (*optional).*





Gambar 2.16*Sequence Diagram*

2.1.5 Permodelan *Waterfall*

Waterfall model muncul pertama kali sekitar tahun 1970. Model ini paling banyak diguanakan dalam pembuatan program. Model ini disebut *waterfall* karena tahap demi tahap yang harus dilalui setelah menunggu tahap sebelumnya selesai dan berjalan sesuai dengan urutan. Tahap-tahap yang dilakukan dalam tahap ini:

1. System Engineering

Permodelan ini diawali dengan mencari kebutuhan dari keseluruhan sistem yang akan diaplikasikan ke dalam bentuk *software*. Hal ini sangat penting, mengingat *software* harus dapat berinteraksi dengan elemen-elemen yang lain seperti *hardware*, *database*, dan sebagainya. Tahap ini sering disebut dengan *Project Definition*.

2. ***Software Requirements Analysis*(Analisis Kebutuhan Piranti Lunak)**

   Fase ini mengumpulkan kebutuhan secara lengkap dari sistem yang akan dibuat. Kemudian membentuk *user stories* yang akan menggambarkan fitur dan fungsional *software* yang dibutuhkan sesuai dengan kebutuhan.

3. ***Design*(Perancangan)**

   Tahap ini akan dirancang *user interface* pada sistem serta arsitektur pengkodean dengan menggunakan *design pattern*. *Design pattern* adalah solusi umum yang dapat digunakan dalam permasalahan umum yang sering terjadi pada *software design* dan bersifat *object oriented programming*.

4. ***Coding*(Pengkodean)**

   Setelah melakukan prencangan *user interface,* kemudian dilakukan pengkodean dengan menggunakan bahasa pemrograman.

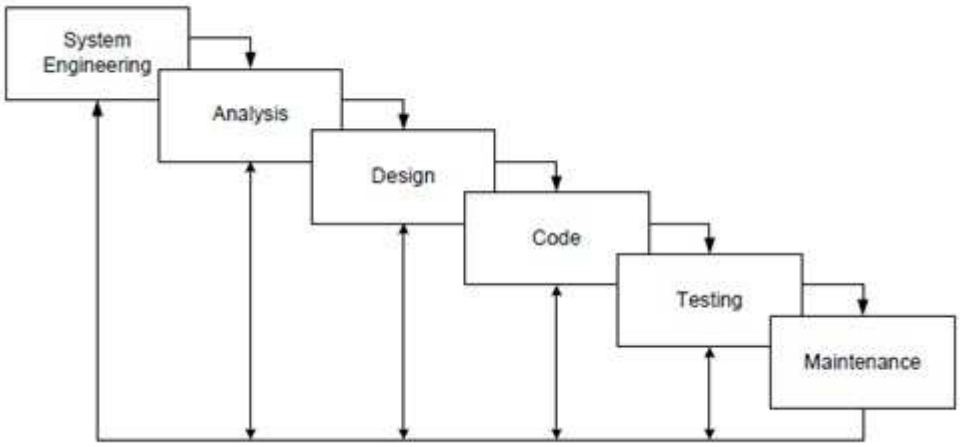
5. ***Testing* (Pengujian)**

   Dalam tahap ini dilakukan pengujian pada sistem yang sudah dibuat dengan menggunakan unit test yang sudah dibuat. Tahap ini terdiri atas dua tahap:

   a. SIT (*System Integration Test*) : uji coba terhadap sistem serta integrasi dengan sistem lainnya. Uji ini dilakukan oleh *developer team*.
   b. UAT (*User Acceptance Test*) : uji coba sistem yang dilakukan oleh *user*.

6. ***Maintenance*(Pemeliharaan)**

   Pada fase ini dilakukan pemeliharaan sistem untuk mengatasi setiap masalah-masalah yang terjadi berkenaan dengan sistem.Fase ini berakhir ketika sistem yang dibuat sudah sesuai dengan kebutuhan yang telah dianalisa dari fase awal dan tidak terjadi kesalahan ketika sistem dijalankan.



Gambar 2.17 *Waterfall Model* (Sommerville, 2011)

2.1.6 Definisi Citra, Pengolahan Citra, dan Pengenalan Pola

Citra adalah suatu representasi dari objek nyata ke dalam gambar digital yang dapat dikenali oleh komputer. Citra tersebut dapat menjadi masukan atau *input* ke dalam komputer yang akan diproses menjadi keluaran atau *output* yang diinginkan.(Gonzales & Woods, 2002: 1).

Pengolahan citra (*image processing*) merupakan bidang yang berhubungan dengan proses transformasi citra yang bertujuan untuk mendapatkan kualitas citra yang lebih baik. Pengolahan citra perlu dilakukan sebelum melakukan proses deteksi atau pengenalan citra wajah dalam penelitian ini. Hal ini dimaksudkan untuk mendapatkan kualitas citra yang lebih baik sehingga dapat mempermudah danmeningkatkan keakuratan sistem dalam melakukan pendeteksian dan pengenalan (Fairhurst, 1988: 5).

Pengenalan pola (*pattern recognition*) adalah ilmu yang mempelajari cara untuk mengenali suatu objek atau pola dengan menggunakan komputer. Pola yang ingin dikenali harus memiliki ciri yang spesifik dan di dalam satu himpunan *class*, objek harus sejenis tetapi tidak harus identik sehingga dapat dibedakan dengan objek pada *class* lain. (Dougherty, 2013: 1)

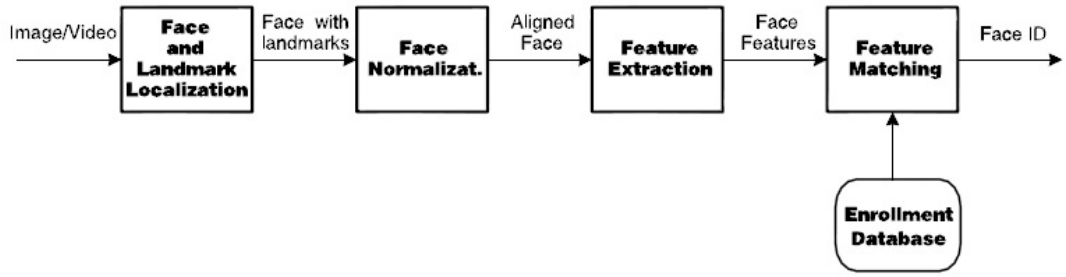
2.1.7 Identifikasi Wajah

Deteksi merupakan proses awal dalam pengenalan wajah. Pendeteksian wajah dikatakan baik jika proses tersebut mampu memberikan informasi tentang skala wajah, mendeteksi posisi bagian yang lain seperti mata, hidung, mulut dan garis wajah sehingga hasil yang didapatkan bisa memberikan informasi wajah yang lengkap dan utuh (Z.Li & K.Jain , 2011: 4).

Normalisasi dilakukan untuk menormalkan wajah secara geometris. Hal ini diperlukan untuk mengenali wajah dengan berbagai *pose* dan pencahayaan serta mengubah wajah menjadi bentuk yang lebih standar. Di dalam proses normalisasi juga dibutuh *warping* atau *morphing* yang digunakan untuk membantu normalisasi ke dalam bentuk geometri.

Ekstrasi fitur adalah langkah selanjutnya setelah normalisasi, ekstrasi ini dilakukan untuk mengambil informasi yang menonjol pada wajah yang sudah di normalisasi.Informasi tersebut digunakan untuk membedakan wajah berdasarkan

geometri wajah tersebut.Fitur wajah yang diekstrasi digunakan untuk pencocokan wajah.



Gambar 2.18 Proses pengenalan wajah (Z.Li & K.Jain , 2011: 4)

2.1.8 Pendekatan Metode Untuk Face Recognition

Secara umun, Face recognition dilaksanakan ada 4 model menurut (Tsang Tat Man, 2009: 17-18), yaitu :

1. *Knowledge Based Method*

   Model digunakan berdasarkan pengetahuan manusia untuk menemukan pola wajah sari gambar pengujian bersadarkan sifat wajah manusia.

2. *Template marching*

   Model ini digunakan dalam beberapa template untuk mengetahui wajah dan ekstrak fitur bentuk wajah .

3. *Nvariant Feature Method*

   Model ini digunakan untuk menemukan fitur wajah (alis, hidung,mata), bahkan dihadapan komposisi perpektif yang berbeda, sehingga sulit untuk menemukan wajah secara realtime menggunakan model ini. Dalam definisi nvariant pengenalan wajah yang digunakan yaitu: bentuk, tekstur dan kulit.

4. *Learning Based Method*

   Model diproses dalam satu training set sebelum melakukan deteksi. Untuk jumlah besar pelatihan, dapat diberikan tingkat pengakuan akurasi tinggi untuk variasi model wajah, ekspresi dan pose wajah. Misal wajah diimpor ke sistem. Metode yang digunakan misalnya Principal Component Analysis, Support Vector Machine, Naïve Bayes Classifier baik digunakakan untuk deteksi wajah. Berbagai metode ini cepat untuk mendeteksi wajah, dapat mendeteksi pose wajah. Dalam metode ini perlu memerlukan banyak sampel untuk pelatihan.

2.1.9 *Image Histogram*

Image Histogram merupakan penggambaran grafis dari frekuensi yang terjadi pada setiap tingkat abu-abu(*grayscale*) dalam gambar. Penyimpanan data frekuensi berupa *array 1D* dengan nilai berupa numerik, setiap elemen *array* tersebut menyimpan persentase tingkat keabuan.

Setiap data yang dimasukan ke dalam *histogram* bisa dinyatakan dalam bentuk matematika sebagai berikut:

$$h(k) = n_k = card\{(x,y)|f(x,y) = k\}$$

(Jorge Marquez Flores, 2008: 171-172)

dimana :

h = elemen dari *array*

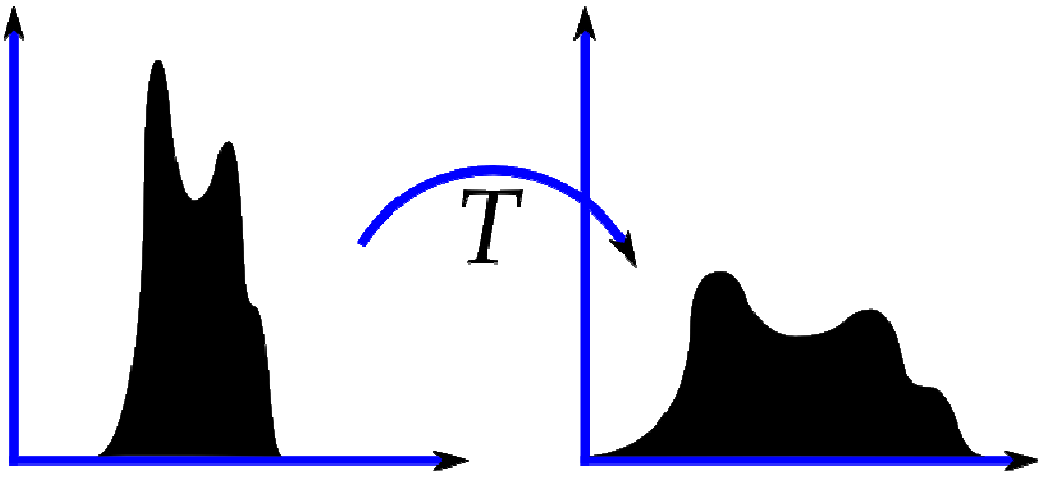
k = indeks pixel dari gambar yaitu k = 0, 1, …, L – 1

Card{…} = himpunan dari piksel yang ada

n_k= jumlah elemen dalam himpunan.

2.1.10 *Histogram Equalization*

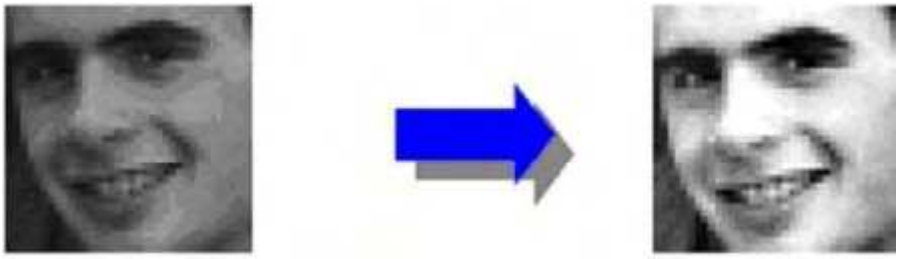
Histogram adalah suatu teknik yang biasa digunakan untuk mengubahdistribusi tingkat keabuan dari gambar agar tingkat keabuan tersebut tersebar secara merata (Tsang Tat Man, 2009: 36).



Gambar 2.19 Perubahan histogram dan setelah melakukan histogram equalization (Tsang Tat Man, 2009: 36).

Dalam grafik di atas, bentuk grafik yang telah melebar yang merupakan arti dari rata-rata tersebar di *histogram*.

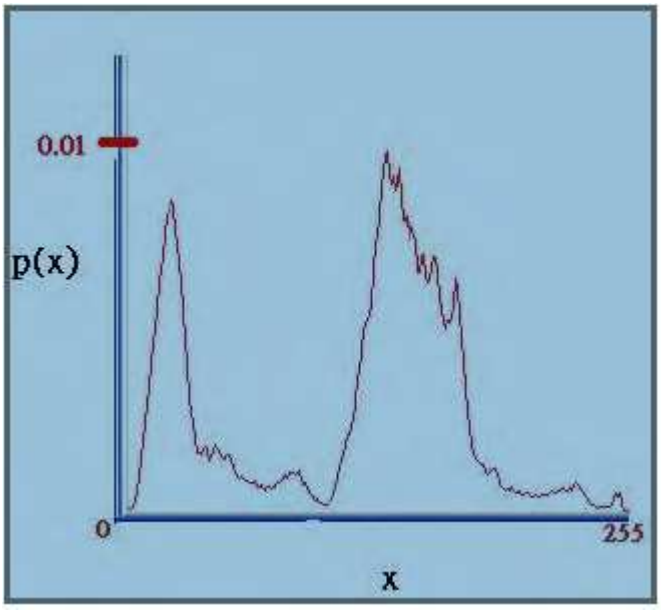
Metode ini biasanya meningkatkan kontras gambar input.Dalam sistem deteksi wajah, Sisi kiri gambar di bawah diubah ukurannya gambar *grayscale*. Selainnya adalahgambar output setelah melanjutkan pengolahan *histogram equalization.*



Gambar 2.20 Ilustrasi *Histogram Equalization*
(Tsang Tat Man, 2009: 36).

Algoritma Histogram Equalication:

1. Gambar *grayscale* memiliki Xn piksel dengan saya mewakili nilai tingkat abu-abu dalam setiap pixel. Tabel berikut ini merupakanhubungan antara probabilitas kejadian dan nilai setiap pixel:



Gambar 2.21 Bagan Probabilitas kepadatan fungsi (PDF)
(Tsang Tat Man, 2009: 36).

Dan

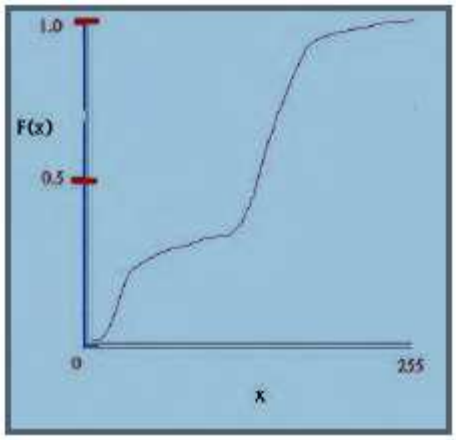
$$\sum_{i=0}^{255} p\ (x_i) = 1 \qquad (1)$$

Dimana (1) :Px sedang historgram gambar dan dinormalkan ke [0,1]

2. Definisikan fungsi distribusi kumulatif sebagai berikut

Fungsi distribusi kumulatif adalah jumlah dari seluruh nilai fungsi probabilitas untuk nilai *x*sama atau kurang dari *x.*

$$F\ (n) = \sum_{i=0}^{n} p(x_i) \qquad n = 0, 1, 2 \ldots 255 \qquad (2)$$



Gambar2.22 Bagan Kalkulasi kumulatif fungsi distribusi (Tsang Tat Man, 2009: 36).

3. Minimum dan nilai maksimum ditemukan dan diterapkan ke dalam persamaan berikut untuk mengetahui histogram pemerataan setiap piksel:

$$h(v) = round\left(\frac{cdf\,(v) - cdf_{min}}{(M\ x\ N) - cdf_{min}} \times (L - 1)\right) \qquad (3)$$

dimana (2) : cdf : cumulative distribution Function.

M : Jumlah pixel yang memiliki derajat keabuan (lebar gambar).

N : Jumlah seluruh pixel dalam citra (tinggi gambar)

L : Indeks Pixel dari gambar yaitu dari 0,1……. L-1

(tingkat abu-abu).

$f = 2x^2 + 2x - 1$

Ditanya nilai maximum dan minimum?

Jawab:

$$f'(x) = 4x + 2$$
$$f'(x) = 0$$
$$4x + 2 = 0$$
$$4x = -2$$
$$x = -\frac{1}{2}$$
$$f''(x) = 4 > 0$$

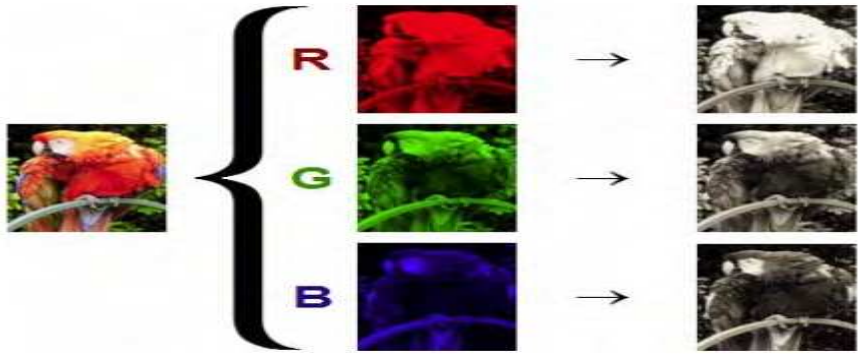
Untuk melakukan proses ini dibutuhkan fungsi yang bernama*transformation function, T(r),* dimana fungsi ini harus memenuhi 2 kriteria yaitu:

1. *T(r)* harus bersifat monoton dalam jangkauan $0 \leq r \leq L - 1$.
2. $0 \leq T(r) \leq L - 1$untuk $0 \leq r \leq L - 1$.

Dimana cdf min adalah nilai minimum dari CDF, M adalah lebar gambar dan N adalah ketinggian gambar. L mewakili Nilai tingkat abu-abu, = 256.

2.1.11 *Greyscale*

Untuk mendapatkan gambar grayscale, gambar harus dilakukan konversi kegrayscale.Setiap warna gambar (gambar RGB) terdiri dari 3 saluran untukmenyajikan komponen merah, hijau dan biru dalam ruang RGB. Dibawah ini adalah contoh untuk mengubah warna gambar menjadi gambar warna RGB (Tsang Tat Man, 2009: 34).



Gambar 2.23 Konversi Gambar Ke Greyscale
(Tsang Tat Man, 2009: 35)

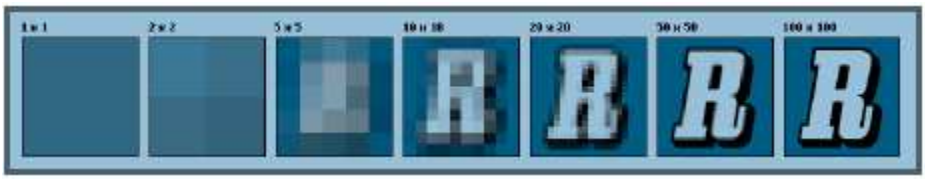
Gambar didefinisikan dengan tingkat *grayscale*, yang berarti piksel dalam gambar disimpan dalam 8-bit integer untuk mewakili warna dari hitam menjadi putih.Selain itu, gambar grayscale juga cukup untuk memproses deteksi wajah.
Konversikan ke *grayscale* gambar algoritma:

1. Berikan contoh gambar (R1, G1, B1), ...(Rn, Gn, Bn) di mana R, G, B adalah nilai dari merah, hijau dan biru masing-masing dan 'n' adalah jumlah piksel di diberikan gambar.
2. Aktifitas gambar grayscale baru memiliki piksel dari G1, ... Gn, di mana menggunakan rumus adalah sebagaiberikut:
   0.21R + 0.71G + 0.07 B = G. Tidak seperti metode rata-rata, proses ini sedang mempertimbangkan rasio.

2.1.12 *Image Resize*

Gambar diekstraksi oleh berbagai pixel yang merupakan unit kecil dalam gambar.Gambar pada pola matriks The 2-dimensi, setiap piksel dalam gambar adalah informasi sesuatu yang diwakili. Misalnya, '0' putih dan '255' hitam di gambar skala abu-abu. Gambar akan melakukan pengolahan *resize* untuk mengurangi resolusi gambar dengan menjaga jumlah yang sama. Di bawah contoh adalah ilustrasi tentang resolusi yang berbeda untuk menggambarkan citra yang sama (Tsang Tat Man, 2009: 35).

Sisi kiri atas setiap gambar adalah resolusi masing-masing.Gambar sisi kiri adalah yangasli.



Gambar 2.24 Ilustrasi *Image Resize*
(Tsang Tat Man, 2009: 35)

Gambar memiliki 3000 piksel lebar dan 2.000 piksel tinggi yang berarti memiliki 3000 x 2000 = 6.000.000 pixel atau 6 megapixel. Jika gambar telah diubah

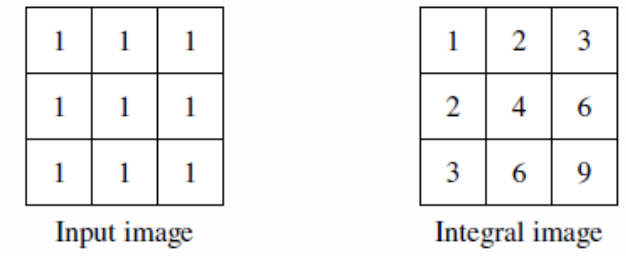
ukurannya menjadi 1000piksel lebar dan 600 pixels tinggi, gambar hanya memiliki 0,6 megapikseel. Pada sistem paling hanya menggunakan 1/10 waktu untuk menanganinya. Namun, jika Anda harus menyesuaikan ukurangambar juga akan mempengaruhi tingkat deteksi wajah.

2.1.13 *Viola-Jones Classifier*

Pada dasarnya pengolahan gambar yang standar akan mengubah skala gambar untuk ukuran yang berbeda kemudian menjalankan detektor dalam bentuk gambar-gambar yang sudah dalam ukuran yang tetap. Karena pendekatan tersebut memakan waktu yang cukup lama dalam perhitungan gambar yang mempunyai ukuran yang berbeda. Maka pada tahun 2001, Paul Viola dan Michael Jones telah menyusun skala detektor *invariant* yang membutuhkan jumlah yang sama dalam melakukan perhitungan pada ukuran gambar yang berbeda. Algoritma ini disebut juga dengan *Viola-Jones*. Algoritma detektor *Viola-Jones*mempunyai 3 fitur (Helvig Jensen, 2008: 10) yaitu:

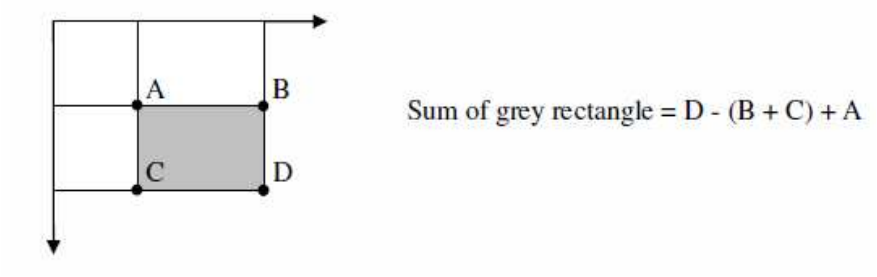
1. *The Scale Invariant detector*

   Pada tahap ini, hal pertama yang dilakukan adalah mengubah citra gambar ke dalam bentuk integral. Hal ini dilakukan dengan membuat setiap piksel mempunyai nilai yang sama dengan jumlah keseluruhan semua piksel yang berada di atas dan di kiri dari piksel yang bersangkutan.



Gambar 2.25 Gambar Integral
(Helvig Jensen, 2008: 10)

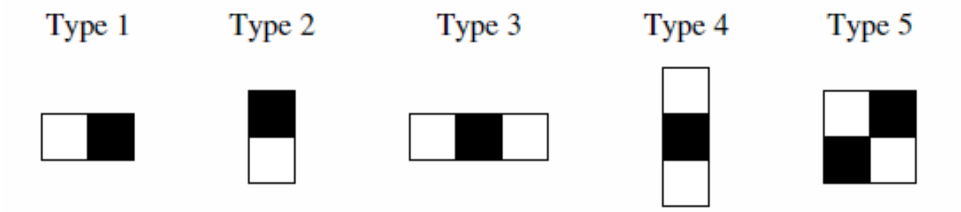
Jika *integral image* sudah dilakukan maka untuk melakukan perhitungan jumlah seluruh piksel yang berada di dalam persegi panjang hanya menggunakan empat nilai yang bertepatan dengan sudut persegi panjang (Helvig Jensen, 2008: 11)



Gambar 2.26 Perhitungan pada gambar integral

*(*Helvig Jensen, 2008: 11*)*

Dalam melakukan analisis perhitungan jumlah seluruh piksel, *Viola-Jones* memberikan *sub-window* yang berukurang 24*24 piksel dengan menggunakan dua atau lebih persegi panjang.Setiap hasil yang didapatkan di hitung dari pengurangan antara jumlah piksel yang putih dengan jumlah piksel yang hitam. (Helvig Jensen, 2008:11).



Gambar 2.27 Sub-window *persegi panjang*

(Helvig Jensen, 2008: 11)

2. *The modified AdaBoost algoritm*

Metode yang dibangun oleh *Viola-Jones* merupakan modifikasi dari algoritma *AdaBoost* yang dibuat oleh Freund dan Schapire pada tahun 1996. *AdaBoost* adalah algoritma *machine learning boosting* yang mampu membuat sebuah *classifier* kuat melalui kombinasi berat pada *classifier* lemah. Dalam matematika *classifier* lemah di representasikan sebagai berikut

$$h(x, f, p, \theta) = \begin{cases} 1 \; jika \; pf > p\theta \\ 0 \; selainnya \end{cases}$$

dimana: x adalah 24 x 24 piksel *sub-window*

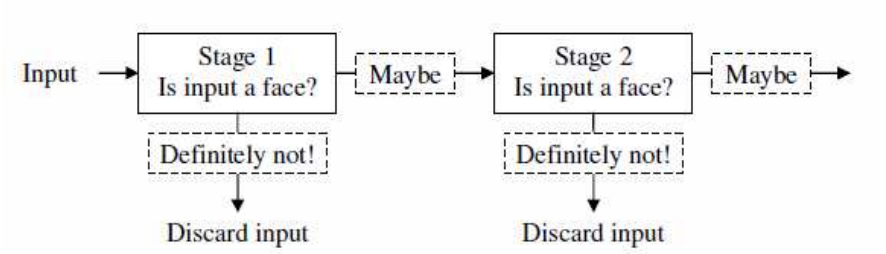
f adalah fitur yang diterapkan

p adalah fitur yang berlawanan

θ adalah*threshold* yang memilih nilai $x$dalam menentukan sebuah wajah atau bukan (Helvig Jensen, 2008: 12).

3. *The cascade classifier*

   Prinsip dasar dari algoritma *Viola-Jones* adalah mencari detektor sebanyak mungkin dalam satu gambar yang sama. Metode ini digunakan untuk menentukan bagian yang bukan wajah dan bagian wajah.Ketika *sub-window* menyatakan bukan bagian wajah maka secara automatis bagian tersebut dihilangkan dan mencari ke bagian yang lainnya.Metode ini melakukannya dalam beberapa tahap dengan memberikan *sub-window* (Helvig Jensen, 2008: 13).



Gambar 2.28 Proses *Cascade Classifier*(Helvig Jensen, 2008:13)

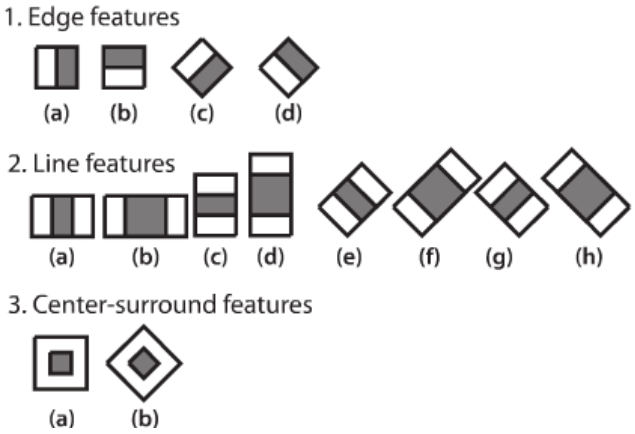
Kerangka *Viola-Jones* telah banyak digunakan oleh para peneliti untuk mendeteksi lokasi wajah dan objek dalam citra yang diberikan. Pengklasifikasi *fitur input* deteksi wajah dikenal oleh banyak peneliti, seperti *OpenCV* (Acosta et al ,2006: 273-281).

Haar Cascade Classifier menggunakan AdaBoost di setiap node dalam kaskade untuk mempelajari tingkat deteksi tinggi dengan *multi-tree classifier* penolakan pada setiap node (Budiharto, 2014:143).

Algoritma ini menggabungkan beberapa fitur inovatif, seperti:

1. Menggunakan *haar*-seperti, *threshold* yang digunakan untuk jumlah dan membedakan daerah persegi dari gambar.
2. Teknik gambar Integral yang memungkinkan perhitungan cepat untuk daerah persegi atau wilayah yang diputar 45 derajat. Struktur data ini digunakan untuk membuat perhitungan dari fitur input Haar-like lebih cepat.
3. Banyaknya proses training Meningkatkan untuk membuat klasifikasi simpul biner (ya / tidak), ditandai dengan tingkat deteksi yang tinggi dan tingkat penolakan lemah.
4. Pengorganisasian node classifier kurang baik dari kaskade penolakan. Dengan kata lain, kelompok pertama dari pengklasifikasi dipilih sehingga deteksi terbaik di wilayah citra terdiri dari obyek meskipun memungkinkan

banyak kesalahan dalam deteksi. Kelompok *classifier* berikutnya adalah deteksi terbaik kedua dengan tingkat kesalahan sedikit dan seterusnya. Dalam pengujian, sebuah objek dapat diketahui jika objek yang membuatnya melalui semua *cascades* (Qing-wiau Y, et al, 2010: 227-238). Fitur *input* Haar-seperti yang digunakan oleh classifier adalah:



Gambar 2.29 Fitur Input Haar
(Qing-wiau Y, 2010: 227-238).

Anda harus menginformasikan ke *classifier*, direktori yang akan digunakan, seperti: ***Program_Files/OpenCV/data/haarcasades/haarcascade_frontalface_default.xml.***

2.1.14 *Face Recognition In OpenCV*

Pengenalan wajah adalah tugas yang mudah bagi manusia, tetapi tidak untuk sistem komputer.Semua model pengenalan wajah di *OpenCV* 2.4 berasal dari kelas dasar abstrak *Face Recognizer*, yang menyediakan akses bersatu untuk semua algoritma wajah *recongition* di *OpenCV* (Budiharto, 2014: 141).

Algoritma yang tersedia saat ini dengan OpenCV adalah :

1. *Eigenfaces.*

2. *Fisherfaces.*

3. *Local Binary Pattern Histograms.*

Pengenalan wajah secara otomatis adalah tentang bagaimana cara mengekstraksi fitur-fitur dari sebuah gambar, menempatkan wajah ke dalam sebuah representasi yang berguna untuk melakukan beberapa jenis klasifikasi pada wajah. Pengenalan wajah berdasarkan fitur geometris wajah mungkin adalah pendekatan yang paling intuitif untuk pengenalan wajah.

Metode *Eigenfaces* dengan mengambil pendekatan holistik untuk pengenalan wajah (Turk, 1991: 71–86). Pengenalan wajah merupakan sebuah gambar wajah diambil dari titik ruang gambar dimensi tinggi dan representasi dimensi yang lebih rendah ditemukan, di mana klasifikasi menjadi mudah.Subruang dimensi yang lebih rendah ditemukan dengan *Principal Component Analysis*, yang mengidentifikasi sumbu dengan varians maksimum.Sementara jenis transformasi yang optimal dari sudut pandang rekonstruksi, tidak mengambil label kelas ke data. Sumbu dengan *varians* maksimum tidak selalu mengandung informasi diskriminatif sama sekali, maka klasifikasi menjadi tidak mungkin. Ide dasarnya adalah untuk meminimalkan varians dalam kelas, sekaligus memaksimalkan varians antara kelas pada saat yang sama.

Principal Component Analysis (PCA), yang merupakan inti dari metode *Eigenfaces*, menemukan kombinasi linear dari fitur yang memaksimalkan total varians dalam data. Sementara ini jelas merupakan cara yang ampuh untuk mewakili data, tidak mempertimbangkan setiap kelas sehingga banyak informasi diskriminatif mungkin akan hilang ketika menghilangkan komponen .Saat varians dalam data yang dihasilkan oleh sumber eksternal, komponen diidentifikasi dengan PCA tidak selalu mengandung informasi diskriminatif sama sekali, sehingga sampel yang diproyeksikan bersama-sama dan klasifikasi menjadi tidak mungkin.

2.1.15 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) dikenal juga dengan metode *Karhunen-Loeve Transformation* (KLT). Dimana metode PCA merupakan teknik standar yang digunakan dalam pengenalan pola statistikal dan pemrosesan sinyal untuk data *reduction* dan ekstraksi fitur. Sebagai pola statistical sering mengandung informasi yang berlebihan, pada pemetaan sebuah vector dapat menyingkirkan redudansi namun sebagian memperbesar instrinsik pola (Eleyan & Demirel, 2007: 94-98).

Sebuah gambar wajah dalam 2D dengan ukuran N x N dapet dianggap sebagai vector dimensi n^2. Gambar wajah yang mirip secara keseluruhan konfigurasi tidak akan secara acak didistribusikan dalam ruang gambar yang besar dengan demikian dapat dijelaskan oleh subruang dimensi yang relative lebih rendah. Pemikiran utama dari prinsip komponen untuk menentukan vektor yang paling cocok untuk distriusi citra wajah dalam keseluruhan ruang gambar vector.Vektor ini mendefinisikan

subruang dari citra wajah, yang kita sebut "face space". Masing-masing vektor tersebut adalah panjangn^2 , menjelaskanN × N gambar,dan merupakan kombinasi linear dari gambar wajah aslinya. Karena vektor ini adalah vektor eigen dari matriks kovarians sesuai dengan gambar wajah aslinya.

PCA memberikan transformasi ortogonal yang disebut juga dengan *eigenimage* yang di mana sebuah citra akan direpresentasikan ke dalam bentuk proyeksi linier searah dengan eigenimage yang bersesuaian dengan nilai Eigen terbesar dari matriks covariance. Dalam prakteknya, matriks *covariance* ini dibangun dari sekumpulan image training yang diambil dari berbagai objek.Sebuah citra 2D dengan dimensi kolom dan baris dapat direpresentasikan ke dalam bentuk citra 1D. Dalam penelitian ini ukuran jumlah kolom dan baris pixel citra adalah sama, sehingga nantinya akan terbentuk dimensi n^2.

Misalnya ada sejumlah N individu yang dijadikan sampel. Dari setiap individu diambil P citra, sehingga total citra di dalam *training set* adalah:

$$M = N \times P \tag{1}$$

Sejumlah Msampel citra dinyatakan sebagai $\{\Gamma_1, \Gamma_2, \Gamma_3, \ldots\ldots, \Gamma_M\}$di dalam sebuah ruang citra n^2 dimensi. Kumpulan citra tersebut dihitung nilai rata-ratanya yang disebut juga sebaga i*average face* dengan perhitungan berikut:

$$\psi = \frac{1}{M}\sum_{n=1}^{M} \Gamma_n \tag{2}$$

Setiap citra wajah dikurangi dengan nilairata-rata membentuk kumpulan vector menggunakan rumus:

$$\phi_i = \Gamma_i - \psi \tag{3}$$

Kemudian kumpulan vector yang sangat besar ini kemudian mengikuti pad aaturan PCA, vector tersebut mencarisejumlah M vektor-vektor ortonormal U_k dan nilai Eigen $\lambda_k$yang terbaik dalam menggambarkan distribusi dari data tersebut.

$$C = \frac{1}{M}\sum_{n=1}^{M} (U_k^T \Phi_n)^2 \tag{4}$$

Dimana nilai maksimumnya $\bigcup_l^T \bigcup_K = \delta_{IK} = \begin{cases} 1, \ if \ I = k \\ 0, otherwise \end{cases}$ (5)

Vektor-vektor $U_k$dan nilai-nilai λ_k adalah vektor-vektor Eigen dan nilai-nilai Eigen dari matriks *covariance*

$$C = \frac{1}{M}\sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T \quad (6)$$

Dimana matriks A = $[\Phi_1 \Phi_2 \dots \Phi_M]$

Matriks *covariance* C, dimana matriks $n^2 \ x \ n^2$adalah benar matriks simetris.Perihitungan *eigenvector*n^2 dan *eigenvalue* adalah untuk mengkonfersi ukuran images.Memerlukan komputasi yang layak untuk menemukan *eigenvector*. Matriks C diambil vektor-vektor *Eigen* terbaik sebanyak jumlah data. Karena vektor-vektor *Eigen* ini memiliki dimensi yang sama dengan dimensi citra yang asli,maka vektor-vektor ini jika disusun menjadi matrik sberukuran $n \times n$ akan membentuk citrayang mirip dengan wajah aslinya.

Eigenvector untuk v_i untuk $A^T A$ adalah

$$A^T A v_i = \mu_i v_i \quad (7)$$

Premultipying dari A adalah

$$AA^T A v_i = \mu_i A v_i \quad (8)$$

Kita melihat bahwa Av_i adalah *eigenvector* dan μ_i adalah *eigenvalue* dari C = AA^T . Setelah analisis ini kita membangun M x M matriks L = $A^T A$ dimana $L_{mn} = \Phi_m{}^T \Phi_n$dan menemukan M eigenvector, v_i , dari L

Vektor ini menentukan kombinasi linear dari M untuk membentuk *Eigenfaces*

$$U_l = \sum_{k=1}^{M} v_{lk} \Phi_k, \ l = 1, \dots, M \quad (9)$$

Dengan menganalisis penggunaaan PCA sangat mengurangi kalkulasi dari sejumlah piksel di dalam citra N^2 ke urutan jumlah gambar pada training M, training set gambar wajah akan relative kecil (M <<N^2) dan perhitungan menjadi sangat mudah. *Eigen* yang berhubungan memungkinkan kita untuk menentukan peringkat

eigen vector menurut kegunaannya dalam menggambarkan variasi antara gambar. Gambar *eigenface* dihitung dari vektor *eigen* dari L span dasar yang dapat digunakan untuk menggambarkan citra wajah. Komponen *eigenface* di proyeksikan dengan operasi sederhana.

$$w_k = U_k^T (\Gamma - \psi) \tag{10}$$

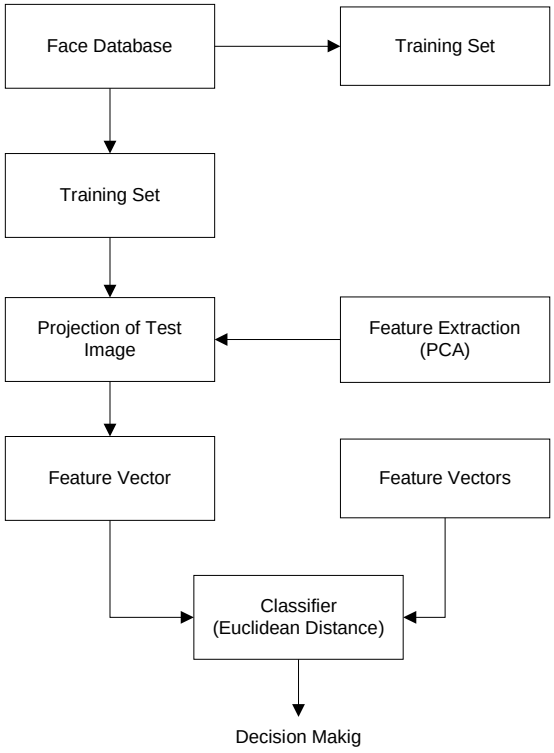
Untuk k = 1,…, M$'$ membentuk vektor proyeksi

$$\Omega^T = [w_1 w_2 \dots w_M] \tag{11}$$

Masing – masing *eigenface* dapat mewakili citra wajah.Vektor proyeksi ini kemudian digunakan dalam algoritma pengenalan wajah standar untuk mengidentifikasi dari sejumlah kelas wajah yang ditetapkan.Titik vektor wajah Ω_k dapat dihitung dengan rata – rata hasil representasi *eigenface* atas sejumlah kecilcitra wajah masing-masing individu. Klasifikasi dilakukan dengan membandingkan vektor proyeksi gambar wajah pelatihan dengan vektor proyeksi gambar input wajah. Perbandingan ini didasarkan pada jarak Euclidean antara kelas wajah dan citra input wajah. Hal ini digunakan untuk meminimalkan *Euclidean distance*.

$$\varepsilon_k = \|(\Omega - \Omega_k)\| \tag{12}$$

Dimana Ω_k adalah vector yang menggambarkan K^{th} wajah



Gambar 2.30 Pendekatan PCAuntuk pengenalan wajah (Eleyan & Demirel, 2007: 94-98).

Dengan kata lain metode PCAmemproyeksikan ruang asal menjadi ruang baru yang berdimensi lebih rendah, bahwa sebanyak mungkin informasi yang dimiliki dimensi asal dipertahankan dan tidak terlalu banyak yang hilang setelah diproyeksikan ke ruang yang baru. Dengan memperkecil dimensi dari sebuah ruang, tentu akan meringankan proseskomputasiyangperlu dilakukan oleh sistem.

2.1.16 *Euclidean Distance*

Euclidean distance adalah salah satu metode yang digunakan dalam pengenalan wajah.Metode ini digunakan untuk melakukan perhitungan jarak antara fitur yang ada pada wajah seperti mata, kuping, hidung serta untuk melakukan klasifikasi citra wajah yang baru dengan citra wajah yang telah diketahui. Nilai *Euclidean distance* bisa dilakukan dengan menggunakan rumus berikut:

$$\|x-y\|_e = \sqrt{|x_i - y_i|^2} \tag{13}$$

Dimana Untuk menentukan region titik $(3,2)$ digunakan rumus *Euclidian Distance* :

$$ED_i = \sqrt{|x_i - y_i|^2}$$
$$ED_1 = \sqrt{|3 - (2)|^2} \quad (14)$$
$$ED_1 = 2.23$$

2.1.17 *Embedded System For Access*

2.1.17.1 Mikrokontroler Arduino Uno

Arduino Uno adalah mikrokontroller ATmega328. Arduino uno ini memiliki 14 digital input / output pin (dimana 6 dapat digunakan sebagai output PWM), 6 input analog, resonator keramik 16 MHz, koneksi USB, jack listrik, header ICSP, dan tombol reset, semua ini berisi yang diperlukan untuk mendukung mikrokontroler. Arduino uno ini hanya terhubung ke komputer dengan kabel USB dengan adaptor AC-DC atau baterai untuk memulai.

Arduino uno berbeda dari semua mikrokontroller sebelumnya dalam hal itu tidak menggunkan FTDI chip driver USB-to-serial. Sebaliknya, fitur Atmega16U2 (Atmega8U2 sampai versi R2) diprogram sebagai konverter USB-to-serial. Arduino memiliki fitur-fitur baru sebagai berikut:

• 1.0 pinout dengan menambahkan SDA dan pin SCL yang dekat dengan pin AREF dan dua pin baru lainnya ditempatkan dekat dengan pin RESET, IOREF yang memungkinkan perisai untuk beradaptasi dengan tegangan yang disediakan dari papan. Perisai akan kompatibel dengan kedua papan yang menggunakan AVR, yang beroperasi dengan 5V dan dengan Arduino Karena yang beroperasi dengan 3.3V. Yang kedua adalah pin tidak terhubung, yang dicadangkan untuk tujuan masa depan.

• Rangkaian RESET yang kuat

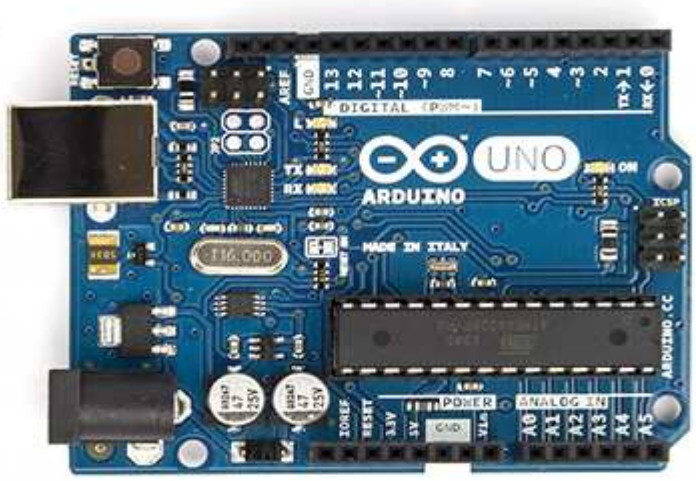
• Atmega 16U2 menggantikan 8U2.

"Uno" berarti satu di Italia dan diberi nama untuk menandai peluncuran Arduino 1.0. The Uno dan versi 1.0 akan menjadi versi referensi dari Arduino, bergerak maju. *The Uno* adalah yang terbaru dalam serangkaian papan Arduino USB, dan model referensi untuk platform Arduino untuk perbandingan dengan versi sebelumnya, lihat indeks Arduino boards

Berikut adalah spesifikasi dari Arduino Uno:

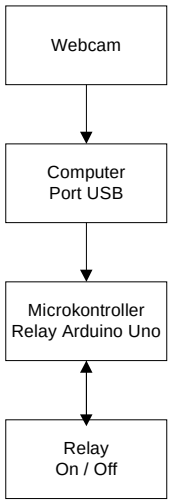
Microcontroller	ATmega328
Operating Voltage	5V

Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz



Gambar 2.31 Mikrokontroler Arduino Uno

2.1.17.2 Sistem Kendali Relay



Gambar 2.32 Sistem Kendali Relay

1. Webcam yang digunakan pada sebuah laptop yang menggunakan *OpenCV*untuk melakukan percobaan.
2. Komputer melakukan pengenalan wajah melalui webcam, setelah menangkap objek wajah serta memasukan database yaitu nama dan jenis kelamin selanjutnya mengambil training wajah sebanyak mungkin untuk mengidentifikasi data yang telah dimasukkan sebelumnya untuk menguji keakuratan data. Selanjutnya akan tampil wajah, nama dan jenis kelamin dalam training set akan tampil gambar wajah dalam format *pgm.
3. Mikrokontroller sebagai alat untuk mengirim data melalui PORT USB ke mikrokontroller untuk mengaktifkan relay.
4. Setelah data dikirm maka dilakukan simulasi terhadap relay.

2.1.18 *OpenCV*

OpenCV (Open Source Computer Vision) adalah library computer vision populer dimulai oleh Intel pada tahun 1999. Perpustakaan cross-platform menetapkan fokus pada pengolahan gambar *real-time* dan mencakup implementasi bebas paten algoritma visi komputer terbaru. Pada tahun 2008 Willow Garage mengambil alih dukungan dan *OpenCV* 2.3.1 kini hadir dengan antarmuka pemrograman untuk C, C + +, Python dan Android. OpenCV dirilis di bawah lisensi BSD sehingga digunakan dalam proyek-proyek akademik dan produk komersial sana. *OpenCV 2.4* kini hadir dengan kelas *FaceRecognizer* sangat baru untuk

pengenalan wajah, sehingga Anda dapat mulai bereksperimen dengan pengenalan wajah segera. *OpenCV* inimenunjukkan bagaimanamelakukan pengenalan wajah dengan *FaceRecognizer* di *OpenCV* dan memberikan sebuah pengantar ke dalam algoritma di belakang.

OpenCV.org (2012) menyatakan bahwa *OpenCV* (*Open Source Computer Vision Library*) adalah program sumber terbuka *computer vision* dan *machine learning*. *OpenCV* dibuat untuk menyediakan infrastruktur umum untuk aplikasi *computer vision* dan mempercepat penggunaan persepsi mesin di produk komersial. Menjadi produk berlisensi BSD, *OpenCV* mempermudah perusahaan untuk memanfaatkan dan memodifikasi kode.

Library OpenCV telah memiliki lebih dari 2500 algoritma yang sudah teroptimasi, yang terdiri dari sekumpulan algoritma klasik dan algoritma *computer vision*. Algoritma ini bisa digunakan untuk mendeteksi dan mengenal wajah, mengidentifikasi objek, mengklasifikasi tindakan manusia dalam video, melacak pergerakan kamera, melacak pergerakan objek, mengekstraksi model 3D dari objek, memproduksi poin 3D dari *stereo camera*, menggabungkan gambar untuk memproduksi gambar yang memiliki resolusi tinggi, mencari gambar yang sama dari database gambar, menghilangkan mata merah dari gambar yang diambil menggunakan *flash*, mengikuti pergerakan mata, mengenal pemandangan dan membangun penanda untuk *augmented reality*, dll.

2.1.19 *Visual Basic NET (VB. NET)*

Visual Basic NET (VB.NET) adalah bahasa pemrograman komputer berorientasi objek diimplementasikan pada NET *Framework*. Meskipun merupakan evolusi dari bahasa *Visual Basic klasik*, tidak mundur-kompatibel dengan VB6, dan setiap kode yang ditulis dalam versi lama tidak dapat dikompilasi di bawah *VB.NET*.

Seperti semua bahasa NET lainnya, *VB.NET* memiliki dukungan lengkap untuk konsep berorientasi objek.Segala sesuatu di VB.NET adalah obyek, termasuk semua tipe primitif (*Short, Integer, Long, String, Boolean, dll)* dan tipe *user-defined*, peristiwa, dan bahkan rakitan.Semua benda mewarisi dari kelas dasar *Object*.

VB.NET dilaksanakan oleh *NET framework Microsoft*. Oleh karena itu, ia memiliki akses penuh ke semua perpustakaan di *.NET Framework*. Ada juga kemungkinan untuk menjalankan program *VB.NET*, alternatif open source untuk *NET*, tidak hanya di bawah *Windows*, tapi bahkan *Linux* atau *D*

Alasan berikut membuat *VB.NET* bahasa profesional banyak digunakan:

1. Modern, tujuan umum.
2. Obyek berorientasi.
3. Komponen berorientasi.
4. Mudah untuk belajar.
5. Bahasa terstruktur.
6. Ini menghasilkan program yang efisien.
7. Hal ini dapat dikompilasi pada berbagai platform komputer.
8. Bagian dari .NET Framework.

(http://www.tutorialspoint.com/vb.net/vb.net_overview.htm)

2.2 *Related Works*

Dari penelitian sebelumnya A.D. Meenakshi (2013).*"Face Recognation Using ICA For Biometric Security System"*. *Independent Component Analysis* (ICA) merupakan generalisasi dari PCA. ICA menggunakan database gambar untuk merepresentasikan wajah seseorang .Hasil penelitian menunjukkan bahwa classifier ICA memberikan kinerja yang baik untuk pengenalan wajah.Kelemahan dari metode ini adalah ICA digunakan untuk mmaksimalkan informasi transmisidalam,sehingga lebihkuat untukvariasi seperti kondisi pencahayaan, rambut, makeup, dan ekspresi wajah. Dalam penelitian ini variasi PCA ditemukan lebih baik dalam hal variasi sudut dan untuk proses ulang.

Dari penelitian sebelumnya Paul, Ruby, et al (2014). *"An Efficient Face Recognition Using DTC, Adaptive LBP And Gabor Filter With Single Sample Perclass"*, Di sini pertama fitur wajah yang diekstraksi menggunakan discretecosine transformasi. Kemudian fitur diekstrak dengan menggunakan filter Gabor dan diubah menjadi Template wajah biner. Dan template wajah biner bertindak seperti masker untuk mengekstrak fitur tekstur lokal dengan menggunakan Adaptive Binary Pattern lokal. Berikut basis data FERET digunakan untuk menyimpan gambar untuk pelatihan dan pengujian. Di sini dalam metode menggunakan sampel tunggal untuk setiap individu maka kurang maksimal terhadap ruang dan waktu pengujian sampel. Maka diharapkan menerapkan pose wajah dengan menggunakan metode PCA untuk meningkatkan kinerja menggunakan single sampe per orang.

Dari penelitian sebelumnya, Zainudin et al (2012). *"Face Recognition using Principle Component Analysis (PCA) and Linear Discriminant Analysis (LDA)"*,

Teknik untuk mengurangi dimensi data sebelum pengklasifikasian menggunakan algoritma *Fisher Faces*. *Linear Dyscriminant Analysis* (LDA) digunakan untuk mengurangi jumlah fitur agar lebih mudah dikelola sebelum proses klasifikasi. Dalam proyek ini dua jenis ekstraksi fitur untuk algoritma pengenalan wajah, PCA dan LDA dipelajari. PCA dan LDA dilaksanakan dengan menggunakan MATLAB dan kinerja ditentukan dalam hal akurasi pengenalan dan waktu eksekusi yang diambil. Percobaan dilakukan dalam kondisi yang berbeda, dengan memvariasikan dataset citra input wajah dan juga dengan memvariasikan parameter algoritma individu. Hasil penelitian menunjukkan bahwa LDA melakukan lebih baik dari PCA dalam hal akurasi pengenalan. Untuk peningkatan, metode untuk menemukan dan mengidentifikasi wajah akurat . Dengan mengontrol orang ekspresi wajah, serta jarak dari kamera, sudut kamera, pencahayaan, gambar yang ditimbulkan dapat meminimalkan jumlah variabel dalam foto. Ini memungkinkan perangkat lunak pengenalan wajah untuk beroperasi di bawah kondisi ideal dekat sangat meningkatkan akurasinya.

Dari penelitian sebelumnya, Bayu, Hendriawan, dan Susetyoko (2009) "*Penerapan Face Recognition Dengan Metode Eigenface Dalam Intelligent Home Security*".memecahkan masalah untuk melakukan percobaan dan pengujian alat yang dapat mengenali citra wajah dengan tingkat keberhasilan mencapai 87%. Pada permasalahan ini menggunakan metode *eigenface* dalam *intelligent home security*. Sesuai permasalahan, alat berhasil dapatmelakukan pengenalan meskipun posisinya berdeda beda. Karena yang digunakan adalah nilai dari *eigenface* tiap citra wajah yang dibandingkan. Tingkat keberhasilan pengenalan wajah menggunakan *euclidean distance dan k- Nearest neighbour* sangat dipengaruhi oleh deteksi wajah,pemrosesan awal, dan penghitungan dengan PCA(*eigenface*) sebelumnya. Pengguanaan *class* untuk pengelompokandata pemilik rumah dan bukan pemilik rumah sangat efektif digunakan dalam proses verifikasi antara pemilik atau pencuri.