BAB 2
LANDASAN TEORI

2.1. Computer Vision

Computer Vision merupakan bidang yang berkembang pesat dikhususkan untuk menganalisa, memodifikasi, dan pemahaman gambar tingkat tinggi. Tujuannya adalah untuk menentukan apa yang terjadi di depan kamera dan menggunakan pemahaman tersebut untuk mengendalikan komputer, sistem robot, atau untuk memberikan citra baru yang lebih informatif dari pada gambar kamera asli. Aplikasi area untuk komputer visi teknologi termasuk *video surveillance*, biometrik, otomotif, fotografi, produksi film, pencarian web, obat-obatan, *game augmented reality*, antarmuka pengguna yang baru, dan banyak lagi [4].

Dengan demikian dari hasil proses tersebut akan mendapatkan keputusan-keputusan berupa peringatan maupun tanda dari hasil proses tersebut. Adapun perubahan kedalam presentasi yang baru seperti perubahan gambar menjadi grayscale atau juga pemotongan objek pada gambar. Berbeda halnya dengan manusia yang memiliki pemikiran untuk meneliti, memahami, dan membandingkan informasi pada objek secara langsung dengan informasi dari pengalaman-pengalaman yang didapatkan selama bertahun-tahun hidup didunia. Dalam sistem penglihatan mesin (*machine vision*) komputer hanya mampu mendapatkan informasi berupa kumpulan angka-angka dari media input data seperti kamera [4]. Berikut contoh gambar ilustrasi mengenai proses *computer vision*.



Gambar 2. 1 Gambar Sebelum dan Sesudah Deteksi

Dengan terlihat dari gambar tersebut hanya terlihat wajah wanita bernama lena dengan menggunakan topi berbulu namun penglihatan sebuah komputer berbeda dengan manusia, komputer hanya melihat gambar dengan jenis 2D tanpa mengetahui yang mana mata maupun hidung dan komputer hanya mendeteksi angka-angka yang dapat dipahami dari setiap pixel yang ada pada gambar. Sehingga informasi yang disampaikan masih kurang akurat. Dengan demikian permasalahan tersebut masih diteliti untuk menemukan cara penelesaiannya. [4]

2.2. Deteksi Objek

Deteksi objek atau pendekatan objek merupakan suatu teknik komputer yang berkaitan dengan *computer vision* dan *image processing* yang berhubungan dengan mendeteksi suatu objek. Metode yang digunakan dalam penerapan ini yaitu *viola jones*.

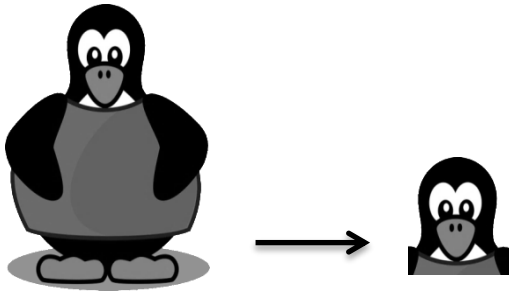
2.3. Sistem Pendeteksian Pelanggaran

Sistem pendeteksian pelanggaran lampu merah merupakan tujuan untuk melakukan seleksi pada area lampu merah dalam mendeteksi sebuah pelanggaran dari sebuah citra *video* kedalam sebuah sistem. Proses ini melakukan pemeriksaan citra dari sebuah gambar masukan apakah terdapat objek sebuah kendaraan yang melanggar maupun tidak melanggar dalam sistem tersebut dan apabila ditemukan terdeteksi pelanggaran maka akan muncul sebuah peringatan pada sistem tersebut.

Pada pendeteksian terebut terdapat beberapa tahapan pengekstraksian citra untuk dapat mengolah sistem tersebut, selain itu juga masukan citra berupa *video* maka proses yang dilakukan berupa *vehicle tracking*. [5]

2.3.1. Cropping

Cropping merupakan pengolahan dengan kegiatan memotong bagian dari gambar untuk mendapatkan bagian yang tertentu. [4] Berikut adalah gambar proses cropping, jika ingin crop hanya bagian kepala. Dapat dilihat pada gambar 2.2



Gambar 2. 2 Cropping

2.3.2. Grayscaling

Grayscale adalah warna-warna piksel yang berada dalam rentang gradasi warna hitam dan putih. Intensitas grayscale biasanya disimpan sebagai data citra 8 bit atau 256 intensitas warna gray dari nilai 0 (hitam) dan 255 (putih) [6].

$$Grayscale = 0.299\ R + 0.587\ G + 0.114\ B \qquad [2.1]$$



Gambar 2. 3 Proses Grayscale Sebelum dan Sesudah

2.3.3. Thresholding

Threshold merupakan tahap untuk membuat warna citra menjadi hanya berwarna hitam atau putih. Untuk melakukan thresholding terlebih dahulu harus dicari suatu nilai yang digunakan untuk batas toleransi suatu warna keabuan dapat dijadikan warna hitam atau warna putih, nilai tersebut biasa disebut nilai ambang. Setelah semua piksel diubah menjadi nilai keabuan, pencarian nilai ambangpun dilakukan. Ada beberapa cara menentukan nilai ambang yaitu pertama, dengan menentukan langsung berdasarkan nilai tengah dari level keabuan yaitu 128. Kedua, menentukan suatu nilai tertentu sebagai nilai ambang. Ketiga, menentukan nilai rata – rata dari total piksel keabuan dalam citra dan menjadikannya sebagai nilai ambang rata – rata. Kemudian setelah didapat nilai ambang tersebut maka akan dilakukan perbandingan setiap nilai piksel dari citra keabuan tersebut dengan nilai ambangnya. Jika nilai piksel lebih besar dari nilai ambangnya maka nilai piksel tersebut akan diubah menjadi 0 (piksel berwarna hitam). Sedangkan jika yang ditemukan sebaliknya yaitu nilai pikselnya lebih kecil dari nilai ambang yang telah ditentukan maka nilai piksel tersebut diubah menjadi 1 (piksel berwarna hitam) [7]

$$f_0(x,y) = \begin{cases} T1, & f_1(x,y) \le T1 \\ T2, & T1 < f_1(x,y) \le T2 \\ T3, & T2 < f_1(x,y) \le T3 \\ . & \\ . & \\ Tn, & Tn\text{-}1 < f_1(x,y) \le Tn \end{cases}$$

Gambar 2. 4 Rumus Threshold

Thresholding atau bisa disebut dengan pengambang-batasan yaitu sebuah teknik untuk mengubah suatu citra abu-abu menjadi citra biner (putih dan hitam). Pengembang-batasan terdapat dua jenis [7]:

1. Pengambang-batasan Tunggal

Suatu citra abu-abu diubah menjadi citra biner (putih dan hitam) dengan cara memilih suatu level keabuan T dalam citra asli, dan kemudian mengubah setiap piksel hitam atau putih tergantung nilai keabuan lebih besar atau kurang dari T. Suatu piksel menjadi:

$$g(x,y) = \begin{cases} 1 & if \ \ f(x,y) > T \\ 0 & if \ \ f(x,y) \leq T \end{cases} \qquad [2.2]$$

- Putih Jika level keabuannya >T
- Hitam jika level keabuannya <= T

Pengambang-batasan merupakan hal yang vital dalam segmentasi citra digital, dimana diinginkan untuk mengisolasi objek-objek dari latar belakang citra. Hal ini juga merupakan bidang yang penting dalam robotika. Pengambang batasasan dapat dilakukan secara sederhana dalam Matlab. Jika diasumsikan dimiliki suatu citra 8 bit dan disimpan dalam suatu variabel X, maka perintahnya X>T.

Citra yang dihasilkan kemudian dapat diproses untuk mencari jumlah bintik atau ukuran rata-rata bintik. Untuk melihat hal ini bekerja, ingat bahwa dalam Matlab, suatu operasi pada suatu angka, ketika diterapkan pada suatu matriks, diinpretasikan sebagai penerapan terhadap semua elemen-elemen matriks. Perintah X>T akan memberikan nilai balik 1 (true) untuk semua piksel yang memiliki intensitas keabuan yang lebih besar dari T, dan nilai balik 0 (false) untuk semua piksel yang memiliki intensitas keabuan yang lebih kecil atau sama dengan T.

Selain metode diatas, Matlab juga memiliki fungsi im2bw, yang mengambangbatas suatu citra dengan sembarang tipe data. Karena level merupakan parameter bernilai antara 0 sampai 1, yang mengindikasikan fraksi nilai-nilai keabuan yang diubah menjadi putih. Perintah ini bisa dipakai untuk citra abu-abu, citra berwarna maupun untuk citra indeks dengan tipe data uint8, uint16, ataupun double. Fungsi im2bw secara otomatis menskalakan nilai level menjadi suatu nilai keabuan yang cocok dengan tipe citra, dan kemudian melakukan pengambang-batasan seperti yang telah dijelaskan sebelumnya. Maka persamaan yang digunakan:

2. Pengambang-batasan Ganda

Disini dipilih dua nilai T1 dan T2 dan diterapkan suatu operasi pengambang-batasan sebagai berikut. Suatu piksel menjadi:

- Putih Jika level keabuan antara T1 dan T2
- Hitam jika level diluar rentang ini

Operasi ini bisa diimplementasikan dengan suatu variasi sederhana dari pengambang-batasan tunggal X>T1 & X<T2. Karena simbol ampersand berperan sebagai operasi logikal, hasilnya akan menghasilkan nilai 1 jika memenuhi kedua pertidak-samaan itu. Berikut contoh gambar hasil threshold:



Gambar 2. 5 Thresholding

2.3.4. Viola Jones

Metode *Viola Jones* adalah metode yang memiliki kapabilitas untuk mendeteksi objek pada suatu citra gambar dengan tingkat keakuratan yang cukup tinggi. Pada standarnya pengolahan citra pada gambar akan dilakukan dengan merubah skala ukuran gambar kedalam ukuran yang ditentukan dan melakukan pendeteksian dengan skala tetap pada gambar tersebut. Dengan hal tersebut simpulkan bahwa waktu untuk pendeteksian akan lebih cepat dibandingkan dengan melakukan pendeteksian terhadap citra gambar yang berukuran berbedabeda [2].

Berbeda halnya dengan Metode *Viola Jones* yang terus merubah ukuran skala pendeteksian dibandingkan harus merubah ukuran dari skala gambar dan menjalankan banyak pendeteksian terhadap gambar dengan ukuran skala pendektesian yang berbeda tiap waktunya. Hal ini diperkirakan akan membutuhkan lebih banyak waktu, tapi Metode *Viola Jones* memiliki fitur pendeteksian yang

mengkonstruksi gambar kedalam gambar integral dan beberapa fitur persegi yang simple yang disebut *Haar wavelets*. Sehingga hasil kalkulasi dalam skala pendeteksian akan sama berapapun ukuran dari gambar yang diproses.

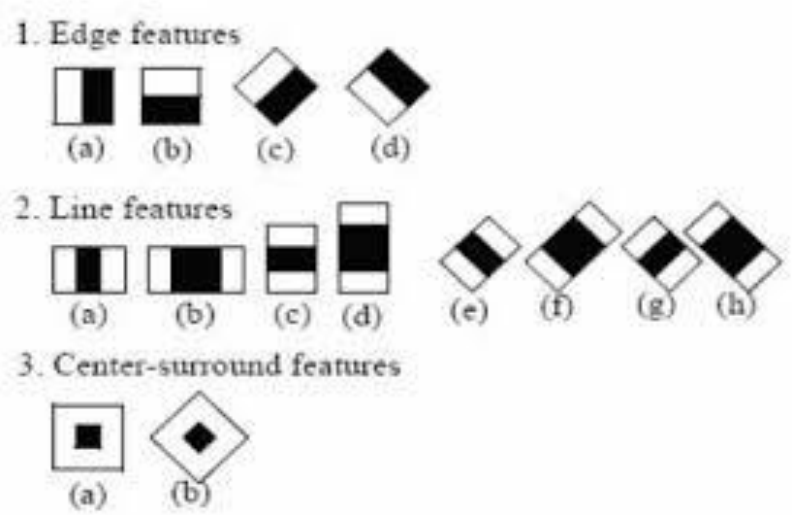
2.3.4.1. Haar Classifier

Haar classifier merupakan pengklasifikasian fitur yang digunakan dalam metode *viola jones*. *Haar classifier* merupakan suatu metode yang membangun sebuah *boosted rejection cascade*, yang membuang data *training negative* sehingga dapat memutuskan data *training positive. Haar classifier* merupakan *supervised learning* yaitu membutuhkan data *training* untuk dapat mendeteksi objek-objek tertentu. Untuk itu *haar classifier* membutuhkan data positif dan data negative, dimana data positif merupakan data yang berisikan objek yang dideteksi, sedangkan data negative merupakan data yang diisikan bukan objek yang didetksi. [4]

2.3.4.2. Haar Fitur

Fitur merupakan tahap pertama dalam pendeteksian *viola jones*. penggunaan fitur karena penggunaannya lebih cepat dibanding dengan perhitungan *per-pixel*. Berdasarkan penelitian yang dilakukan arhutomo (2010) fitur yang digunakan oleh *viola jones* yaitu mengguanakan *wavelet haar. Wavelet haar* adalah gelombang tunggal bujur sangkar yang mempunyai satu interval tinggi dan satu interval rendah. Dan kemudian dikembangkan menjadi fitur pendeteksian objek atau yang disebut *haar* atau *haarlike.* [2]

Kemudian hasil nilai tersebut akan menghasilkan fitur *sub-window* yang memiliki dua atau lebih kombinasi kotak hitam dan putih yang akan digunakan untuk tahap pendetaksian selanjutnya. Berikut contoh gambar untuk tipe-tipe dari kombinasi kotak.



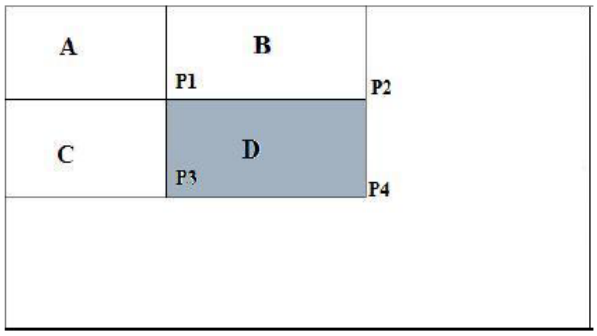
Gambar 2. 6 Haar Fitur

Selanjutnya, setelah nilai fitur didapatkan *Viola-Jones* menggunakan metode *AdaBoost machine-learning* untuk mengetahui suatu fitur apakah merepresentasikan ada tidaknya wajah dalam suatu citra masukan. Metode ini menggabungkan banyak *classifier* lemah menjadi satu *classifier* kuat. *Classifier* adalah informasi ciri yang menandakan adanya suatu objek pada suatu citra misalnya ciri pada wajah. Sedangkan *classifier* lemah adalah suatu jawaban benar namun memiliki tingkat kebenaran yang kurang akurat, maka jika digabungkan classifier lemah tersebut akan menghasilkan suatu *classifier* kuat.[2]

2.3.4.3. Integral Image

Setelah melalu proses fitur lalu gambar dihitung cintra integralnya dari pendeteksian ini yaitu dengan cara mengubah contoh gambar yang berdimensi 5x5 kedalam gambar integral. Yaitu dengan cara menjumlahkan tiap pixel dengan seluruh pixel yang terdapat di daerah sebelah kiri dan atas pixel tersebut. Dengan demikian, selanjutnya perhitungan dapat dilakukan dengan cara menjumlahkan 4 nilai pixel kedalam pixel yang telah dipilih.

Rumus integral image : $P(\mathrm{x},\mathrm{y}) = \sum_{x \leq x, y \leq y} i(x,y)$ [2.3]



Gambar 2. 7 Integral Image

Sebagai contoh, piksel (a,b) memiliki nilai akumulatif untuk semua piksel (x,y) dimana x<=a dan y <= b. [5] Salah satu contoh seperti perhitungan berikut:

F(x) = D

D = (A+B+C+D)

F(x) = A+(A+B+C+D) – (A+C+A+B)



Gambar 2. 8 Contoh Perhitungan Integral Image

$$s(x,y)=s(x,y)+s(x-1,y)+s(x,y-1)-s(x-1,y-1) \quad [2.4]$$

1	2	3
2	3	4
4	5	1

1	2	3
2	3	4
4	5	1

1	1+2	1+2+3
1+2	1+2+2+3	1+2+3+2+3+4
1+2+4	1+2+3+4+5	1+2+3+2+3+4+4+5

Maka hasil perhitungannya:

1	3	6
3	8	15
7	17	25

f(x) =Sumblack rectangle (pixel gray level) – Sumwhite rectangle (pixel gray level)

= |(6+0-(0+0))+(25+3-(15+0))-(15+1-(6+0))|

= |6+13-10|

= 9

2.3.4.4. Adaptive Boosting atau AddBoost

AdaBoost merupakan tahap ketiga dalam metode *Viola Jones*. Algoritma *AdaBoost* berfungsi untuk melakukan pemilihan fitur-fitur dalam jumlah banyak dengan hanya memilih fitur-fitur tertentu. Dalam prakteknya tidak satupun yang mampu melakukan pengklasifikasian dengan *error* yang kecil. Algoritma *AdaBoost* berfungsi mencari fitur-fitur yang memiliki tingkat pembeda yang tinggi. Hal ini dilakukan dengan mengevaluasi setiap fitur terhadap terbesar antara wajah dan non wajah dianggap sebagai fitur terbaik.

Sebelum masuk *addaboost* dilakukan normalisasi pada gambar yang akan dilatih, menggunakan persamaan berikut:

$$\sigma^2 = m^2 - \frac{1}{n}\sum x^2 \qquad [2.5]$$

Di mana σ adalah standard deviasi, *m* adalah *mean*, dan x adalah nilai pixel pada gambar. Feature yang diekstrak oleh algoritma adaboost adalah haar feature. Nilai haar feature adalah perbedaan antara jumlah piksel dalam daerah hitam dan daerah putih sehingga dapat mencerminkan perubahan skala pada gambar grayscale. Haar feature diekstrak menggunakan integral image. Kemudian algoritma adaboost akan melakukan seleksi terhadap feature dan melakukan pembobotan untuk membentuk classifier lemah. Gabungan dari classifier lemah membentuk classifier kuat. Pada Gambar 4 ditunjukkan skema detail algoritma adaboost. Haar feature dihitung melalui integral image. Nilai pada integral image pada setiap titik (x,y) dapat dinyatakan sebagai jumlah titik piksel atas kiri saat ini dan dihitung melalui persamaan berikut:

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y') \qquad [2.6]$$

Dimana ii(x,y) adalah integral image yang telah dihitung dan i(x',y') adalah gambar grayscale original. Varian Algoritma adaboost yang akan digunakan adalah gentle adaboost. Adapun langkah demi langkah algoritma gentle adaboost adalah sebagai berikut:

1. Dimulai dengan inisialisasi bobot Wi = 1/N,I = 1,2,…,N, F(x) = 0
2. Ulangi untuk m = 1,2,…,M:
    a. Ulangi untuk m = 1,2,…,M:Neg = menset nilai sample negative
    b. Sesuaikan dengan fungsi regresi $f_m(x)$ dengan pembobotan least-squares dari y_i ke x_i dengan bobot w_i
    c. Update $F(x) \leftarrow F(x) + f_m(x)$
    d. Update $w_i(x) \leftarrow w, \exp(-y_i f_m(x))$ dan normalisasi ulang
3. Output classifier

    Sign [F(x)] = *sign* $\left[\sum_{m=1}^{M} f_m\right]$

Proses berikutnya adalah membentuk cascade detector. Tahapan dalam cascade ini dibangun oleh pelatihan classifier menggunakan adaboost dan kemudian menyesuaikan nilai ambang batas untuk meminimalkan negatif palsu. Pada Gambar 5 ditunjukkan cacade detector. Algoritma pelatihan untuk membangun cascade detector adalah sebagai berikut:

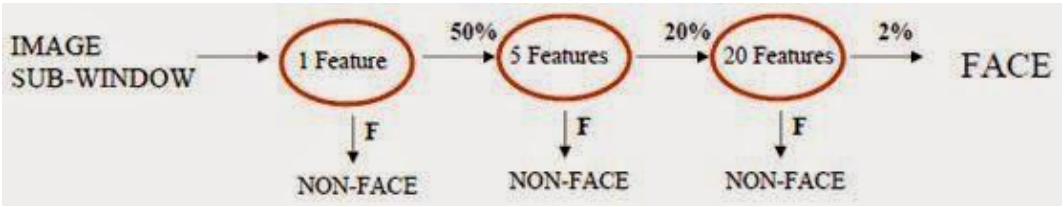
1. Pengguna memilih nilai untuk f, yaitu tingkat maksimum *false positive* dapat diterima per tahap dan d, yaitu tingkat deteksi minimum yang dapat diterima per tahap.
2. Pengguna memilih target keseluruhan dari tingkat *false positive* F_{target}
3. Pos = menset nilai sample positif
4. Neg = menset nilai sample negative
5. $F_0 = 1{,}0, D_0 = 1.0, i = 0$
6. $i = 0$
7. While Fi > F_{target}
    a. $i \leftarrow i + 1$
    b. $n_i = 0;\ F_i = F_i - 1$
    c. While F_i > f x $F_i - 1$
        i. $n_i \leftarrow n_i + 1$
        ii. Gunakan Pos dan Neg untuk melatih *classifier* dengan fitur n_i menggunakan *adaboost*
        iii. Lakukan evaluasi terhadap *classifier cascade* saat ini untuk menentukan F_i,D_i
        iv. Menurunkan threshold untuk classifier ke *i* sampai classifier saat ini memiliki tingkat deteksi setidaknya d X D_{i-1}
    d. $N \leftarrow \varphi$
    e. If $F_i + 1 > F_{target}$ maka lakukan evaluasi pada cascade detector saat ini menjadi objek dan menempatkan setiap pendeteksian palsu ke dalam Neg

Boosting merupakan meta-algoritma dalam *machine learning* untuk melakukan *supervised learning*. *boosting* merupakan sekumpulan *weak learner*

menciptakan satu kesatuan *strong learner*. *weak learner* adalah *classifier* yang hanya memiliki sedikit korelasi dengan klasifikasi yang sebenarnya, sementara *strong learner* adalah *classifier* yang memiliki kolerasi kuat dengan klasifikasi yang sebenarnya.

2.3.4.5. Kombinasi Cascade of Classifier

Kombinasi *Cascade of Classifier* merupakan tahap terakhir dari *viola jones*. dengan mengkombinasikan pengklasifikasian dalam struktur *cascade* atau *Cascade of Classifier* kecepatan pengklasifikasian dapat terukur dan meningkat yaitu dengan cara mempusatkan pada area pendeteksian yang dituju saja atau dengan memusatkan perhatian pada daerah yang berpeluang saja. Hal ini dilakukan untuk mengetahui tata letak suatu *image*. Dibawah ini terdapat alur kerja klasifikasi bertingkat. [4]



Gambar 2. 9 Alur Klasifikasi

2.4. Menentukan Parameter Akurasi

Untuk menentukan parameter fungsi untuk mengoptimalkan tahapan yang digunakan, *false positive rate, true positive rate,* dan jenis fitur yang digunakan untuk training. Ada beberapa parameter yang perlu diperhatikan saat melakukan training dengan klasifikasi bertingkat.[9] Parameter tersebut yaitu:

2.4.1. Object Training Size

Ukuran objek latih harus ditentukan berdasarkan nilai vektor 2-elemen [tinggi, lebar] atau dengan menggunakan format '*Auto*'. Sebelum proses training, fungsi akan mengubah ukuran sampel positif dan negatif dalam piksel. Jika menggunakan 'Auto' maka fungsi menentukan ukuran secara otomatis berdasarkan rasio tinggi-lebar median dari contoh positif. Untuk menghasilkan keakuratan sistem deteksi yang optimal sebaiknya ukuran objek data latih mendekati ukuran

yang diharapkan dari objek dalam gambar. Namun untuk mempercepat training dan pendeteksian maka setting ukuran objek training menjadi lebih kecil dari ukuran objek dalam gambar.

2.4.2. Negative Sample Factor

Merupakan penentu berapa banyak jumlah sampel negatif yang akan digunakan pada setiap tahap klasifikasi bertingkat.

2.4.3. Num Cascade Stage

Merupakan jumlah tahap klasifikasi bertingkat dalam training yang dinyatakan dalam bilangan bulat positif. Meningkatkan jumlah tahap dapat menghasilkan detektor yang lebih akurat tetapi juga akan meningkatkan lamanya waktu training. Tahap yang lebih banyak akan memerlukan gambar latih yang lebih juga karena pada setiap tahap beberapa jumlah sampel positif dan negatif kemungkinan akan dihilangkan. Hal tersebut tergantung pada pengaturan false alarm rate dan true positive rate.

2.4.4. False Alarm Rate

Adalah tingkat alarm salah yang dapat diterima pada setiap tahap klasifikasi dengan nilai dalam kisaran [0 1]. False alarm rate merupakan sebagian kecil dari sampel training negatif yang salah diklasifikasikan sebagai sampel positif. Nilai pengaturan yang lebih rendah pada false alarm rate akan meningkatkan kompleksitas dari setiap tahap yang artinya kemampuan untuk mendeteksi objek salahnya rendah dapat mengakibatkan waktu training dan pendeteksian yang lebih cepat. Sedangkan jika nilai pengaturan lebih tinggi maka akan memerlukan lebih banyak tahapan klasifikasi agar dapat mencapai keakuratan dalam deteksi.

2.4.5. True Positive Rate

Tingkat positif benar adalah fraksi yang mengklasifikasi kebenaran dari sampel positif dalam proses training. Nilai true positive rate yang lebih tinggi akan meningkatkan kompleksitas dari setiap tahap. Peningkatan kompleksitas dapat menghasilkan sejumlah besar deteksi objek yang tepat tetapi akan menyebabkan lamanya waktu training dan deteksi.

2.4.6. Feature type

Ada beberapa jenis fitur yang disediakan dalam train cascade object detector dan dapat digunakan untuk mengenali objek antara lain: *Haar-like features, Local Binary Patterns, dan Histogram of Oriented Gradients*. Fitur yang digunakan yaitu Haar-like features berdasarkan konsep dari metode Viola-Jones yang digunakan dalam Tugas Akhir ini

2.5. Pemrograman Berorientasi Objek

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan suatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur data dab perilaku dalam suatu entitas. Model berorientasi objek bermanfaat untuk memahami masalah, komunikasi dengan ahli aplikasi, pemodelan suatu organisasi, meyiapkan dokumentasi serta perancangan program dan basis data. Pertama-tama suatu model analisis dibuat untuk menggambarkan aspek dasar dari domain aplikasi, dimana model tersebut berisi objek yang terdapat dalam domain aplikasi termasuk deskripsi dari keterangan objek dan perilakunya.

Secara spesifik, pengertian berorientai objek berarti bahwa mengorganisasi perangkat lunak sebagai kumpulan dari objek tertentu yang memiliki struktur data dan perilakunya. Hal ini yang membedakan dengan pemograman konvensional dimana struktur data dan perilaku hanya berhubungan secara terpisah. Terdapat beberapa cara untuk menentukan karateristik dalam pendekatan berorientasi objek, tetapi secara umum mencakup empat hal, yaitu identifikasi, klasifikasi, polymorphism (polimorfisme) dan inheritance (pewarisan).[11]

Metodologi pengembangan sistem berorientasi objek mempunyai tiga karateristik utama, yaitu:

a. Encapsulation

Encapsulation (pengkapsulan) merupakan dasar untuk pembatasan ruang lingkup program terhadap data yang diproses. Data dan prosedur atau fungsi dikemas dalam bersama-sama dalam suatu objek, sehingga prosedur atau fungsi

lain dari luar tidak dapat mengaksesnya. Data terlindung dari prosedur atau objek lain kecuali prosedur yang berada dalam objek itu sendiri.

b. Inheritance

Inheritance (pewarisan) adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi atribut dan metoda dari induknya langsung. Atribut dan metoda dari objek induk diturunkan kepada anak objek, demikian seterusnya. Pendefinisian objek dipergunakan untuk membangun suatu hirarki dari objek turunannya, sehingga tidak perlu membuat atribut dan metoda lagi pada anaknya, karena telah mewarisi sifat induknya.

c. Polymorphism

Polymorphism (polimorfisme) yaitu konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku berbeda. Polimorfisme mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda.

2.6. UML

Pada perkembangan teknologi perangkat lunak, diperlukan adanya bahasa yang digunakan untuk memodelkan perangkat lunak yang akan dibuat dan perlu adanya standarisasi agar orang di berbagai negara dapat mengerti pemodelan perangkat lunak. Seperti yang diketahui bahwa menyatukan banyak kepala untuk menceritakan sebuah ide dengan tujuan untuk memahami, merupakan hal yang tidak mudah, oleh karena itu diperlukan sebuah bahasa pemodelan perangkat lunak yang dapat dimengerti oleh banyak orang. Banyak orang yang telah membuat bahasa pemodelan pembangunan perangkat lunak sesuai dengan teknologi pemrograman yang berkembang pada saat itu, misalnya yang sempat berkembang dan digunakan banyak pihak adalah *Data Flow Diagram* (DFD) untuk memodelkan perangkat lunak yang menggunakan pemrograman prosedural atau struktural, kemudian juga ada *State Transition Diagram* (STD) yang digunakan untuk memodelkan sistem *real time* (waktu nyata).

Pada perkembangan teknik pemrograman berorientasi objek, munculah sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang

dibangun dengan menggunakan teknik pemrograman berorientasi objek, yaitu *Unified Modeling Language* (UML). UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

Seperti yang diketahui di dunia sistem informasi yang tidak dapat dibakukan, semua tergantung kebutuhan, lingkungan dan konteksnya. Begitu juga dengan perkembangan penggunaan UML bergantung pada level abstraksi penggunaannya. Jadi, belum tentu pandangan yang berbeda dalam penggunaan UML adalah suatu yang salah, tapi perlu ditelaah dimanakah UML digunakan dan hal apa yang ingin digambarkan. Secara analogi jika dengan bahasa yang digunakan sehari-hari, belum tentu penyampaian bahasa dengan puisi adalahhal yang salah. Sistem informasi bukanlah ilmu pasti, maka jika ada banyak perbedaan dan interpretasi di dalam bidang sistem informasi merupakan hal yang sangat wajar.

Berikut ini penjelasan singkat dari pembagian kategori uml:

1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antarsubsistem pada suatu sistem.[7]

2.7. Use Case Diagram

Use Case atau diagram *Use Case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. Use Case mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, Use Case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsifungsi itu.[10]

Syarat penamaan pada *Use Case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *Use Case* yaitu pendefinisian apa yang disebut aktor dan *Use Case.*

2.7.1. Aktor

Aktor adalah sesuatu (entitas) yang berhubungan dengan sistem dan berpartisipasi dalam use case. Aktor menggambarkan orang, sistem atau entitas eksternal yang secara khusus membangkitkan sistem dengan input atau masukan kejadian-kejadian, atau menerima sesuatu dari sistem. Aktor dilukiskan dengan peran yang mereka mainkan dalam use case, seperti Staff, Kurir dan lain-lain. Dalam use case diagram terdapat satu aktor pemulai atau initiator actor yang membangkitkan rangsangan awal terhadap sistem, dan mungkin sejumlah aktor lain yang berpartisipasi atau participating actor. Akan sangat berguna untuk mengetahui siapa aktor pemulai tersebut.

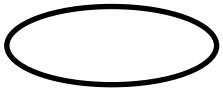


<nama aktor>

Gambar 2. 10 Aktor UseCase

2.7.2. Use Case

Use case yang dibuat berdasarkan keperluan aktor merupakan gambaran dari “apa” yang dikerjakan oleh sistem, bukan “bagaimana” sistem mengerjakannya. Use case diberi nama yang menyatakan apa hal yang dicapai dari interaksinya dengan aktor.[10]

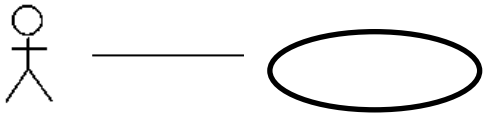


<nama usecase>

Gambar 2. 11 Use Case

2.7.3. Relationship

Relasi (relationship) digambarkan sebagai bentuk garis antara dua simbol dalam use case diagram. Relasi antara actor dan use case disebut juga dengan asosiasi (association). Asosiasi ini digunakan untuk menggambarkan bagaimana hubungan antara keduanya. Relasi-relasi yang terjadi pada use case diagram bisa antara actor dengan use case atau use case dengan use case.



Gambar 2. 12 Relasi Use Case

2.8. Activity Diagram

Diagram aktivitas atau activity diagram menggambarkan workflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut ini.

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan

2. Urutan atau pengelompokan tampilan dari sistem/user interface dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan

3. Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya. [10]

2.9. Sequence Diagram

Diagram sequence menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sequence maka harus diketahui objek-objek yang terlibat dalam sebuah use case beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya diagram sequence yang harus digambar adalah sebanyak pendefinisian use case yang memiliki proses sendiri atau yang penting semua use case yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sequence sehingga semakin banyak use case yang didefinisikan maka diagram sequence yang harus dibuat juga semakin banyak.

Penomoran pesan berdasarkan urutan interaksi pesan. Penggambaran letak pesan harus berurutan, pesan yang lebih atas dari lainnya adalah pesan yang berjalan terlebih dahulu.[10]

2.10. Class Diagram

Diagram kelas atau class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem. Susunan struktur kelas yang baik pada diagram kelas ebaiknya memiliki jenis-jenis kelas berikut.

1. Kelas main yaitu kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.

2. Kelas yang menangani tampilan yaitu Kelas yang mendefinisikan dan mengatur tampilan ke pemakai.

3. Kelas yang diambil dari pendefinisian usecase yaitu kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian use case.

4. Kelas yang diambil dari pendefinisian data yaitu Kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

Jenis-jenis kelas diatas juga dapat digabungkan satu sama lain sesuai dengan pertimb``angan yang dianggap baik asalkan fungsi-fungsi yang sebaiknya ada pada struktur kelas tetap ada. Susunan kelas juga dapat ditambahkan kelas utilitas seperti koneksi ke basis data, membaca file teks, dan lain sebagainya sesuai kebutuhan.

Dalam mendefinisikan metode yang ada di dalam kelas perlu memperhatikan apa yang disebut dengan *cohesion* dan *coupling*. *Cohesion* adalah ukuran seberapa dekat keterkaitan instruksi di dalam sebuah metode terkait satu sama lain sedangkan *coupling* adalah ukuran seberapa dekat keterkaitan instruksi antara metode yang satu dengan metode yang lain dalam sebuah kelas. Sebagai aturan secara umum maka sebuah metode yang dibuat harus memiliki kadar choesion yang kuat dan kadar coupling yang lemah.[10]

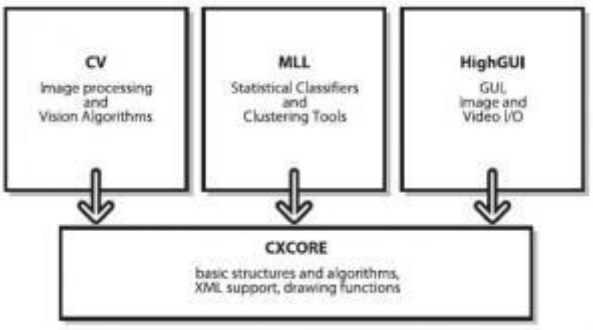
2.11. Open CV Library

OpenCV (*Open Computer Vision*) adalah sebuah API (*Application Programming Interface*) Library yang sudah sangat familiar pada Pengolahan Citra *Computer Vision*. *Computer Vision* itu sendiri adalah salah satu cabang dari Bidang Ilmu Pengolahan Citra (*Image Processing*) yang memungkinkan komputer dapat melihat seperti manusia. Dengan *vision* tersebut komputer dapat mengambil keputusan, melakukan aksi, dan mengenali terhadap suatu objek. Beberapa pengimplementasian dari *Computer Vision* adalah *Face Recognition, Face Detection, Face/Object Tracking, Road Tracking, dll*. OpenCVadalah library *Open Source* untuk *Computer Vision* untuk C/C++, OpenCV didesain untuk aplikasi real-time, memiliki fungsi-fungsi akuisisi yang baik untuk *image/video*.[11]

OpenCV sendiri terdiri dari 5 library, yaitu :

1. CV : untuk algoritma *Image processing dan Vision.*
2. ML : untuk *machine learning library*
3. Highgui : untuk GUI, *Image* dan *Video* I/O.
4. CXCORE : untuk struktur data, *support XML* dan fungsi-fungsi grafis.

Struktur dan Konten OpenCV :



Gambar 2. 13 Struktur Konten Open CV

2.12. Microsoft Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak yang dapat digunakan untuk melakukan pengembangunan aplikasi, baik itu aplikasi bisnis atau aplikasi personal dalam bentuk aplikasi console, aplikasi Windows, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, Visual Basic .NET, Visual InterDev, Visual J++, Visual J#, Visual FoxPro, dan Visual SourceSafe [12].

Visual C# dibuat berdasarkan pemrograman C# yang merupakan bahasa pemrograman berorientasi objek dan mempunyai banyak kesamaan dengan C++, Java, dan VB. C# pada faktanya merupakan kombinasi antara efisiensi pemrograman C++, kesederhanaan pemrograman Java, dan penyederhanaan dari pemrograman Visual Basic.

Seperti pemrograman Java, C# juga tidak memperbolehkan multiple inheritance atau pengunaan pointer (pada safe/managed code), tetapi C# menyediakan *garbage memory colletion* pada saat runtime dan pada saat pengecekan akses memori. Meskipun bertentangan dengan pemrograman Java, C# tetap mempertahankan operasi unik yang terdapat pada bahasa pemrograman C++ seperti *overloading, enumerations, pre-processor directive, pointer (pada unmanaged/unsafe code),* dan fungsi *pointer*. Seperti halnya Visual Basic, bahasa pemrograman C# juga dilengkapi dengan *properties*.

Sebagai tambahan, bahasa pemrograman C# juga datang dengan beberapa fitur baru dan sangat menarik seperti *reflections, attributes, marshalling, remote, threads, streams data acces* dengan ADO.NET, dan masih banyak lagi.[12]