BAB 2
SISTEM DETEKSI DAN PENGHITUNG OBYEK

Bab ini membahas mengenai analisis sistem yang dibutuhkan, kemudian arsitektur sistem, serta tahapan deteksi dan penghitung obyek. Pada tahapan deteksi obyek, terdapat beberapa proses mulai dari ekstraksi ciri, proses pelatihan, proses deteksi dan penjejakan obyek, dan proses menghitung jumlah obyek. Metode-metode yang digunakan dalam penelitian ini dijelaskan juga pada bab ini.

2.1 Analisis Kebutuhan

Bagian analisis ini tersusun atas berbagai permasalahan yang melatarbelakangi pengembangan ini beserta kebutuhan yang harus dapat diatasi oleh sistem ini.

Sesuai dengan yang telah disampaikan pada bagian latar belakang pada bab sebelumnya, para pengusaha tempat umum membutuhkan data berbagai jumlah pengunjung yang mengunjungi tempat yang dikelolanya, baik yang berjalan kaki, mengendarai motor ataupun mobil. Penghitungan jumlah pengunjung juga diharapkan dapat dilakukan secara otomatis, karena akan dilakukan dalam jangka waktu yang lama. Oleh karena itu, dibutuhkan sebuah sistem yang dapat melakukan proses penghitungan pengunjung secara otomatis.

Dalam melakukan proses penghitungan pengunjung, sistem harus memiliki fitur untuk mendeteksi dan membedakan orang yang berjalan kaki, mengendarai motor ataupun mobil. Video merupakan gambar bergerak (setiap gambar dikenal dengan istilah *frame*), sehingga proses pendeteksian obyek akan dilakukan pada setiap *frame*. Dengan adanya tahap deteksi obyek tersebut, maka dimungkinkan dibangunnya tahap penjejakan obyek untuk menjejaki obyek yang terdeteksi antar *frame* pada video. Dengan proses penjejakan tersebut, sistem akan mengunci obyek yang terdeteksi antar *frame*. Setelah itu sistem dapat memutuskan apakah obyek yang dijejaki tersebut akan digolongkan sebagai manusia yang berjalan kaki, mengendarai motor ataupun mengendarai mobil atau tidak, berdasarkan pada proses penjejakan obyek sebelumnya.

Berdasarkan hal-hal yang telah disebutkan sebelumnya, berikut adalah kebutuhan-kebutuhan yang perlu disediakan solusinya dalam sistem yang akan dikembangkan:

- Sistem yang akan dibangun adalah sistem yang dapat menerima data masukan berupa video maupun rekama secara waktu nyata (*realtime*).
- Sistem harus memiliki fitur pendeteksi pejalan kaki, pengendara motor, dan juga mobil. Fitur ini didapatkan dengan mengimplementasikan metode Boosting termodifikasi yang diajukan dalam tesis ini.
- Sistem yang akan dibangun memiliki kemampuan untuk menjejaki pengunjung. Penjejakan akan dilakukan untuk setiap jenis obyek.
- Sistem dapat menghitung jumlah pengunjung yang terdapat pada rekaman secara langsung, dibedakan untuk jumlah pejalan kaki, pengendara motor, dan mobil.

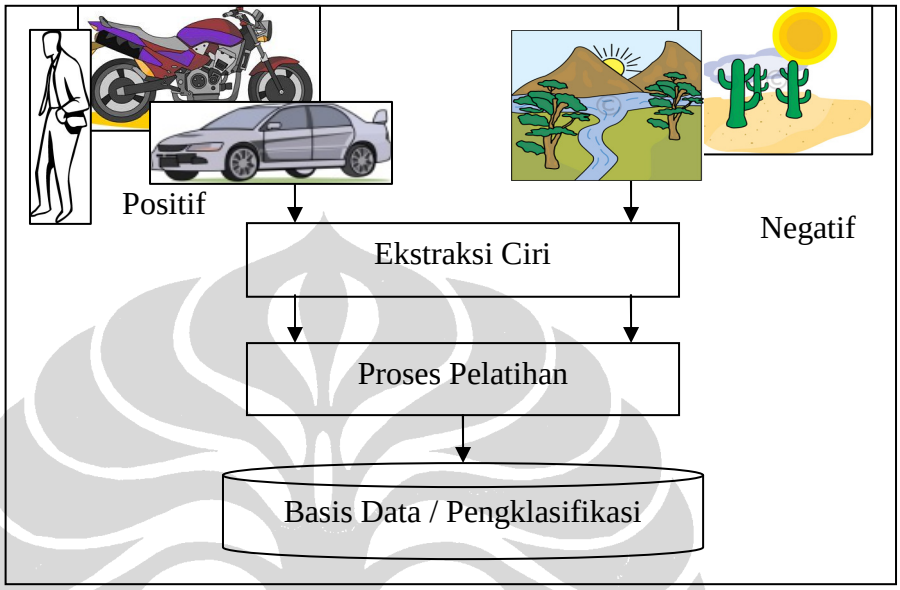
2.2 Arsitektur Sistem Deteksi dan Penghitung Obyek

Sebelum menghitung obyek, obyek yang muncul pada video harus dideteksi terlebih dahulu. Deteksi obyek merupakan proses untuk menentukan apakah suatu citra atau *frame* video mempunyai obyek yang ingin dideteksi atau tidak, sedangkan menghitung obyek adalah proses menghitung berapa banyak obyek yang muncul di dalam video dalam beberapa urutan *frame* berturut-turut.

Sistem harus dilatih terlebih dahulu agar bisa membedakan citra yang mempunyai obyek yang ingin dideteksi atau tidak. Citra yang mempunyai obyek yang ingin dideteksi dikelompokkan menjadi citra kelas positif, sedangkan citra yang tidak mempunyai obyek tersebut dikelompokkan menjadi citra kelas negatif. Pada tesis ini, obyek yang akan dideteksi adalah pejalan kaki, pengendara motor, dan mobil. Masing-masing jenis obyek tersebut sangat mudah berubah karena banyaknya variasi tipe pakaian dan pose, ataupun warna, bentuk, dan juga tekstur. Variasi tersebut mengakibatkan sulitnya membedakan obyek yang ingin dideteksi dari obyek-obyek lainnya pada sebuah citra. Untuk itu diperlukan suatu metode ekstraksi ciri yang bisa menyelesaikan permasalahan tersebut.

Arsitektur sistem pada deteksi obyek yang digunakan terdiri dari dua tahapan yaitu tahapan pelatihan dan tahapan penghitungan obyek. Sistem yang

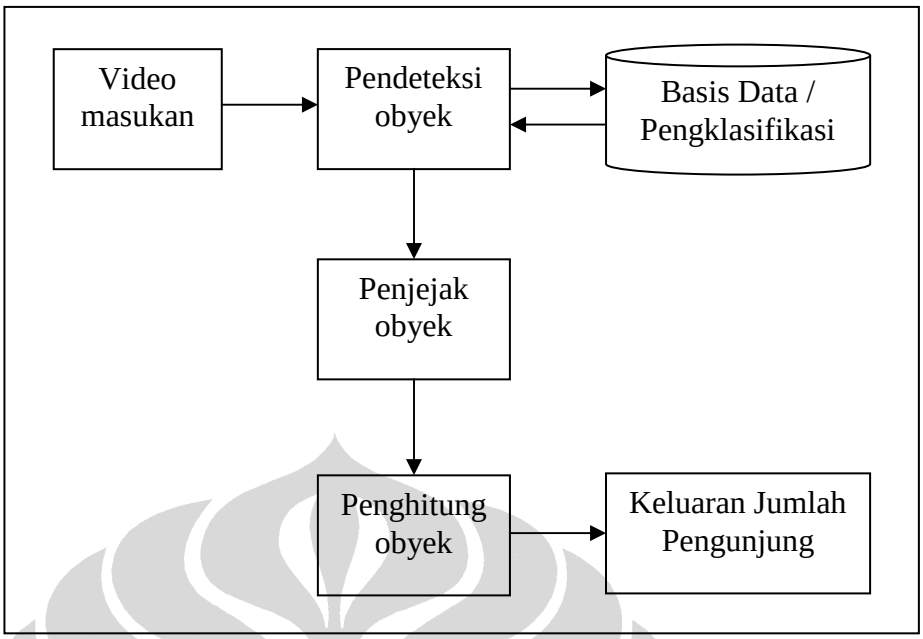
diimplementasikan untuk pelatihan dan deteksi obyek berdasarkan pada kerangka kerja [4], sedangkan sistem penghitung pengunjung merupakan pengembangan sistem dari [13]. Arsitektur sistem untuk tahapan pelatihan ditunjukkan pada Gambar 2.1.



Gambar 2.1 Arsitektur Tahap Pelatihan

Pada tahap pelatihan, citra obyek yang ingin dideteksi (citra positif) dan citra latar belakang (citra negatif) diekstrak cirinya. Kemudian ciri-ciri tersebut akan menjadi input untuk proses pelatihan. Hasil dari tahap pelatihan adalah suatu basis data atau pengklasifikasi dengan format file XML.

Sedangkan pada tahapan penghitungan obyek, masukannya adalah file video ataupun hasil rekam kamera CCTV secara realtime. Setiap *frame* video akan diproses oleh pendeteksi obyek berdasarkan pengklasifikasi yang dihasilkan dari proses pelatihan. Tiap obyek yang terdeteksi akan dijejaki oleh penjejak obyek. Kemudian penghitung obyek akan menghitung obyek yang dijejaki jika memenuhi syarat-syarat tertentu. Hasil dari tahap penghitungan obyek adalah jumlah pejalan kaki, pengendara motor, atau mobil yang melintas pada video tersebut. Arsitektur sistem untuk tahapan penghitungan obyek ditunjukkan pada Gambar 2.2.



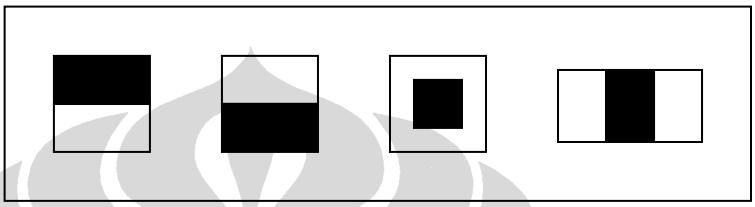
Gambar 2.2 Arsitektur Tahap Penghitungan Obyek

2.2.1 Ekstraksi Ciri

Ekstraksi ciri adalah proses untuk mengekstraksi informasi penting seperti warna, tekstur, dan bentuk. Deteksi obyek merupakan salah satu cara untuk ekstraksi informasi penting pada sebuah citra misalnya manusia, motor, mobil, atau obyek lainnya. Manusia, motor, dan mobil merupakan obyek yang dibahas, karena deteksi obyek-obyek tersebut dapat digunakan untuk berbagai aplikasi seperti sistem pengawasan. Deteksi manusia pada citra statis merupakan permasalahan yang sulit karena memiliki beberapa karakteristik yang seringkali berubah, misalkan variasi tipe dan warna pakaian, dan juga gaya. Sedangkan untuk deteksi motor atau mobil, karakteristik yang sering berubah adalah warna dan bentuk kendaraan, dan juga si pengendara motor untuk deteksi obyek motor. Variasi tersebut menyebabkan sulitnya memisahkan obyek manusia, motor ataupun mobil dari obyek-obyek yang lain pada sebuah citra [14].

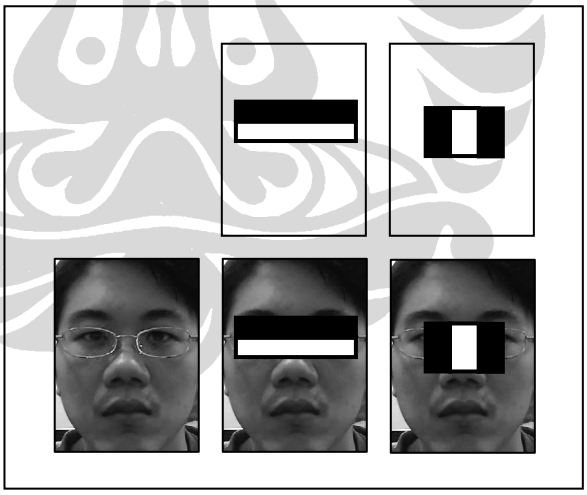
Sistem yang digunakan [4] menggunakan ekstraksi ciri Haar *wavelet* yang digunakan [17]. *Wavelet* adalah suatu fungsi matematis yang digunakan untuk membagi signal pada citra menjadi komponen-komponen yang lebih kecil sehingga dapat diproses. Haar *wavelet* adalah satu *wavelength square wave*.

Apabila digambarkan dalam 2 dimensi, *square wave* adalah sepasang persegi panjang yang bersebelahan, satu berwarna terang dan satunya lagi berwarna gelap. Kombinasi yang digunakan untuk deteksi obyek pada citra sebenarnya bukan Haar *wavelet* yang asli, melainkan menggunakan kombinasi persegi panjang yang lebih cocok untuk pendeteksian obyek. Oleh karena perbedaan ini, ciri yang diekstrak tidak disebut ciri Haar *wavelet*, melainkan ciri Haar-*like*. Gambar 2.3 menunjukkan contoh ciri Haar-*like* yang digunakan.



Gambar 2.3 Contoh Ciri Haar-*like* yang Digunakan

Ciri ciri Haar-*like* ini akan diekstrak dari citra, dan kemudian digunakan sebagai masukan untuk proses pelatihan. Gambar 2.4 menunjukkan contoh ekstraksi ciri Haar-*like* dari gambar wajah manusia.



Gambar 2.4 Contoh Ekstraksi Ciri Haar-*like* pada Wajah Manusia

2.2.2 Proses Pelatihan

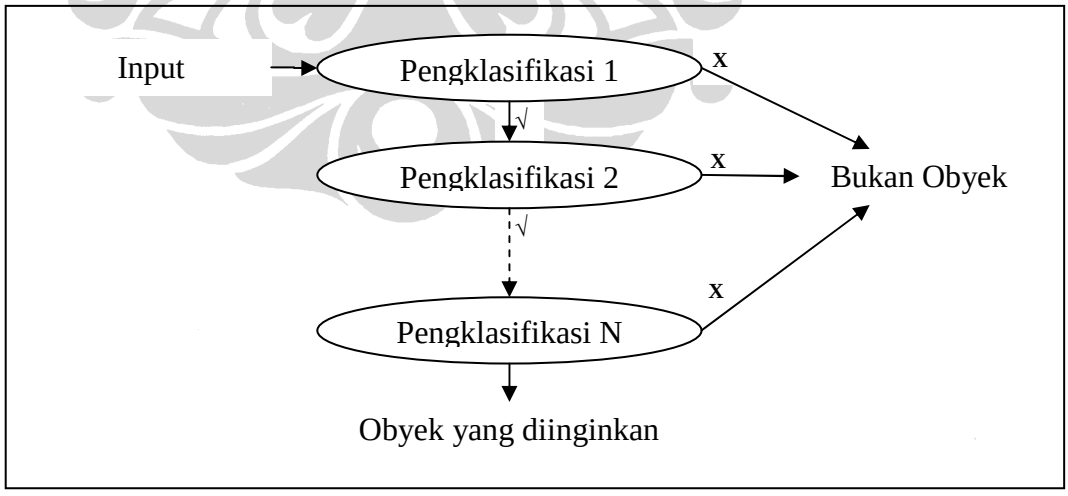
Pada proses pelatihan, sistem dilatih untuk belajar membedakan citra yang mempunyai obyek yang ingin dideteksi atau tidak. Data pelatihan yang digunakan pada proses pelatihan terdiri dari citra kelas positif dan kelas negatif. Citra kelas

positif merupakan citra yang mempunyai satu obyek manusia, motor atau mobil. Sedangkan citra kelas negatif merupakan citra dengan latar belakang natural ataupun bangunan yang tidak memiliki obyek manusia, motor ataupun mobil.

Data pelatihan menggunakan citra untuk setiap obyek berukuran berbeda. Untuk manusia atau pejalan kaki, ukuran citra yang digunakan adalah 20 x 50 piksel. Untuk motor, ukuran citra yang digunakan adalah 38 x 38 piksel. Sedangkan untuk mobil, ukuran citra yang digunakan adalah 50 x 25 piksel. Ukuran-ukuran tersebut sesuai dengan bentuk skala tinggi dan lebar obyek yang terlihat dari samping. Berdasarkan pelatihan-pelatihan yang dilakukan sebelumnya, ukuran citra yang digunakan dapat berukuran lebih besar karena tekstur akan dapat terdeteksi lebih baik. Namun karena batasan perangkat keras yang digunakan, ukuran citra diperkecil dari aslinya untuk mengurangi kapasitas memori komputer yang dibutuhkan dan juga untuk mempercepat proses pelatihan.

Ciri Haar-*like* dari citra positif dan citra negatif kemudian diekstrak dan digunakan sebagai masukan dari proses pelatihan. Proses pelatihan menggunakan metode Boosting akan dibahas di bab berikutnya.

Hasil dari pelatihan adalah suatu basis data pengklasifikasi yang berlapis (*cascade*), dimana suatu citra akan dianggap memiliki obyek yang ingin dideteksi jika berhasil melewati seluruh lapisan pengklasifikasi.



Gambar 2.5 Ilustrasi Cara Kerja Pengklasifikasi yang Berlapis

Dengan pengklasifikasi yang berlapis ini, setiap pengklasifikasi tidak perlu memiliki akurasi deteksi yang maksimal, namun gabungan dari semua

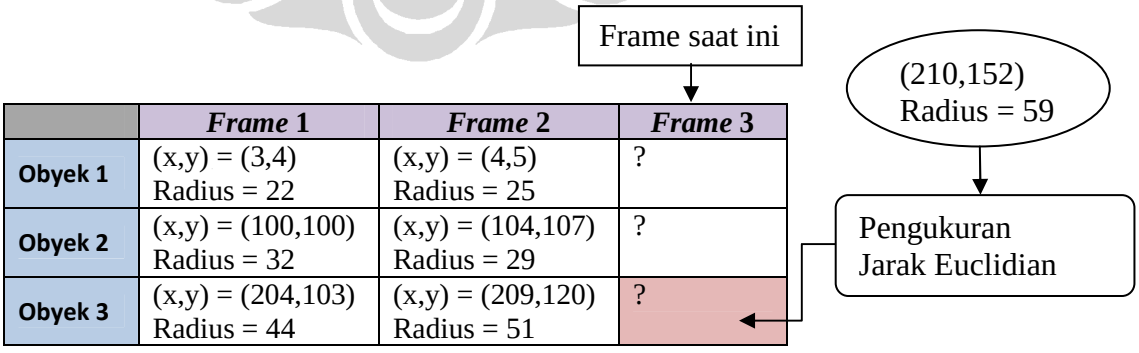
pengklasifikasi dapat memiliki akurasi deteksi yang tinggi. Sebagai contoh, apabila satu pengklasifikasi memiliki *hit rate* sebesar 0,999 dan *false alarm* sebesar 0,5, maka 20 pengklasifikasi berlapis akan memiliki *hit rate* sebesar $0{,}999^{20} \approx 0{,}98$ dan *false alarm* sebesar $0{,}5^{20} \approx 10^{-6}$.

2.2.3 Proses Penjejakan Obyek

Proses penjejakan obyek merupakan proses deteksi menggunakan basis data pengklasifikasi hasil pelatihan dan mengikuti pergerakan obyek yang terdeteksi dari satu *frame* video ke *frame* lainnya. Proses deteksi dilakukan dengan mengaplikasikan pengklasifikasi secara bertahap per bagian area dari satu *frame*, sesuai dengan ukuran latihan pengklasifikasi. Pengklasifikasi juga diskalakan, sehingga pengklasifikasi dapat mendeteksi obyek yang skalanya berbeda dengan saat pelatihan. Hasil deteksi obyek berbentuk lingkaran, sehingga didapatkan informasi letak koordinat piksel titik tengah dari obyek dan ukuran radiusnya yang menunjukkan tinggi atau lebar obyek yang terdeteksi. Proses penjejakan obyek akan dilakukan pada titik tengah obyek tersebut.

Proses penjejakan obyek akan memadankan obyek yang terdeteksi di *frame* saat ini terhadap tepat satu *frame* sebelumnya. Proses pemadanan obyek akan menggunakan metode jarak Euclidian yang akan dijelaskan di sub bab 2.3.

Untuk melakukan pemadanan, digunakan array untuk mencatat berbagai informasi yang diperlukan untuk menjejaki obyek yang terdeteksi.



	***Frame* 1**	***Frame* 2**	***Frame* 3**
Obyek 1	(x,y) = (3,4) Radius = 22	(x,y) = (4,5) Radius = 25	?
Obyek 2	(x,y) = (100,100) Radius = 32	(x,y) = (104,107) Radius = 29	?
Obyek 3	(x,y) = (204,103) Radius = 44	(x,y) = (209,120) Radius = 51	?

Gambar 2.6. Ilustrasi Array yang Mencatat Informasi Lokasi Obyek

Pada gambar 2.6 di atas dapat dilihat ilustrasi untuk menjelaskan proses penjejakan obyek antar *frame*. Pada ilustrasi tersebut, diketahui bahwa *frame* saat ini adalah *frame* ketiga, dan juga diketahui bahwa pada *frame* pertama dan *frame* kedua ada tiga obyek yang terdeteksi lengkap dengan koordinat dan radiusnya masing-masing. Pada *frame* ketiga, sedang terdeteksi sebuah obyek dengan koordinat piksel (210,152) dan memiliki radius sebesar 59 piksel. Kedua informasi ini akan dijadikan masukan ke dalam proses pengukuran jarak Euclidian yang akan memutuskan berdasarkan jarak paling minimum. Proses ini akan menghasilkan keputusan, kepada obyek nomor berapakah ia termasuk. Seharusnya proses penghitungan yang tepat akan menghasilkan keputusan untuk memadankannya ke dalam obyek nomor 3, karena dilihat dari koordinat piksel dan radiusnya, obyek yang sedang terdeteksi saat ini terletak paling dekat dengan pengunjung nomor 3 pada *frame* sebelumnya.

2.2.4 Proses Menghitung Jumlah Obyek

Proses menghitung obyek adalah proses menghitung berapa banyak obyek yang terdeteksi pada video. Obyek akan dianggap melewati daerah yang diawasi dan dihitung apabila ia telah memenuhi semua kondisi berikut, yaitu:

- Terdeteksi pada 4 *frame* sebelumnya.
- Telah menghilang keberadaannya dari *frame* saat ini.
- Tepat pada satu *frame* sebelumnya berada di dalam zona penghitungan.

Syarat pertama berguna untuk memastikan validitas bahwa obyek yang terdeteksi bukanlah *false positives*, sehingga ia perlu tampil pada empat *frame* sebelumnya. Angka empat dipilih, karena mengingat proses pendeteksian wajah pada banyak citra berurutan sangat rawan terjadinya *false negative*, sehingga jika angka yang dipilih terlalu besar, ada kemungkinan proses penjejakan terputus dan berpengaruh pada kinerja penghitungan pengunjung.

Syarat kedua untuk memastikan bahwa pengunjung telah menghilang keberadaannya atau telah meninggalkan daerah yang diawasi. Sedangkan kondisi ketiga memastikan bahwa pengunjung telah meninggalkan daerah melalui daerah zona penghitungan atau masuk melewati pintu masuk. Zona penghitungan (dikenal juga sebagai *virtual gate* [16]) yang terdapat pada syarat ketiga

merupakan zona yang telah ditentukan sebelumnya dan tergantung pada posisi ruangan. Zona penghitungan umumnya berada tepat di dekat pintu masuk.

Pada gambar 2.7 diilustrasikan letak zona penghitungan. Letak zona penghitungan diletakkan di sebelah kiri dengan asumsi pengunjung akan datang dari arah kanan menuju pintu masuk di sebelah kiri. Jika kamera akan dipasang di tempat lain, maka zona penghitungan beserta dengan proses penjejakan harus diatur ulang.

Gambar 2.7 Letak Zona Penghitungan pada Video Ujicoba

2.3 Jarak Euclidean

Apabila terdapat lebih dari satu obyek yang ingin dideteksi pada video, sistem menggunakan metode jarak Euclidean untuk membandingkan koordinat obyek pada *frame* sebelumnya dan pada *frame* saat ini untuk menentukan padanan obyek masing-masing.

Jarak Euclidian adalah jarak terpendek antara dua buah titik. Apabila terdapat dua buah titik, maka jarak terpendek didapatkan dengan cara menarik garis lurus yang menghubungkan kedua titik tersebut. Dalam ruang Euclidian berdimensi n, R^n, jarak antara titik x dan y dapat dirumuskan sebagai berikut [15]:

$$D = |x - y| = \sqrt{\sum_{i=1}^{n} |x_i - y_i|^2} \qquad (2.1)$$

Dimana n adalah jumlah titik dalam R^n.

Karena sistem yang dikembangkan bekerja dalam ruang Euclidian berdimensi dua, maka jarak Euclidian antara titik $p(x_1,y_1)$ dan $q(x_2,y_2)$ dapat dihitung dengan menggunakan rumus berikut:

$$D(p,q) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \qquad (2.2)$$

Jarak Euclidian (jarak antar obyek pada citra) maksimum yang ditoleransi dalam berbagai video dapat beragam, tergantung dari letak kamera. Pada tesis ini, toleransi jarak maksimum adalah 50 piksel. Nilai ini mendapatkan hasil terbaik diukur berdasarkan eksperimen.

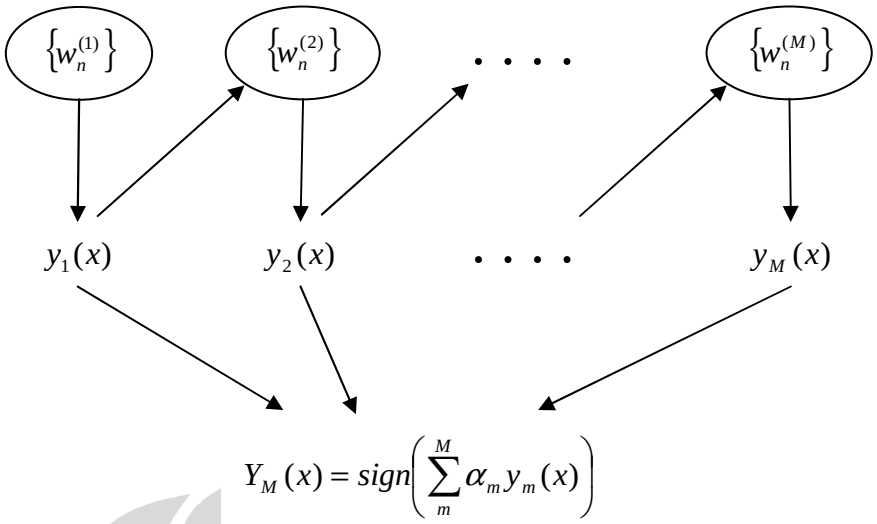
BAB 3
METODE MULTIKELAS BOOSTING TERMODIFIKASI

Bab ini mengkaji teori-teori yang mendasari penelitian ini, serta mengusulkan metode multikelas Boosting termodifikasi. Teori dan penjelasan tentang Boosting dan AdaBoost M2 akan dibahas terlebih dahulu pada bab ini.

3.1 Boosting

Boosting, atau juga dikenal dengan nama AdaBoost, adalah suatu teknik untuk mengkombinasikan beberapa pengklasifikasi dasar (*multiple base classifer*) untuk menghasilkan suatu *committee* (rata-rata prediksi dari beberapa pengklasifikasi dasar) yang memiliki kinerja jauh lebih baik daripada pengklasifikasi-pengklasifikasi dasarnya [3]. Boosting didefinisikan juga sebagai metode untuk menciptakan pengklasifikasi kuat (*strong classifier*) yang akurat dengan mengkombinasikan beberapa pengklasifikasi lemah (*weak classifier*) [11]. Boosting dapat memberikan hasil yang baik walaupun pengklasifikasi dasar memiliki performa yang sedikit lebih baik daripada memilih acak (*random*). Metode Boosting seringkali disebut sebagai algoritma AdaBoost, kependekan dari "*Adaptive* Boosting".

Apabila ada permasalahan klasifikasi untuk 2 kelas, dimana training data terdiri dari vektor input $x_1,\ldots, x_N$ dan memiliki target $t_1,\ldots, t_N$ dimana $t_N \in \{-1,1\}$. Untuk setiap data diberikan bobot w_n, dimana awalnya diset sebagai 1/N untuk semua titik. Proses pelatihan pengklasifikasi dasar menggunakan bobot data untuk fungsi $y(x) \in \{-1,1\}$. Dalam boosting, pengklasifikasi dasar dilatih secara berurutan, dimana setiap pengklasifikasi dasar dilatih dengan koefisien pembobotan yang lebih besar untuk data yang salah diklasifikasikan oleh pengklasifikasi dasar sebelumnya. Setelah melakukan proses pelatihan, hasil koefisien bobot akan dikombinasikan untuk membentuk suatu pengklasifikasi gabungan yang lebih kuat. Ilustrasi skema boosting dapat dilihat pada gambar 3.1.



Gambar 3.1 Skema Metode Boosting

Setiap pengklasifikasi dasar $y_m(x)$ dilatih menggunakan bobot $w_n^{(m)}$ tergantung kepada performa pengklasifikasi dasar sebelumnya $y_{m-1}(x)$. Hasil pelatihan semua pengklasifikasi dasar dikombinasikan menjadi pengklasifikasi baru $Y_m(x)$. Algoritma AdaBoost adalah sebagai berikut:

1. Inisialisasi bobot data $\{ w_n \}$ dengan $w_n^{(m)} = 1/N$ untuk n = 1,2,…,N.
2. For m = 1,…, M:
   a. *Train* $y_m(x)$ dengan meminimalkan fungsi kesalahan (*error function*) sebagai berikut:

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(x_n) \neq t_n) \qquad (3.1)$$

   Dimana $I(y_m(x_n) \neq t_n)$ adalah fungsi indikator yang bernilai 1 jika $y_m(x_n) \neq t_n$ dan bernilai 0 jika sebaliknya.

   b. Evaluasi kesalahan dengan

$$\varepsilon_m = \frac{\sum_{n=1}^{N} w_n^{(m)} I(y_m(x_n) \neq t_n)}{\sum_{n=1}^{N} w_n^{(m)}} \qquad (3.2)$$

Dan kemudian hasilnya digunakan untuk evaluasi:

$$\alpha_m = \ln\left\{\frac{1-\varepsilon_m}{\varepsilon_m}\right\} \tag{3.3}$$

c. Memperbaiki bobot data dengan:

$$w_n^{m+1} = w_n^m \exp\{\alpha_m I(y_m(x_n) \neq t_n)\} \tag{3.4}$$

3. Membuat prediksi menggunakan model terakhir sebagai berikut:

$$Y_M(x) = sign\left(\sum_{m=1}^{M} \alpha_m y_m(x)\right) \tag{3.5}$$

Pengklasifikasi dasar yang pertama $y_1(x)$ dilatih menggunakan koefisien bobot $w_n^{(1)}$ yang nilainya sama semua. Pada persamaan (3.4), nilai bobot w_n^m dinaikkan untuk data yang salah diklasifikasi dan diturunkan untuk data yang terklasifikasi dengan benar. Pengklasifikasi yang terbaik adalah yang memberikan bobot lebih besar pada data yang salah terklasifikasi sehingga dapat diperbaiki di klasifikasi sesudahnya. Besarnya ε_m menggambarkan evaluasi bobot dengan menghitung rata-rata kesalahan setiap pengklasifikasi dasar pada data. Koefisien bobot α_m didefinisikan dengan persamaan (3.3) untuk memberikan bobot terbesar agar pengklasifikasi lebih akurat ketika menghitung keluaran pada persamaan (3.5). Persamaan (3.5) adalah pengklasifikasi hasil yang merupakan persamaan fungsi tujuan yang digunakan sebagai penentuan prediksi hasil data input. Persamaan (3.5) merupakan fungsi kombinasi dari bobot α_m dan hasil klasifikasi dari beberapa pengklasifikasi dasar $y_m(x)$.

3.2 AdaBoost M2

Metode AdaBoost digunakan untuk memecahkan masalah klasifikasi dalam dua kelas. AdaBoost M2 adalah pengembangan dari metode AdaBoost untuk permasalahan multikelas. Dalam metode AdaBoost di atas, basis data pengklasifikasi akan menghasilkan hipotesa dengan bentuk $h: X \times Y \rightarrow [0,1]$, yaitu bernilai 1 jika masukan sesuai dengan obyek yang dilatihkan dan bernilai 0

jika sebaliknya. Dapat dikatakan bahwa $h(x, y)$ mengukur derajat kepercayaan apakah y adalah label yang tepat untuk instansi x. Untuk permasalahan multikelas (banyak y), apabila $h(x, y)$ bernilai sama untuk semua y, maka dapat dikatakan bahwa hipotesa yang dihasilkan tidak ada gunanya untuk instansi x. Kebalikannya, sedikit saja perbedaan nilai yang dihasilkan sangatlah berguna, karena itu berarti y tertentu lebih dipercaya dari yang lainnya. Sebagai contoh, apabila kita memakai contoh perangkat lunak pengenal karakter tulisan pada layar (*optical character recognition* / OCR), angka "7" dan angka "9" akan agak sulit dibedakan. Apabila ada dua pengklasifikasi AdaBoost untuk memeriksa, apakah karakter tersebut adalah "7" dan apakah karakter tersebut adalah "9", keduanya menghasilkan nilai 1, maka tidak dapat ditentukan mana yang tepat.

Metode AdaBoost M2 menggunakan *pseudo loss* untuk mengevaluasi kesalahan dalam pelatihan agar dapat lebih menekankan ke bagian yang sulit dibedakan [21]. Hasil perhitungan *pseudo loss* ini akan mempengaruhi hasil prediksi akhir ketika pengklasifikasi dijalankan.

Algoritma AdaBoost M2 adalah sebagai berikut:

1. Inisialisasi bobot data $\{ w_n \}$ dengan $w_n^{(m)} = 1/N$ untuk n = 1,2,…,N.
2. For m = 1,…, M:
   a. Menghitung fungsi bobot label q_m sebagai berikut:

$$\text{Hitung } W_n = \sum_{n \neq n_i} w_n \text{ , kemudian } q_m(n) = \frac{w_n}{W_n} \tag{3.6}$$

   b. Hitung distribusi D_m sebagai berikut:

$$D_m(n) = \frac{W_n}{\sum_{n=1}^{N} W_n} \tag{3.7}$$

   c. *Train* $y_m(x)$ dengan meminimalkan fungsi kesalahan (*error function*) sebagai berikut:

$$J_m = \sum_{n=1}^{N} w_n^{(m)} I(y_m(x_n) \neq t_n) \tag{3.8}$$

Dimana $I(y_m(x_n) \neq t_n)$ adalah fungsi indikator yang bernilai 1 jika $y_m(x_n) \neq t_n$ dan bernilai 0 jika sebaliknya, untuk singkatnya disebut h_m.

d. Evaluasi kesalahan dengan menghitung *pseudo loss*:

$$\varepsilon_m = \frac{1}{2}\sum_{n=1}^{N} D_m(n)\left(1 - h_m(x_n, y_n) + \sum_{y \neq y_n} q_m(n) h_m(x_n, y)\right) \quad (3.9)$$

Dan kemudian hasilnya digunakan untuk evaluasi:

$$\beta_m = \frac{\varepsilon_m}{1-\varepsilon_m} \quad (3.10)$$

e. Memperbaiki bobot data dengan:

$$w_n^{m+1} = w_n^m \beta^{(1/2)(1+h_m(x_n,y_n)-h_m(x_n,y))} \quad (3.11)$$

3. Membuat prediksi menggunakan model terakhir sebagai berikut:

$$Y_M(x) = \arg\max \sum_{m=1}^{M} \left(\log \frac{1}{\beta_m}\right) h_m(x, y) \quad (3.12)$$

Setelah bobot data diinisialisasi, pengklasifikasi dasar dilatih serupa metode Boosting biasa. Namun untuk setiap pengklasifikasi dasar, diperhitungkan nilai W_n, q_m dan D_m untuk setiap ciri. W_n adalah jumlah bobot dari ciri-ciri lain selain ciri ke n. Fungsi bobot label q_m yang merupakan perbandingan bobot satu ciri dengan W_n. Distribusi D_m adalah perbandingan antara W_n satu ciri dengan total W_n. Evaluasi kesalahan memperhitungkan *pseudo loss*, yang kemudian digunakan untuk memperbaiki bobot data pengklasifikasi dasar selanjutnya dan juga untuk memprediksi model terakhir.

3.3 Usulan Multikelas Boosting Termodifikasi dan Implementasinya

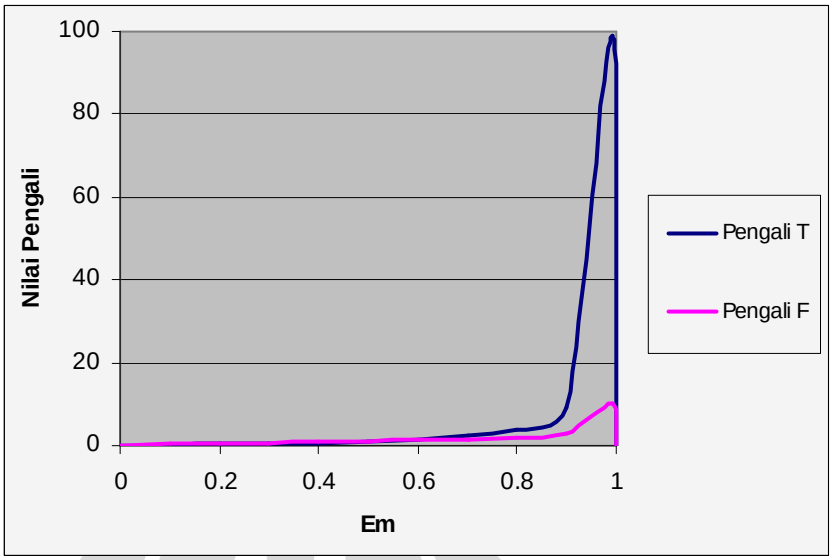
Pada sub sub bab berikut akan dijelaskan tentang usulan dari metode yang diusulkan dan implementasinya pada sistem.

3.3.1 Usulan Modifikasi AdaBoost M2 dengan Fungsi Indikator

Pada algoritma AdaBoost, terdapat fungsi indikator $I(y_m(x_n) \neq t_n)$, dimana fungsi akan bernilai 1 jika $y_m(x_n) \neq t_n$ (hasil pembobotan training belum sesuai dengan y tujuan) dan bernilai 0 apabila sebaliknya. Namun pada AdaBoost M2, perubahan bobot akan selalu dilakukan walaupun hasil pembobotan training sudah sesuai. Tabel 3.1 dan gambar 3.2 menunjukkan perubahan nilai perubahan pengali bobot $\beta^{(1/2)(1+h_m(x_n,y_n)-h_m(x_n,y))}$ pada persamaan (3.11) terhadap pseudo loss ε_m pada persamaan (3.9) dengan contoh nilai ε_m berkisar dari 0 sampai 1. Pengali T (*true*) adalah pengali bobot jika hasil pembobotan training sesuai dengan y tujuan, sedangkan pengali F (*false*) adalah pengali bobot jika sebaliknya.

Tabel 3.1 Tabel Perbandingan Nilai Pengali T dan F terhadap ε_m

ε_m	**Pengali T**	**Pengali F**
0	0	0
0.1	0.1111111	0.333333
0.2	0.25	0.5
0.3	0.4285714	0.654654
0.4	0.6666667	0.816497
0.5	1	1
0.6	1.5	1.224745
0.7	2.3333333	1.527525
0.8	4	2
0.9	9	3
0.99	99	9.949874
1	-	-



Gambar 3.2 Grafik Perbandingan Nilai Pengali T dan F terhadap ε_m pada AdaBoost M2

Ide awal pada Boosting adalah memberikan pembobotan lebih tinggi pada ciri yang tingkat kesalahannya tinggi dan tidak merubah bobot ciri yang sudah benar diklasifikasikan. Oleh karena itu, tesis ini mengajukan metode yang mengkombinasikan metode AdaBoost dengan AdaBoost M2 yang disebut metode multikelas Boosting termodifikasi, dengan cara menambahkan fungsi indikator $I(y_m(x_n) \neq t_n)$ pada persamaan untuk mengubah bobot AdaBoost M2 sehingga persamaan (3.11) berubah menjadi:

$$w_n^{m+1} = w_n^m \beta^{(1/2)(1+h_m(x_n,y_n)-h_m(x_n,y))(I(y_m(x_n)\neq t_n))} \qquad (3.13)$$

Dengan demikian, pada awalnya nilai bobot w_n diinisialisasi dengan nilai yang sama. Kemudian dilakukan iterasi dari 1 sampai M melakukan proses pelatihan dengan dengan meminimalkan fungsi kesalahan persamaan (3.8) dan menghitung *pseudo loss* ε_m untuk mengevaluasi kesalahan. Setelah itu dilakukan pembobotan ulang seperti persamaan (3.13). Berbeda dengan AdaBoost M2, pada perbaikan bobot untuk iterasi selanjutnya apabila hasil pembobotan sudah sesuai dengan y tujuan, bobot tidak akan dirubah lagi. Setelah iterasi selesai, barulah diprediksi model terakhir.

3.3.2 Implementasi Multikelas Boosting Termodifikasi pada Sistem Penghitung Pengunjung

Pada implementasi sistem rangka kerja [4] dan [13], kode-kode program yang dimodifikasi adalah sebagai berikut:

1. cvboost.cpp [haartraining.sln], berisi metode pelatihan pada sistem. Modifikasi yang dilakukan adalah penambahan metode pelatihan AdaBoost M2 dan multikelas Boosting termodifikasi.
2. cvtypes.h, berisi deklarasi *struct* yang digunakan pada rangka kerja. Ditambahkan beberapa variabel yang dibutuhkan pada kode agar dapat digunakan sebagai penanda multikelas.
3. cv.h [opencv.sln], berisi deklarasi fungsi-fungsi pada rangka kerja. Ditambahkan deklarasi fungsi cvHaarDetectObjects2 pada kode, yang nantinya diimplementasikan pada cvhaar.cpp.
4. cvhaar.cpp [opencv.sln], ditambahkan fungsi cvHaarDetectObjects2 yang digunakan untuk menerima pengklasifikasi dari 3 kelas dan mendeteksi obyek.
5. facedetect2.c [13], berisi kode untuk pendeteksi dan penghitung obyek. Kode ini dimodifikasi agar dapat menggunakan fungsi pendeteksi multikelas, mendeteksi pada zona penghitungan yang tepat, dan menampilkan hasil penghitungan.

BAB 4
SKENARIO UJICOBA

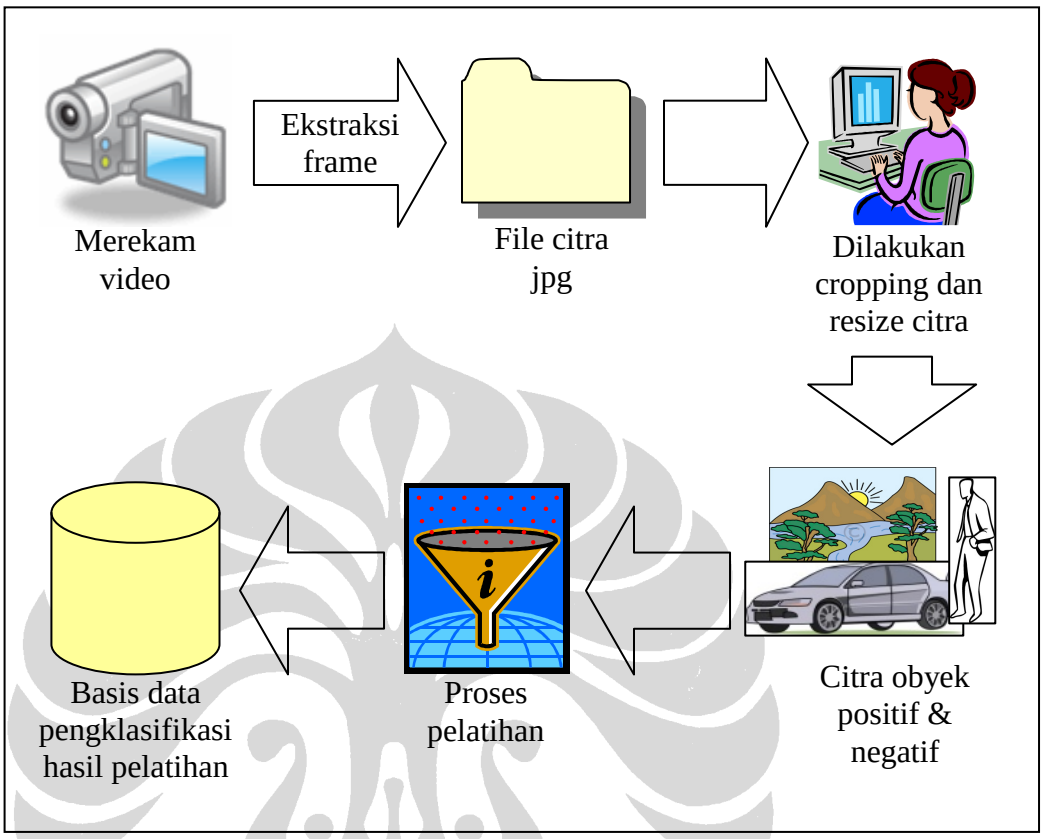
Bab ini membahas hal-hal yang berkaitan dengan ujicoba terhadap sistem penghitung pengunjung menggunakan metode multikelas Boosting termodifikasi. Pembahasan meliputi data ujicoba, lingkungan ujicoba, dan skenario ujicoba yang digunakan untuk menguji kinerja metode yang diusulkan.

4.1 Data Ujicoba

Data yang digunakan dalam tesis ini merupakan rekaman video di luar ruangan (*outdoor*) dengan pencahayaan yang cukup sehingga obyek pengunjung yang ingin dideteksi terlihat dengan jelas. Pengambilan video menggunakan handycam digital Panasonic VDR-D160 DVD Camcorder. Lokasi pengambilan video adalah di tempat parkir Fakultas Teknik Universitas Indonesia di Depok. Video yang direkam adalah rekaman yang telah diskenariokan oleh penulis agar dapat digunakan untuk ujicoba. Jumlah obyek yang direkam berjumlah tiga, yaitu satu obyek pejalan kaki, satu obyek pengendara motor, dan satu obyek mobil. Masing-masing obyek tersebut akan digunakan sebagai input data positif untuk pelatihan, sedangkan latar belakang tanpa obyek yang ingin dideteksi akan digunakan sebagai data negatif.

Setelah video direkam, video tersebut diekstrak per *frame*. Dari keseluruhan *frame*, diambil sembarang beberapa *frame* untuk masukan pelatihan. Citra tersebut kemudian dipotong (*crop*) dari latarnya, dengan ukuran 20 x 50 piksel untuk obyek pejalan kaki, 38 x 38 piksel untuk obyek pengendara motor, dan 50 x 25 piksel untuk obyek mobil. Ukuran tersebut digunakan karena adanya perbandingan skala tinggi dan lebar masing-masing obyek, dan agar perbedaan ciri Haar-*like* yang terdapat pada citra masih dapat dideteksi oleh sistem pelatihan. Apabila ukuran citra untuk pelatihan terlalu besar, waktu pelatihan akan jauh lebih lama dan terkadang muncul pesan kesalahan bahwa sistem tidak dapat menghitung data yang dimasukkan akibat penggunaan memori komputer yang terlalu besar. Sedangkan apabila ukurannya terlalu kecil, pola ciri Haarlike pada citra tidak dapat terdeteksi sehingga pelatihan tidak pernah selesai.

Berikut adalah alur proses mulai pengambilan video sampai penyimpanan basis data pelatihan.



Gambar 4.1 Alur Proses Pembuatan Data Pelatihan Dimulai dari Merekam Video sampai Hasil Pelatihan

Alur proses pembuatan data pelatihan dimulai dengan proses pengambilan rekaman video dimana obyek-obyek yang ingin dideteksi diatur sedemikian rupa agar bergerak dari sisi kanan video ke sisi kiri video. Hasil rekaman video tersebut diekstrak per *frame*-nya menjadi file-file citra dengan ekstensi bmp atau jpeg. Kemudian pada file-file citra yang berisi obyek positif dilakukan proses cropping dan penyekalaan untuk menghasilkan data citra positif yang mempunyai satu obyek pengunjung yang terletak tepat di tengah. Sedangkan file-file citra yang tidak berisi obyek positif digunakan sebagai data negatif. Data citra positif dan citra negatif kemudian dimasukkan ke dalam sistem pelatihan. Hasil pelatihan adalah suatu basis data pengklasifikasi yang dapat digunakan untuk membedakan apakah ada obyek positif di dalam video atau tidak.

Untuk pengujian, digunakan dua macam data yaitu data citra dan data video. Pengujian data video dilakukan dengan cara penghitungan manual, sedangkan pengujian data citra dilakukan secara otomatis menggunakan perangkat lunak. Data citra yang diujikan didapatkan dengan mengaplikasikan data citra positif pada citra negatif. Detail skenario pengujian akan dijelaskan pada sub bab berikut.

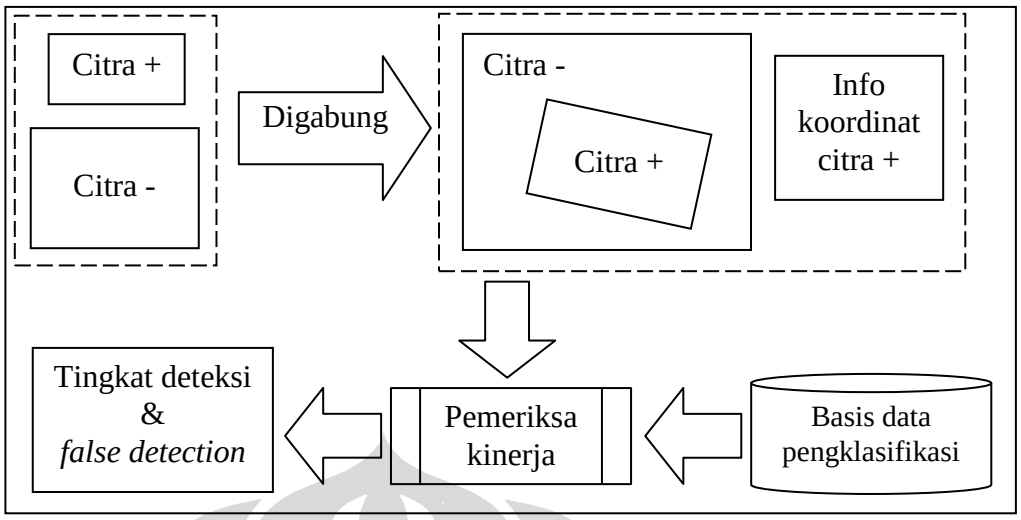
4.2 Skenario Ujicoba

Ujicoba dilakukan agar dapat menganalisis kinerja dari sistem penghitung pengunjung dengan metode multikelas Boosting termodifikasi. Lingkungan ujicoba pada penelitian ini penggunakan metode penghitungan manual dan juga perangkat lunak dengan bahasa pemrograman C [4]. Spesifikasi perangkat keras untuk ujicoba menggunakan komputer dengan prosesor Intel Core 2 Duo E6550 dan memori 2 Gbyte.

Sebelum melakukan analisis kinerja metode yang diusulkan, maka dilakukan perancangan skenario ujicoba terlebih dahulu.

4.2.1 Skenario ujicoba pertama: mengukur akurasi dari tiap pengklasifikasi hasil pelatihan dengan data citra

Skenario ujicoba yang pertama adalah untuk mengetahui tingkat akurasi masing-masing basis data pengklasifikasi hasil pelatihan. Pada skenario ini, setiap pengklasifikasi obyek yang dilatih diuji secara terpisah dan diperiksa hasilnya. Dari hasil pengujian akan dianalisa tingkat deteksi dan *false detection* pengklasifikasi, dimana tingkat deteksi menunjukkan berapa banyak obyek yang terdeteksi dengan benar dibandingkan dengan jumlah seluruh obyek ujicoba, dan *false detection* menunjukkan adanya suatu citra yang tidak berisi obyek yang ingin dideteksi, namun terdeteksi sebagai obyek yang diinginkan.



Gambar 4.2 Alur Perangkat Lunak Ujicoba Skenario Ujicoba Pertama, Mengukur Akurasi Tiap Pengklasifikasi

Untuk melakukan ujicoba, digunakan perangkat lunak yang dapat membantu menghitung kinerja dari pengklasifikasi. Alur perangkat lunak yang digunakan dapat dilihat pada gambar 4.2.

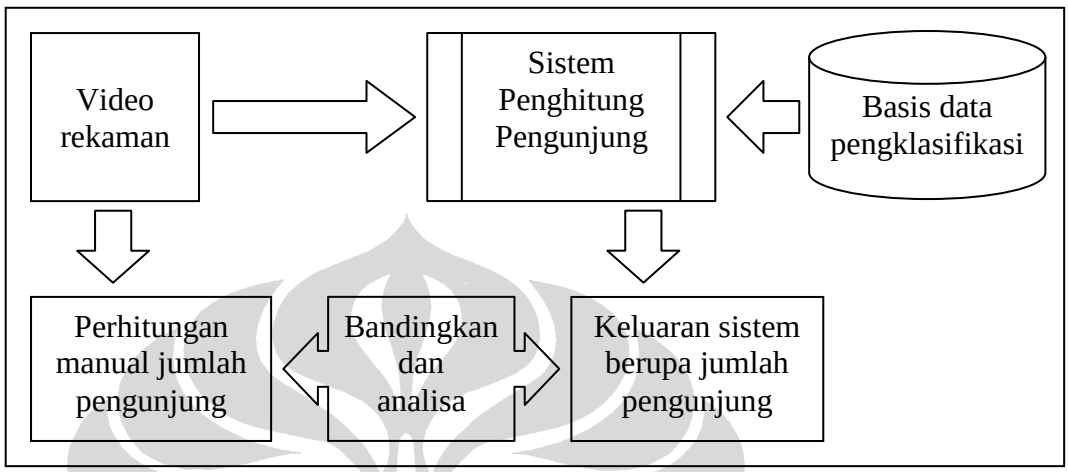
Untuk data ujicoba, disiapkan citra-citra positif dan citra-citra latar belakang sebagai citra negatif. Kemudian citra-citra tersebut digabungkan dengan batasan distorsi dan rotasi yang ditentukan. Hasil penggabungan tersebut adalah citra positif yang memiliki latar belakang citra negatif, beserta informasi koordinat letak citra positif tersebut. Kemudian basis data pengklasifikasi hasil pelatihan akan diaplikasikan ke citra gabungan untuk memeriksa kinerja pengklasifikasi tersebut. Keluaran dari proses ini adalah tingkat deteksi (*hit and miss rate*) dan *false detection* pengklasifikasi.

4.2.2 Skenario ujicoba kedua: mengukur akurasi sistem penghitung pengunjung dengan data video

Skenario ujicoba yang kedua adalah untuk mengetahui tingkat akurasi dari keseluruhan basis data pengklasifikasi beserta proses penjejakan dan penghitungan. Pada skenario ini, pengklasifikasi-pengklasifikasi obyek yang sudah dilatih dengan metode tertentu diimplementasikan ke sistem penghitung pengunjung. Hasil pengujian sistem penghitung pengunjung akan dibandingkan

dengan penghitungan pengunjung secara manual, kemudian akan dianalisa tingkat deteksi dari metode yang digunakan.

Alur dari ujicoba sistem penghitung pengunjung dapat dilihat pada gambar 4.3 berikut.

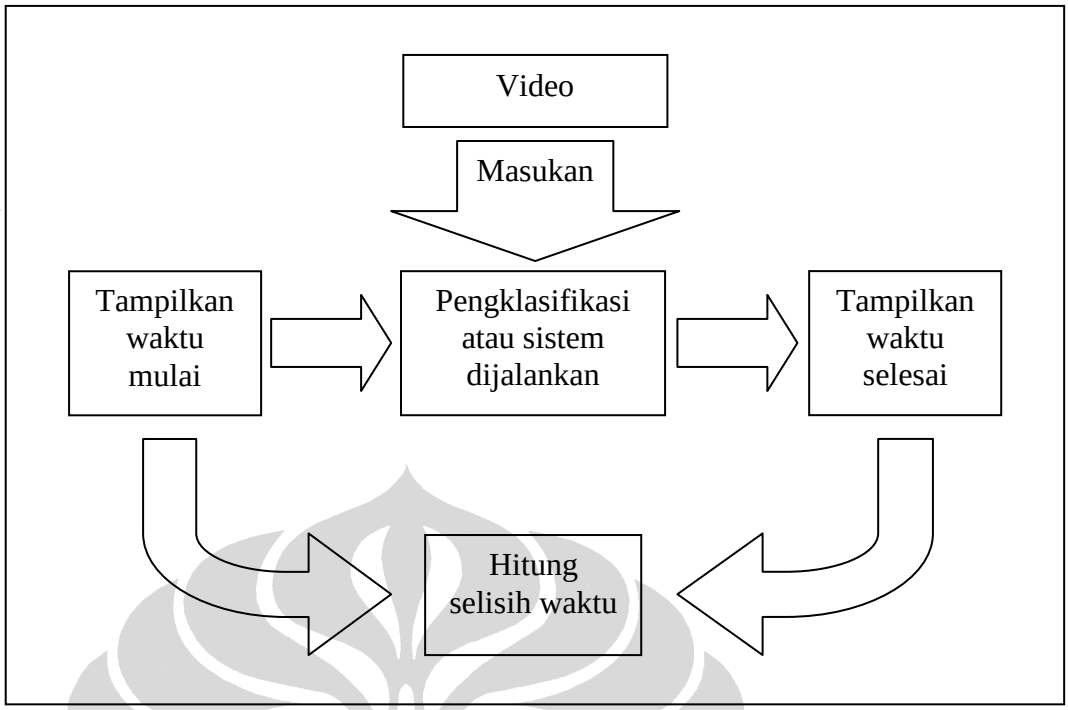


Gambar 4.3 Alur Skenario Ujicoba Kedua, Mengukur Akurasi Sistem Penghitung Pengunjung

Untuk data ujicoba, disiapkan video rekaman pengunjung yang ingin dihitung. Video rekaman itu dijadikan masukan untuk sistem penghitung pengunjung, yang kemudian menggunakan basis data pengklasifikasi hasil pelatihan untuk mendeteksi, menjejaki, dan menghitung jumlah pengunjung yang ada dalam video tersebut. Hasil dari sistem penghitung pengunjung adalah jumlah pengunjung yang terhitung dari video. Hasil ini kemudian dibandingkan dengan hasil penghitungan pengunjung secara manual dan dianalisa.

4.2.3 Skenario ujicoba ketiga: mengukur kecepatan tiap pengklasifikasi dan sistem penghitung pengunjung dengan data video

Skenario ujicoba yang ketiga adalah untuk mengetahui rata-rata kecepatan dari tiap pengklasifikasi dan keseluruhan sistem. Pada skenario ini, dibuat penghitung waktu untuk menghitung berapa lama waktu yang diperlukan oleh tiap pengklasifikasi atau seluruh sistem untuk mendeteksi video berdurasi 82 detik. Alur ujicoba dapat dilihat pada gambar 4.4.



Gambar 4.4 Alur Skenario Ujicoba Ketiga, Mengukur Kecepatan Tiap Pengklasifikasi dan Keseluruhan Sistem

BAB 5
HASIL UJICOBA DAN ANALISISNYA

Bab ini membahas hal-hal yang berkaitan dengan hasil ujicoba yaitu keluaran dari skenario ujicoba dan analisa kinerja pengklasifikasi dan sistem penghitung pengunjung dengan metode multikelas Boosting termodifikasi, dibandingkan dengan metode asalnya.

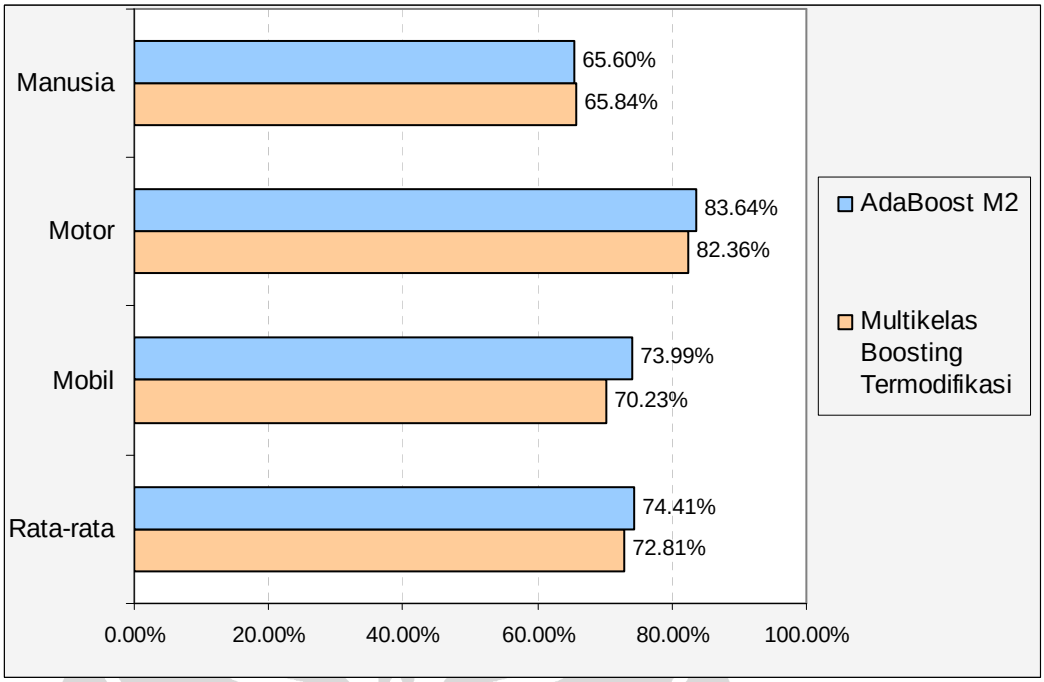
5.1 Hasil Ujicoba

Ujicoba dilakukan berdasarkan dua skenario yang sudah dirancang pada bab sebelumnya. Ujicoba dilakukan agar dapat mengukur kinerja dari sistem penghitung pengunjung dengan metode multikelas Boosting termodifikasi, dibandingkan dengan metode AdaBoost M2. Pada sub bab selanjutnya dibahas hasil ujicoba pada dua skenario.

5.1.1 Hasil ujicoba skenario pertama: mengukur akurasi dari tiap pengklasifikasi hasil pelatihan dengan data citra

Skenario pertama bertujuan untuk mengukur akurasi dari tiap basis data pengklasifikasi hasil pelatihan dengan ujicoba data citra. Ujicoba dilakukan dengan memasukkan citra gabungan yang mengandung citra positif beserta koordinatnya ke perangkat lunak, yang kemudian memeriksa ada tidaknya obyek dengan mengaplikasikan basis data pengklasifikasi hasil pelatihan. Keluaran dari perangkat lunak tersebut adalah tingkat deteksi dan *false detection* dari basis data pengklasifikasi yang diuji.

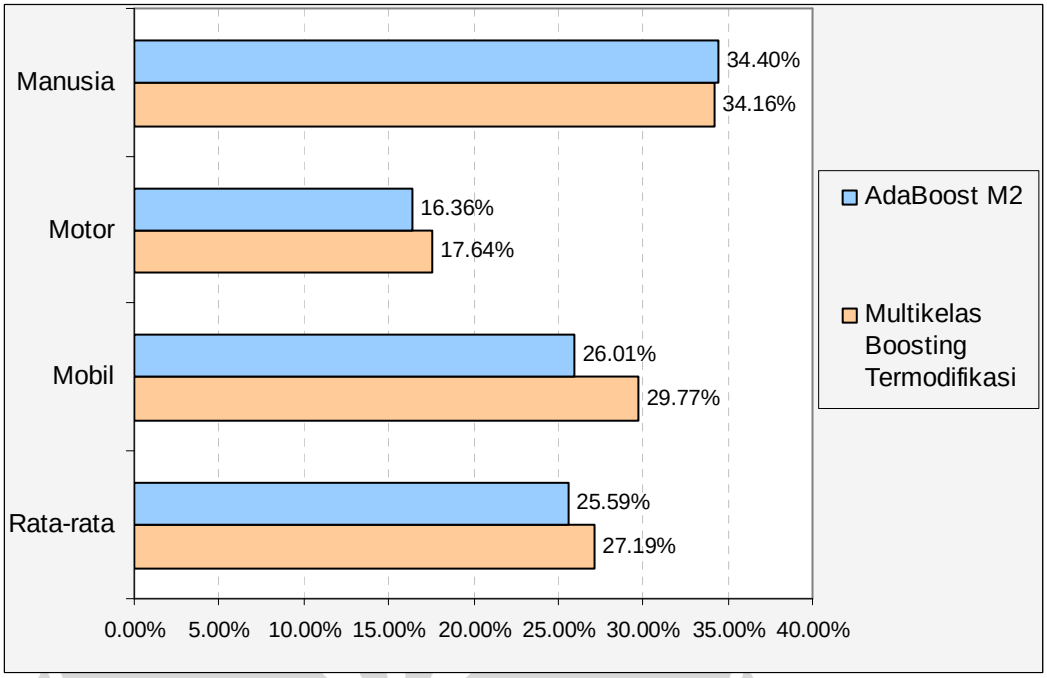
Gambar 5.1, gambar 5.2, dan gambar 5.3 berikut secara berurutan menunjukkan grafik persentase tingkat deteksi (*hit rate*), tidak terdeteksi (*missed*) dan *false detection* dari setiap tipe pengklasifikasi yang dilatih, dibandingkan terhadap jumlah sampel pengujian. Hasil dari kedua metode pengklasifikasi ditampilkan agar dapat dibandingkan. Selain persentase tingkat deteksi dari setiap pengklasifikasi, juga disertakan rata-rata dari tiap metode pengklasifikasi.



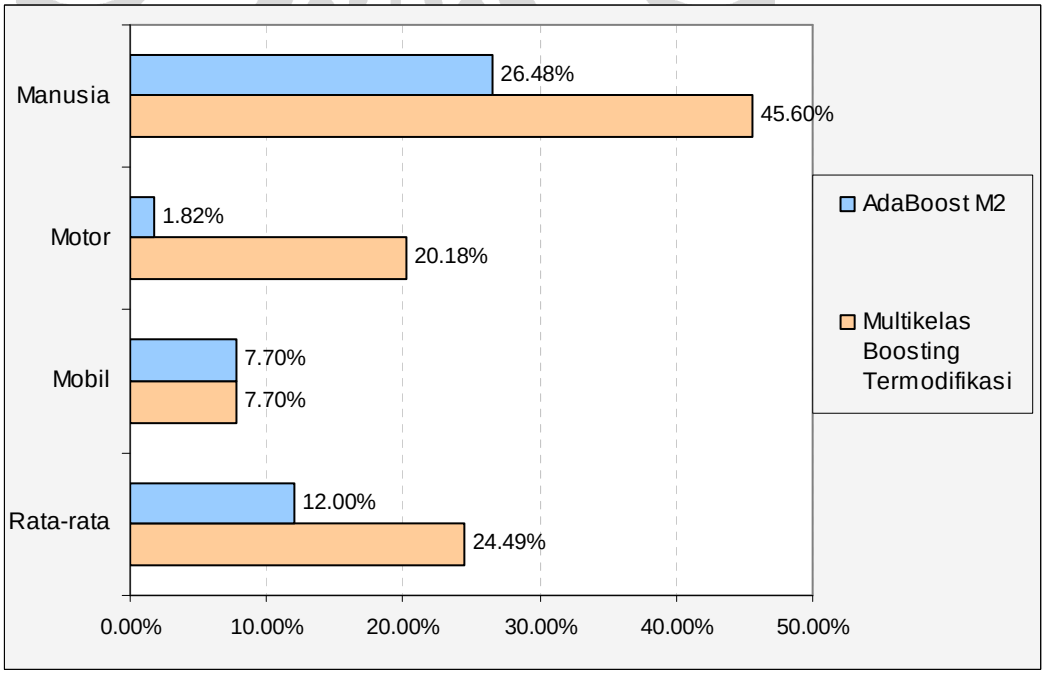
Gambar 5.1 Grafik Hasil Ujicoba, Persentase Tingkat Deteksi (*hit rate*), lebih tinggi lebih baik

Dari gambar 5.1 di atas, terlihat bahwa rata-rata tingkat deteksi (*hit rate*) untuk metode AdaBoost M2 74.41% dan untuk metode multikelas Boosting termodifikasi 72.81%. Metode yang diusulkan memiliki tingkat deteksi lebih tinggi 0.24% untuk obyek pejalan kaki (manusia), namun lebih rendah 1.28% dan 3.76% untuk obyek motor dan mobil. Terlihat bahwa tingkat deteksi dari metode multikelas Boosting termodifikasi dapat bersaing dengan metode AdaBoost M2. Kebalikan dari tingkat deteksi adalah tingkat tidak terdeteksi (*missed*), dapat dilihat pada gambar 5.2.

Selain tingkat deteksi, juga diukur tingkat *false detection*. *False detection* atau deteksi palsu menandakan adanya citra negatif atau bukan obyek yang terdeteksi sebagai obyek atau citra positif. Perbandingan hasil ujicoba tingkat *false detection* dari kedua metode dapat dilihat pada gambar 5.3. Untuk obyek manusia dan motor, metode yang diusulkan mendapat tingkat *false detection* yang lebih tinggi dibandingkan dengan metode AdaBoost M2. Sedangkan untuk obyek mobil, kedua metode mendapatkan hasil yang sama. Rata-rata tingkat *false detection* untuk metode yang diusulkan adalah 24.49%, sedangkan rata-rata untuk metode AdaBoost M2 adalah 12%.



Gambar 5.2 Grafik Hasil Ujicoba, Persentase yang Tidak Terdeteksi (*missed*), lebih rendah lebih baik



Gambar 5.3 Grafik Hasil Ujicoba, Persentase Tingkat *False detection*, lebih rendah lebih baik

Data lengkap hasil ujicoba jumlah dan persentase dari tingkat deteksi, tidak terdeteksi, dan *false detection* dapat dilihat pada tabel 5.1 berikut.

Tabel 5.1 Hasil ujicoba tingkat deteksi dan *false detection*

Metode	Obyek		Hits		Missed		False Detect.	
		Total	#	%	#	%	#	%
AdaBoost M2	Manusia	1250	820	65.60%	430	34.40%	331	26.48%
	Motor	1100	920	83.64%	180	16.36%	20	1.82%
	Mobil	1169	865	73.99%	304	26.01%	90	7.70%
	Rata2	1173	868.3	74.41%	304.7	25.59%	147	12.00%
Multikelas Boosting Termodifikasi	Manusia	1250	823	65.84%	427	34.16%	570	45.60%
	Motor	1100	906	82.36%	194	17.64%	222	20.18%
	Mobil	1169	821	70.23%	348	29.77%	90	7.70%
	Rata2	1173	850	72.81%	323	27.19%	294	24.49%

5.1.2 Hasil ujicoba skenario kedua: mengukur akurasi sistem penghitung pengunjung dengan data video

Skenario kedua bertujuan untuk mengukur tingkat akurasi dari keseluruhan pengklasifikasi beserta proses penjejakan dan penghitungan. Ujicoba dilakukan dengan memberikan video masukan ke sistem penghitung pengunjung. Keluaran dari sistem penghitung pengunjung adalah jumlah pengunjung pejalan kaki, pengendara motor, dan mobil.

Dengan penghitungan manual rekaman video, ada total 17 pengunjung yang memasuki daerah dari kanan ke kiri, yang terdiri dari 3 pejalan kaki, 10 pengendara motor, dan 4 mobil. Selain itu, juga ada percobaan dimana pengunjung mendekati zona penghitungan atau pintu masuk di sebelah kiri, namun berputar kembali ke sebelah kanan, ataupun bergerak dari kiri ke kanan (berlawanan arah).

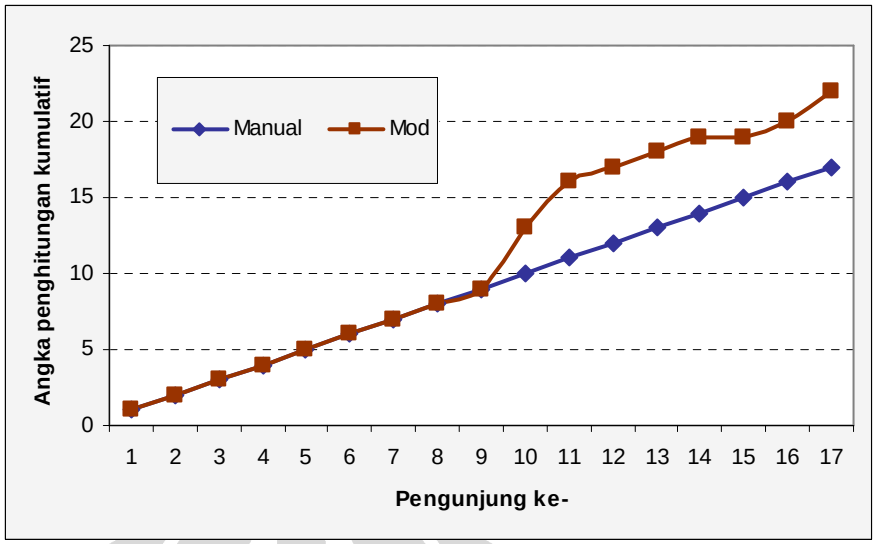
Dalam proses penghitungan sistem pengunjung dengan metode AdaBoost M2, terhitung total 22 pengunjung, yang terdiri dari 7 pejalan kaki, 10 pengendara motor, dan 5 mobil. Tabel 5.2 merangkum hasil penghitungan dari metode AdaBoost M2. Pengendara motor terhitung dengan benar. Kesalahan penghitungan pada mobil akibat terputusnya proses deteksi dan penjejakan obyek sehingga satu obyek terhitung di zona penghitungan beberapa kali dan sistem tidak dapat menjejaki obyek yang lewat terlalu cepat. Sedangkan kesalahan

penghitungan pejalan kaki diakibatkan adanya obyek pejalan kaki yang berputar kembali keluar dari pintu masuk (dari arah kiri ke kanan), serta putusnya proses deteksi obyek sehingga mengakibatkan penjejakan terputus dan terhitung beberapa kali.

Tabel 5.2 Hasil Ujicoba Sistem Penghitung Pengunjung Metode AdaBoost M2

No.	Keterangan	Deskripsi
1	Motor	Terdeteksi dengan benar
2	Pejalan kaki	Terdeteksi dengan benar
3	Motor	Terdeteksi dengan benar
4	Mobil	Terdeteksi dengan benar
5	Motor	Terdeteksi dengan benar
6	Pejalan kaki	Terdeteksi dengan benar
7	Motor	Terdeteksi dengan benar
8	Motor berhenti dan berjalan kembali	Terdeteksi dengan benar
9	Motor melewati mobil	Terdeteksi dengan benar
-	Pejalan kaki berputar kembali (tidak wajar)	Terjadi kesalahan penghitungan, penjejakan terputus dan terhitung berulang kali
10	Motor berhenti dan melewati mobil	Terdeteksi dengan benar
11	Mobil berhenti dan berjalan kembali	Terdeteksi dengan benar, namun penjejakan terputus dan terhitung berulang kali
12	Motor	Terdeteksi dengan benar
13	Mobil	Terdeteksi dengan benar
14	Motor	Terdeteksi dengan benar
15	Mobil	Terjadi kesalahan penghitungan, tidak terdeteksi akibat terlalu cepat
16	Pejalan kaki	Terdeteksi dengan benar
-	Pejalan kaki berlawanan arah (tidak wajar)	Terjadi kesalahan penghitungan, terhitung akibat putusnya penjejakan pada zona penghitungan
17	Motor	Terdeteksi dengan benar

Baris pada tabel dengan latar belakang putih menandakan bahwa obyek terdeteksi dengan benar. Latar belakang kuning menunjukkan bahwa obyek bergerak secara tidak wajar dan seharusnya tidak terdeteksi, tetapi terjadi kesalahan penghitungan. Sedangkan latar belakang merah jingga menandakan bahwa obyek bergerak secara wajar, namun terjadi kesalahan penghitungan.



Gambar 5.4 Perbandingan Penghitungan Manual dengan Sistem Penghitung Metode AdaBoost M2

Gambar 5.4 menggambarkan perbandingan antara angka penghitungan kumulatif yang menggunakan sistem penghitung metode AdaBoost M2 dengan penghitungan manual. Dari tabel 5.2 dan gambar 5.4 di atas, terlihat bahwa ada kecenderungan sistem penghitung melakukan kesalahan apabila terjadi pergerakan pengunjung yang tidak sesuai dengan asumsi pergerakan normal.

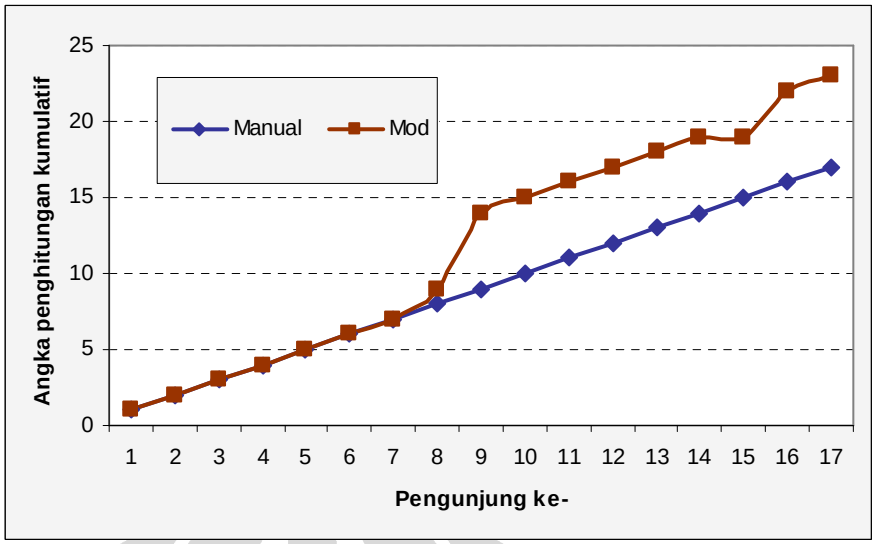
Sedangkan untuk proses penghitungan sistem pengunjung dengan metode Multikelas Boosting Termodifikasi, terhitung total 23 pengunjung, yang terdiri dari 9 pejalan kaki, 10 pengendara motor, dan 4 mobil (untuk mobil terjadi *false detection* dan tidak terdeteksi, sehingga jumlahnya kebetulan tepat). Tabel 5.3 berikut merangkum hasil penghitungan dari metode multikelas Boosting termodifikasi.

Penyebab kesalahan penghitungan serupa dengan ujicoba pada metode AdaBoost M2, yaitu akibat terputusnya proses deteksi dan penjejakan obyek sehingga satu obyek terhitung di zona penghitungan beberapa kali dan sistem tidak dapat menjejaki obyek yang lewat terlalu cepat. Pada pengunjung ke-8, terjadi false detection dimana motor terdeteksi dengan benar, namun juga terdeteksi sebagai mobil.

Tabel 5.3 Hasil Ujicoba Sistem Penghitung Pengunjung Metode Multikelas Boosting Termodifikasi

No.	Keterangan	Deskripsi
1	Motor	Terdeteksi dengan benar
2	Pejalan kaki	Terdeteksi dengan benar
3	Motor	Terdeteksi dengan benar
4	Mobil	Terdeteksi dengan benar
5	Motor	Terdeteksi dengan benar
6	Pejalan kaki	Terdeteksi dengan benar
7	Motor	Terdeteksi dengan benar
8	Motor berhenti dan berjalan kembali	Terdeteksi, namun terjadi kesalahan penghitungan akibat *false detection*
9	Motor melewati mobil	Terdeteksi dengan benar
-	Pejalan kaki berputar kembali (tidak wajar)	Terjadi kesalahan penghitungan, penjejakan terputus dan terhitung berulang kali
10	Motor berhenti dan melewati mobil	Terdeteksi dengan benar
11	Mobil berhenti dan berjalan kembali	Terdeteksi dengan benar
12	Motor	Terdeteksi dengan benar
13	Mobil	Terdeteksi dengan benar
14	Motor	Terdeteksi dengan benar
15	Mobil	Terjadi kesalahan penghitungan, tidak terdeteksi akibat terlalu cepat
16	Pejalan kaki	Terdeteksi dengan benar
-	Pejalan kaki berlawanan arah (tidak wajar)	Terjadi kesalahan penghitungan, terhitung akibat putusnya penjejakan pada zona penghitungan
17	Motor	Terdeteksi dengan benar

Gambar 5.5 menggambarkan perbandingan antara angka penghitungan kumulatif yang menggunakan sistem penghitung metode multikelas Boosting termodifikasi dengan penghitungan manual. Pada metode ini, juga terlihat bahwa ada kecenderungan sistem penghitung melakukan kesalahan apabila terjadi pergerakan pengunjung yang tidak sesuai dengan asumsi pergerakan normal. Namun untuk pergerakan pengunjung yang normal, hasil sistem penghitung mendekati aslinya, yaitu terhitung yaitu 17 pengunjung dari 17 pengunjung, dimana ada kesalahan penghitungan obyek mobil, namun kebetulan jumlahnya sesuai. Jumlah pengunjung sebenarnya yang terhitung dengan benar adalah 15 pengunjung.



Gambar 5.5 Perbandingan Penghitungan Manual dengan Sistem Penghitung Metode Multikelas Boosting Termodifikasi

5.1.3 Hasil ujicoba skenario ketiga: mengukur kecepatan tiap pengklasifikasi dan sistem penghitung pengunjung dengan data video

Skenario ketiga bertujuan untuk mengukur kecepatan dari tiap pengklasifikasi yang dilatih, serta kecepatan dari keseluruhan sistem penghitung pengunjung. Ujicoba dilakukan dengan memberikan video masukan berdurasi 82 detik untuk dideteksi beberapa kali, kemudian lamanya waktu deteksi keseluruhan video dihitung dan dirata-ratakan. Perbandingan rata-rata waktu deteksi video dapat dilihat pada tabel 5.4 berikut.

Tabel 5.4 Hasil Ujicoba Rata-rata Kecepatan Tiap Pengklasifikasi dan Sistem Penghitung Pengunjung (dalam satuan detik)

Metode	Manusia (20x50)	Motor (38x38)	Mobil (50x25)	Seluruh Sistem
AdaBoost M2	39.2067	99.92	71.89	217.69
M.B. Termodifikasi	38.7367	100.55	71.63	213.10

Terlihat bahwa ukuran citra pelatihan mempengaruhi waktu deteksi tiap pengklasifikasi. Pengklasifikasi manusia dengan luas citra pelatihan 1000 piksel memiliki waktu rata-rata lebih cepat daripada citra mobil dengan luas 1250 piksel

dan citra motor dengan luas 1444 piksel. Dari kedua metode yang diujikan, keduanya memiliki rata-rata kecepatan yang hampir sama untuk satu jenis obyek.

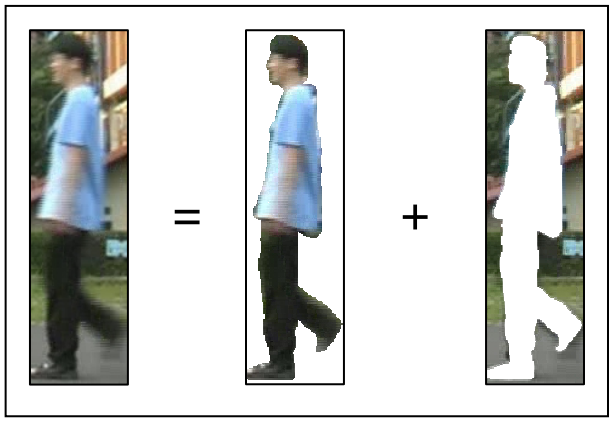
5.2 Analisis Hasil Ujicoba

Dari hasil ujicoba pertama, terlihat bahwa jumlah citra positif yang tepat terdeteksi jauh lebih banyak daripada tingkat citra positif yang tidak terdeteksi untuk semua jenis obyek. Perbandingan rata-rata menunjukkan bahwa tingkat deteksi metode multikelas Boosting lebih rendah 1.6% dibandingkan tingkat deteksi metode AdaBoost M2. Selain itu, terlihat bahwa tingkat false detection untuk metode multikelas Boosting termodifikasi rata-rata dua kali lipat dibandingkan metode AdaBoost M2. Hal ini disebabkan karena modifikasi penambahan fungsi indikator seperti pada formula 3.13:

$$w_n^{m+1} = w_n^m \beta^{(1/2)(1+h_m(x_n,y_n)-h_m(x_n,y))(I(y_m(x_n)\neq t_n))}$$

Dimana apabila ciri salah terklasifikasi, maka perubahan bobot akan memperhitungkan faktor kesalahan dari kelas yang ingin dideteksi dan faktor kesalahan deteksi dari kelas lainnya. Namun apabila ciri sudah benar terklasifikasi, ada faktor kesalahan deteksi kelas lainnya akan hilang karena fungsi indikator. Hal ini menyebabkan rata-rata akurasi pada metode yang diajukan menurun dan tingkal *false detection* menjadi lebih tinggi.

Tingkat *false detection* rata-rata untuk kedua metode cukup tinggi disebabkan karena batasan bentuk *cropping* berbentuk persegi panjang pada citra positif untuk pelatihan sehingga masih ada gambar latar yang turut dianggap sebagai bagian dari citra positif. Contoh cropping gambar yang dilatihkan dapat dilihat pada gambar 5.6. Akibatnya ada gambar latar yang terdeteksi sebagai obyek sehingga mengakibatkan tingkat *false detection* yang tinggi. Ketika dicoba menghilangkan latar belakang pada citra positif, basis data pengklasifikasi yang dihasilkan tidak dapat mendeteksi obyek. Sedangkan untuk obyek mobil, didapatkan tingkat *false detection* yang sama. Hal ini disebabkan karena obyek mobil yang dilatihkan memiliki tekstur yang lebih sederhana dibandingkan tekstur manusia ataupun motor.



Gambar 5.6 Ilustrasi Gambar, Citra Positif yang Dilatihkan Sebenarnya adalah Citra Positif yang Diperlukan Ditambah dengan Sisa Latar Belakang

Dari hasil ujicoba kedua di tabel 5.2 dan 5.3 terlihat bahwa hasil penghitungan untuk pergerakan obyek yang normal dari kanan ke kiri cukup baik dengan tingkat kesalahan 2 obyek dari 17 obyek yang bergerak normal (tidak mengikut sertakan kasus obyek bergerak berlawanan arah). Hal ini disebabkan karena *false detection* dapat diatasi dengan pengaturan posisi kamera dan zona penghitungan yang tepat, sehingga jumlah *false detection* yang terjadi dapat ditekan. Namun demikian, tetap sempat terjadi *false detection* sehingga perhitungan yang seharusnya tidak bertambah menjadi bertambah. Juga terdapat kasus dimana obyek tidak terhitung karena melewati daerah terlalu cepat (pengunjung 15, mobil). Hal ini dapat diatasi dengan cara memperjauh posisi kamera agar pergerakan obyek yang tertangkap tidak terlalu cepat, atau menambahkan mekanisme untuk memperlambat kendaraan pada zona pengawasan sehingga kendaraan dapat terhitung dengan benar.

Untuk pergerakan tidak wajar yang terjadi dua kali pada rekaman, keduanya mengakibatkan perhitungan bertambah dimana seharusnya tidak bertambah. Hal ini diakibatkan karena putusnya penjejakan di daerah zona penghitungan. Apabila basis data dapat ditingkatkan akurasinya dan berhasil menjejaki dengan sempurna, maka kedua pergerakan tidak wajar tersebut tidak akan mengakibatkan salah penghitungan.

Dari hasil ujicoba ketiga, terlihat bahwa ukuran pelatihan pengklasifikasi berbanding lurus dengan lamanya waktu deteksi. Namun hampir tidak ada perbedaan lama waktu deteksi diantara kedua metode tersebut.

Dari ketiga skenario ujicoba tersebut, terlihat bahwa fungsi indikator yang ditambahkan pada metode multikelas Boosting termodifikasi tidak memiliki pengaruh signifikan dari segi akurasi maupun kecepatan dibandingkan metode AdaBoost M2 yang merupakan basisnya. Tingkat deteksi diantara kedua metode tersebut agak fluktuatif dan mirip, terlihat pada gambar 5.1 dimana metode multikelas Boosting termodifikasi lebih unggul pada deteksi obyek manusia, sedangkan AdaBoost M2 lebih unggul untuk deteksi obyek motor dan mobil.