# PROPOSAL
# PENELITIAN MANDIRI

## JUDUL :

**PENGEMBANGAN SISTEM DETEKSI HELM PADA PENGENDARA SEPEDA MOTOR MENGGUNAKAN METODA *HOUGH CIRCLE TRANSFORM* (HCT)**


**PENGUSUL :**

**ASEP NANA HERMANA  004221166003**



**PROGRAM STUDI INFORMATIKA**
**FAKULTAS TEKNOLOGI INDUSTRI**
**INSTITUT TEKNOLOGI NASIONAL**

**2022**

**Halaman Pengesahan**

| | | |
|---|---|---|
| Judul Penelitian | : | PENGEMBANGAN SISTEM DETEKSI HELM PADA PENGENDARA SEPEDA MOTOR MENGGUNAKAN METODA *HOUGH CIRCLE TRANSFORM* (HCT) |
| Kode/ Nama Rumpun Ilmu | : | 140312 / Teknologi Informasi dan Komunikasi - Pengolahan Sinyal Multimedia -- Visi Komputer |
| **Ketua Peneliti** | : | |
| a. Nama Lengkap | : | Asep Nana Hermana |
| b. NIDN | : | 0422116603 |
| c. Jabatan Fungsional | : | Lektor / IIIC |
| d. Program Studi | : | Informatika |
| e. Nomor HP | : | 08122125874 |
| f. Alamat surel (e-mail) | : | asep_nana@gmail.com |
| **Anggota-1** | | |
| a. Nama lengkap | : | Uung Ungkawa |
| b. NIDN | : | 0411105902 |
| c. Program Studi | | Informatika |
| **Anggota-2** | | |
| a. Nama lengkap | : | |
| b. NIDN | : | |
| c. Program Studi | | |

Mengetahui,
Ketua Prodi Informatika

( Lisa Kristiana, Dr.sc., MT)
NIDN : 0425107503

Bandung, 14 September 2022
Ketua tim penyusun

(Asep Nana Hermana., MT)
NIDN : 0422116603

Disahkan oleh :

Dekan Fakultas Teknologi Industri Itenas

(Jono Suhartono, S.T., M.T., Ph.D.)
NIDN : 0406017801

Ketua LP2M Itenas

Iwan Juwana, S.T., M.EM., Ph.D.
NIDN : 0403017701

# DAFTAR ISI

# DAFTAR TABEL

# BAB I
# PENDAHULUAN

## 1.1 Latar Belakang

HCT (Hough Circle Transform) adalah teknik yang paling umum digunakanuntuk mendeteksi objek yang berbentuk kurva seperti garis, lingkaran, elips dan parabola. (Auliannisa, Suratman S.T., & Rizal ST., 2017). HCT merupakan metodeyang digunakan dalam pengolahan citra digital dalam tahap ekstrasksi fitur dan dapat digunakan untuk pendeteksian bentuk lingkaran pada objek citra.

Identifikasi adalah penentuan dan pemastian identitas orang yang hidup maupun orang mati berdasarkan ciri khas yang terdapat pada orang tersebut. (Septadina, 2015). Helm sebagai salah satu *safety gear* bagi pengendara sepeda motor memang masih sering diabaikan. Akibat banyak pengendara yang abai, angka kecelakaan pengendara motor tak gunakan helem juga cukup tinggi. (Mufrod, 2018). Setiap orang yang mengemudikan Sepeda Motor dan PenumpangSepeda Motor wajib mengenakan helm yang memenuhi standar nasional Indonesia.(Undang-Undang Lalu Lintas Angkutan Jalan Tahun 2009). Kewajiban pengguna jalan dalam berlalu lintas yaitu menaati pertaturan yang berlaku, salah satunya yaitupengendara sepeda motor diwajibkan menggunakan helm. Pengguna sepeda motoryang melanggar peraturan tersebut akan mendapatkan peringatan satau sanksi, tetapi masih banyak pengguna sepeda motor yang didapati tidak menggunakan helm dijalan dikarenakan tidak terpantau oleh petugas atau hal lainnya.

Sebuah penelitian identifikasi penggunaan helm pada pengendara sepeda motor dilakukan dengan memanfaatkan teknologi pengolahan citra digital. Bagian utama pada penelitian ini terletak pada seberapa akurat sistem dapat melakukan pendeteksian objek pengendara sepeda motor yang menggunakan helm dengan

menggunakan algoritma HCT (*Hough Circle Transform*) untuk melakukan deteksihelm sehingga dapat membedakan pengendara sepeda motor yang menggunakan helm dan tidak menggunakan helm. Keluaran yang dihasilkan yaitu objek citra bukan helm dan peringatan. Penelitian ini dapat membantu mengidentifikasi pelanggaran yang dilakukan pengguna sepeda motor di jalan.

## 1.2 Rumusan masalah dan ruang lingkup

Dari berbagai pelanggaran, salah satu yang banyak ditemukan dalam razia lalu lintas, banyak pengendara tidak memakai helm. (Christina, 2017). Salah satu solusi alternatif dalam meminimalisir pelanggaran jalan adalah dengan menerapkandeteksi pelanggaran secara visual. (Setiawan, Setyawan, Alam, & Sulistiyanti, 2018). Di jalan-jalan kota Bandung, masih terdapat banyak pengendara sepeda motor yang tidak menggunakan helm. Oleh karena itu, diperlukan suatu sistem untuk memberikan peringatan terhadap pengendara yang melanggar aturan tersebut. Dari uraian diatas maka penelitian ini memiliki rumusan masalah sebagaiberikut :

a. Bagaimana sistem dapat mendeteksi pengendara sepeda motor menggunakan algoritma *Haar Cascade Classifier*.
b. Bagaimana sistem dapat mengidentifikasi penggunaan helm pada pengendara sepeda motor menggunakan algoritma HCT (*Hough Circle Transform*) yang kemudian akan menyeleksi pengendara sepeda motor yangtidak menggunakan helm.

## 1.3 Tujuan

Dalam penelitian yang dilakukan, dibatasi ruang lingkup yang akan dibahasyaitu sebagai berikut :

a. Data set video yang digunakan hanya diambil beberapa contoh dari jalan-jalan di kota bandung.
b. Proses pengambilan data set video dilakukan secara manual denganpencahayaan yang stabil.
d. Proses pengambilan data set video dilakukan dengan jarak, arah dan sudutyang telah ditentukan.
e. Proses pengambilan data set video ada kondisi tidak hujan serta pada pagi,siang atau sore hari dan tidak malam hari.
f. Proses pengambilan data set video objek kendaraan dan pengendara sepedamotor tidak berhimpit dengan kendaraan lain.
g. Background diasumsikan dalam keadaan statis.
h. Penelitian ini tidak melihat warna objek hanya melihat bentuk objek.
i. Metode yang digunakan untuk mendeteksi pengendara sepeda motor denganmenggunakan algoritma *Haar Cascade Classifier*.

j. Metode yang digunakan untuk mendeteksi penggunaan helm padapengendara sepeda motor menggunakan algoritma *Hough Circle Transform*.

### 1.4 Sistematika Penulisan

Sistem penulisan yang digunakan dalam pembuatan laporan ini adalah sebagai berikut :

BAB I PENDAHULUAN

Bab ini menjelaskan latar belakang, rumusan masalah, ruang lingkup, tujuan, metode penelitian, tinjauan pustaka, kontribusi penelitian dan sistematikapenulisan.

BAB II LANDASAN TEORI

Bab ini menjelaskan mengenai teori – teori yang digunakan dalam pembuatan rancang bangun sistem identifikasi terhadap kendaraan yang berhenti melewati garis marka di persimpangan jalan.

BAB III PERANCANGAN

Pada bagian ini dipaparkan metode penelitian, dalam hal ini merupakan perancangan uraian dari penelitian yang diusulkan. Pada bagian ini terdapat penjelasan mengenai proses perancangan terhadap pembangunan sistem yang dibuat.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Pada bagian sub-bab ini disajikan hasil rancangan yang diajukan. Pada bagian ini disajikan seperti potongan program yang di dekatkan dengan *flowchart*, model sistem dan sebagainya. Pada sub-bab pengujian disajikan proses pencapaian hasil penelitian berupa pengujian dari hasil implementasi yang telah dilakukan.

BAB V PENUTUP

Pada sub-bab ini disajikan kesimpulan dan saran dari penelitian yang telah dilakukan dan diuji.

# BAB II
# LANDASAN TEORI

## 2.1. Pengolahan Citra Digital

Pengolahan citra adalah pemrosesan citra, khususnya dengan menggunakankomputer, menjadi citra yang kualitasnya lebih baik. Pengolahan Citra bertujuan memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau mesin. (Herlambang, Nurhayati, & Martono, 2016). Pengolahan citra digital merupakan suatu proses untuk representasi dan manipulasi citra yang diproses dengan menggunakan perangkat digital atau komputer, baik citra diam maupun citra bergerak. Masukan dari pengolahan citra digital yaitu citra dan keluaran dari dapatberupa citra atau sekumpulan karakteristik atau parameter yang berhubungan dengan citra.

Definisi yang lebih luas, pengolahan citra digital juga mencakup semua datadua dimensi. Citra digital merupakan barisan bilangan nyata maupun kompleks yang diwakilkan oleh bit-bit tertentu. Pengolahan citra memiliki beberapa fungsi, diantaranya :

1. Sebagai proses memperbaiki kualitas citra agar mudah diinterpretasi oleh manusia atau komputer.
2. Sebagai proses mentrasformasikan citra menjadi citra lain. Salah satu contohnya yaitu pemampatan citra (*image compression*) Sebagai proses awal (*preprocessing*) dari *computer vision*.

## 2.2. Frame Rate

*Frame Rate* adalah frekuensi dimana gambar (*frame*) berturut-turut ditampilkan. Istilah ini berlaku sama untuk film, video kamera, kompter grafis, dansistem motion capture (Andrew, Buliali, & Wijaya, 2017). Frame rate merupakan ukuran kecepatan *frame* atau gambar yang ditunjukkan per detiknya,satuan dari *frame rate* yaitu *frame per second* (fps).

## 2.3. Ruang Warna RGB

Ruang warna RGB adalah warna utama yang dimiliki citra dengan tiga warna primer yaitu red, green, dan blue. Rentang nilai yang dimiliki citra RGB dalam setiap piksel citra adalah 0 sampai dengan 225. (Jandrisno, 2017). Jika masing-masing warna dari RGB (*R ed Green Blue*) memiliki range 0 - 255, maka totalnya adalah 2553 = 16.581.375 (16 K) variasi warna berbeda pada gambar. Gambar yang memiliki jumlah bit yang diperlukan untuk setiap pixel disebut juga gambar-bit warna. Merah (*Red*), Hijau (*Green*) dan Biru (*Blue*) merupakan warna dasar yang dapat diterima

oleh mata manusia. Setiap piksel pada citra warna mewakili warna yang merupakan kombinasi dari ketiga warna dasar RGB. Setiap titik pada citra warna membutuhkan data sebesar 3 byte. RGB didasarkan pada teoribahwa mata manusia peka terhadap panjang gelombang 630nm (merah), 530 nm (hijau), dan 450 nm (biru).

## 2.4. Grayscallling

*Grayscalling* merupakan salah satu proses *pre-processing* dalam pengolahan citra digital. Proses *grayscalling* merubah citra yang semula memiliki tiga ruang warna menjadi satu ruang warna keabuan. Proses ini dilakukan untuk menyederhanakan model citra sehingga setiap proses perhitungan yang dilakukan dalam pengolahan citra hanya perlu dilakukan satu kali pada tiap *pixel*-nya (terhadap nilai keabuan), tidak perlu melakukan tiga kali perhitungan terhadap tigaruang warna dalam satu *pixel*.

Proses perhitungan citra warna RGB menjadi *grayscale* dilakukan dengan mengalikan tiap nilai warna dengan konstanta *grayscale* dan menjumlahkan hasil kali tersebut.

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \quad \text{.....(1)}$$

Keterangan:

Y = nilai hasil *grayscale*

R = nilai warna merah (*red*) *pixel* G = nilai warna hijau (*green*) *pixel*B = nilai warna biru (*blue*) *pixel*

## 2.5. Haar Cascade

Rangkaian pengklasifikasi adalah pohon keputusan yang terdegenerasi di mana pada setiap tahap, pengklasifikasi dilatih untuk mendeteksi hampir semua objek yang diminati sambil menolak sebagian kecil dari pola non-objek. (Pathasu & Klubsuwan, 2016). Ide dari Haar-like feature adalah sebuah classifier yang di- training dengan sejumlah sampel citra dari suatu obyek. Classifier di-training menggunakan algoritma Adaboost (Lazaro, Buliali, & Amaliah, 2016). *Haar Cascade Classifier* merupakan metode yang lazim digunakan dalam pendeteksian obyek. *Haar Cascade*

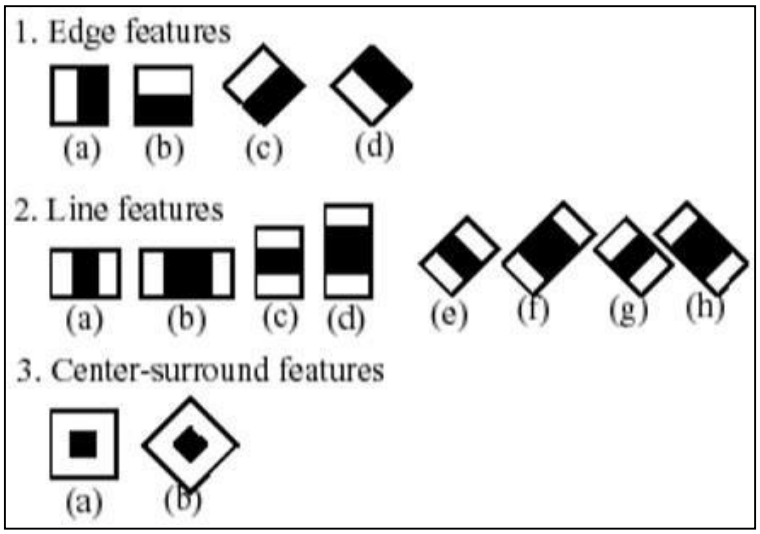
*Classifier* merupakan *rectangular features* (fungsi persegi), yang memberikan indikasi secara spesifik pada sebuah gambar. Pada prinsipnya yaitu mengenali objek berdasarkan nilai sederhana dari fitur tetapi bukan merupakan nilai piksel dari image obyek tersebut.

*Haar Cascade Classifier* merupakan gabungan dari empat algoritma sehingga memiliki tingkat akurasi yang baik. Adapun empat algoritma tersebut yaitu:

1. Pemindaian citra uji menggunakan fitur segi empat hitam dan putih bernama *Haar-Like feature*
2. *Integral image* (citra integral) untuk pendeteksian cepat
3. Metode *machine learning Adaptive Boosting (AdaBoost)*
4. *Cascade classifier* untuk deteksi objek secara bertingkat

### 2.5.1. Haar-Like feature

Fitur yang digunakan dalam *Haar Cascade Classifier* yaitu gelombang tunggal segi empat yang terdiri dari *pixel* hitam dan putih. Adapun jenis-jenis fitur*Haar* ditunjukkan pada Gambar 3.




Gambar 3. Jenis-Jenis Fitur *Haar*

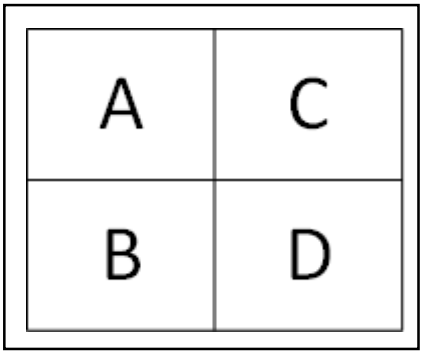
Sumber: (Lazaro, Buliali, & Amaliah, 2016)

Pemindaian dilakukan dengan cara meletakkan fitur *Haar* dimulai dari *pixel*koordinat (x,y) = (0,0) atau *pixel* ujung kiri atas. Setelah dilakukan perhitungan pada proses-proses berikutnya, maka fitur *Haar* berpindah ke kanan yaitu koordinat(1,0) dan kembali dilakukan proses perhitungan. Jika telah mencapai koordinat x maksimal, maka x kembali ke 0 dan fitur *Haar* berpindah ke nilai y berikutnya, yaitu koordinat (0,1). Proses terus dilakukan hingga mencapai koordinat terakhir yaitu

*pixel* bagian ujung kanan bawah.

### 2.5.2. Integral Image

*Integral image* digunakan untuk menghilangkan perhitungan yang besar sehingga memungkinkan semua daerah dihitung dengan kecepatan konstan. Perhitungan integral image diperoleh melalui proses seperti pada Gambar 4 dan Rumus 2.




Gambar 4.Contoh Posisi Piksel

Rumus untuk menghitung integral image pada contoh posisi pixel Gambar3yaitu:

$$D = (D + A) - (B + C.) \quad \text{.....(2)}$$

Perhitungan integral image ini hanya mengganti pixel pada posisi D, sedangkan pixel pada posisi A, B, dan C nilainya tetap.

### 2.5.3. Adaptive Boosting (AdaBoost)

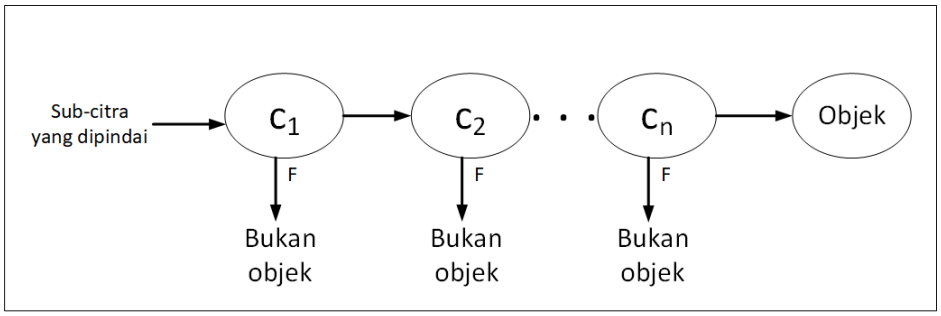
Dalam bahasa inggris, *boosting* memiliki arti "menaikkan". Sesuai namanya, algoritma *boosting* menaikkan nilai-nilai *classifier* lemah (berpotensi sebagai objek) menjadi satu *classifier* kuat. Perhitungan untuk melakukan boostingditunjukkan pada persamaan (2.2).

$$h(x, f, p, \theta) = \begin{cases} 1 \; if \; pf(x) > p(\theta) \\ 0 \; if \; pf(x) \leq p(\theta) \end{cases} \quad \text{.....(3)}$$

Dimana x adalah nilai *pixel* yang telah dirubah menjadi pixel integral, f adalah fitur terpilih, dan $\theta$ adalah ambang batas yang akan memutuskan apakah x harus diklasifikasikan sebagai positif (objek) atau negatif (bukan objek).

### 2.5.4. Cascade Classifier

*Cascade Classifier*, klasifikasi bertingkat, atau juga biasa disebut penyaringan bertingkat merupakan algoritma terakhir dalam deteksi objek menggunakan metode *Haar Cascade*. Algoritma ini berfungsi untuk mendapatkanakurasi terbaik dalam deteksi objek. Klasifikasi pada algoritma ini terdiri dari tingkatan-tingkatan berjumlah sesuai *stage* yang ditentukan pada saat pembuatan *dataset*. Semakin banyak jumlah *stage*, maka akan semakin sedikit sub-citra yang tersaring sebagai objek dengan nilai *classifier* yang paling kuat.




Gambar 5.*Cascade Classifier*

Pada tahap ini, dilakukan perhitunan pada setiap sub-citra yang dipindai menggunakan *Haar-Like feature* sebanyak n *stage*. Perhitungan yang dilakukan pada setiap *classifier* yaitu menghitung selisih antara rata-rata jumlah *pixel* area gelap dengan rata-rata jumlah *pixel* area terang pada pemindaian menggunakan fitur *Haar* yang telah melalui *Adaptive Boosting*. Jika nilai selisih tersebut lebih besar dari nilai ambang, maka sub-citra yang dipindai tersebut dikatakan sebagai objek.

## 2.6. Cropping

Cropping citra atau pemotongan citra merupakan cara pengambilan area tertentu yang akan diamati (area of interest) dalam citra, yang bertujuan untuk mempermudah penganalisaan citra dan memperkecil ukuran penyimpanan citra. Dalam proses pengolahan citra, biasanya tidak secara keseluruhan Scence dari citrayang kita gunakan. Untuk mendapatkan daerah yang kita inginkan kita dapat memotong citra tersebut. Cropping citra dapat digunakan untuk data spasial

maupun data spektral. Pemotongan citra dapat dilakukan berdasarkan titikkoordinat, jumlah pixel atau hasil zooming daerah tertentu.

## 2.7. Canny Edge Detection

Deteksi tepi *Canny*, dikembangkan oleh John F. Canny pada tahun 1986, biasa digunakan untuk mendeteksi tepi pada suatu citra digital. (Mustafid & 'Uyun, 2017). Deteksi tepi *Canny* diimplementasikan sebagai tahap segmentasi citra yangdapat memisahkan piksel objek dengan latar (bukan objek) sehingga dapat meningkatkan tingkat akurasi dalam beberapa deteksi objek. Langkah-langkah pada*Canny Edge Detection* adalah sebagai berikut :

1. Penghaluan Citra (*Noise Reduction)*

   Pada tahap penghalusan citra digunakan filter gaussian blur untuk menghilangkan noise. Dalam hal pendeteksian tepi akan sangat sensitif terhadap adanya noise pada citra. Salah satu cara untuk menghilangkan noise dengan menerapkan *Gaussian Blur* untuk menghaluskan citra tersebut. Hal tersebut dilakukan dengan teknik konvolusi citra yang dilakukan dengan Kernel Gaussian bisa berukuran 3x3, 5x5, 7x7 an seterusnya. Ukuran kernel tergantung pada efek pengaburan yang dibutuhkan. Berikut ini merupakan salah satu contoh dari kernel *Gaussian* ditunjukkan pada Gambar 6dengan ukuran 5x5 piksel.

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Gambar 6. Kernel Gaussian 5x5 Sumber:
(Mustafid & 'Uyun, 2017)

**2.** Perhitungan *Gradient (*Magnitude & Orientation*)*

Pada langkah ini dapat menghasilkan dua buah informasi dari gambar, yaitu kekuatan garis tepi (magnitude/edge strength) dan arah garis tepi (orientation/edge direction). Langkah perhitungan *Gradient* dilakukan untuk mendeteksi intensitas tepi menggunakan operator deteksi tepi. Cara paling mudah untuk dapat mendeteksi tepi pada citra adalah menerapkan filter yang menyoroti perubahan intensitas tepi arah horizontal (x) dan vertikal (y). Pada saat gambar dilakukan penghalusan, piksel turunannya menjadi Sx dan Sy. Perhitungan ini dapat diimplementasikan dengan menggabungkan nilai piksel dengan kernel *Sobel* untuk Kx horizontal dan Ky vertikal seperti pada Gambar 7.

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, K_y = \begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix}.$$

Gambar 7. Kernel SobelSumber:

(Sahir, 2019)

Proses selanjutnya, dilakukan perhitungan nilai pixel S baru untuk mengganti nilai pixel tengah pada area yang dilakukan konvolusi yang terlihat pada Rumus 3.

Keterangan:

$$|S| = \sqrt{Sx^2 + Sy^2} \quad \cdots\cdots (4)$$

S = Nilai pixel baru

Sx = Hasil konvolusi KxSy =

Hasil konvolusi Ky

3. *Non-Maximum Suppression*

   Pada proses ini akan menghasilkan garis tipis yang lebih ramping. Disinilah kita akan menggunakan nilai dari orientation yang akan kita gunakan untukmengetahui arah piksel.

4. *Double Threshold*

   Langkah Double Threshlod mengidentifikasi 3 jenis piksel yaitu kuat,lemah, dan tidak relevan. *Strong pixel* (piksel yang kuat) merupakan pikselyang memiliki intensitas tinggi dan sangat berkontribusi ke proses tepi akhir. *Weak pixel* (piksel yang lemah) merupakan piksel yang memiliki nilaiintensitas yang tidak cukup untuk dianggap sebagai tidak relevan untuk deteksi tepi. Piksel yang tidak masuk kedalam keduanya ditas maka tidak termasuk dianggap tidak relevan untuk tepi. *Double Threshold* digunakan untuk mengidentifikasi *strong pixel* (intensitas lebih besar dari nilai *high threshold*). *Low Threshold* digunakan untuk mengidentifikasi piksel yang tidak relevan (intensitas lebih kecil dari nilai *low threshold*)

5. *Edge Tracking by Hysteresis*.

   Langkah terakhir dari algoritma *Canny* yaitu melakukan Hysteris *Thresholds*. Jika nilai piksel dari hasil proses sebelumnya memiliki nilai lebih dari *High Threshold* (Batas atas), maka piksel akan diterima sebagai tepi dari gambar. Jika nilai piksel dari hasil proses sebelumnya memiliki nilai yang lebih rendah dari Lower Threshold (Batas bawah), maka piksel tadi ditolak atau tidak dianggap sebagai tepi gambar. Jika nilai piksel dari proses sebelumnya memiliki nilai antara *High Threshold* (Batas atas) dan *Low Threshold* (Batas bawah), maka piksel ini akan diterima hanya jika terhubung dengan piksel yang nilainya lebih besar dari nilai *High Threshold* (Batas atas).

## 2.8. Region of Interest (ROI)

Dalam melakukan proses pengolahan citra digital, terkadang kita hanya menginginkan pengolahan hanya pada area tertentu dari citra dan area tersebut disebut dengan *Region of Interest* (ROI).

*Region of Interest* (ROI) adalah sampel yang diambil dari satu set data untuktujuan tertentu. Tahap pengambilan ROI ini bertujuan untuk lebih fokus menganalisis area, dan menghindari error pada sistem. (Andrew, dkk., 2017).

## 2.9. Hough Circle Transform (HCT)

*Hough Transform* adalah teknik transformasi citra yang dapat digunakan untuk mengisolasi atau dengan kata lain memperoleh fitur dari sebuah citra. Karenatujuan dari sebuah transformasi adalah mendapatkan suatu fitur yang lebih spesifik,Classical Hough Transform merupakan teknik yang paling umum digunakan untukmendeteksi objek yang berbentuk kurva seperti garis, lingkaran, elips dan parabola.(Auliannisa, Suratman S.T., & Rizal ST., 2017). *Hough Cicrcle Transform* (HCT) merupakan transformasi citra yang memungkinkan objek melingkar diekstraksi daricitra, bahkan jika lingkaran tidak lengkap. HCT juga selektif untuk lingkaran, dan umumnya akan mengabaikan elips memanjang. HCT dapat mengukur pusat massadan jari-jari setiap objek lingkaran dalam citra. *Hough Circle Transform* dapat diterapkan untuk mendeteksi keberadaan bentuk lingkaran pada gambar yang diberikan. Ini digunakan untuk mendeteksi beberapa bentuk atau menemukan helm.(Pathasu & Klubsuwan, 2016). Tujuan dari HCT ini adalah untuk menemukan lingkaran dalam input gambar. Fitur-fitur dari lingkaran dihasilkan dengan "pemilihan" di ruang parameter Hough dan kemudian memilih maxima lokal atau matriks akumulator. Sebuah lingkaran lebih mudah direpresentasikan dalam parameter ruang dibandingkan dengan sebuah garis. Karena parameter lingkaran dapat langsung dirubah ke dalam parameter ruang. Persamaan lingkaran dituliskan sebagai berikut :

$$(x-a)^2 + (y-b)^2 = r^2$$

$$r = \sqrt{(x_i - a)^2 + (y_i - b)^2}$$

……(5)

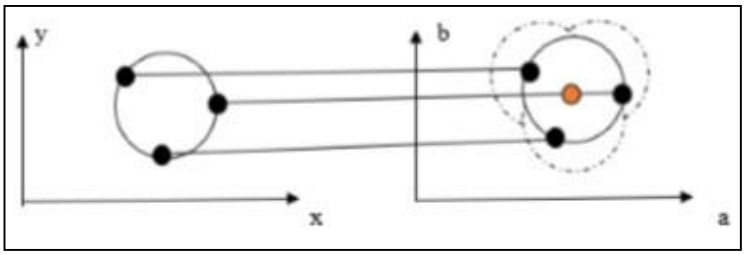
Dari persamaan diatas, lingkaran mempunyai tiga parameter yaitu r, a, dan b. Dimana a dan b adalah pusat lingkaran dalam koordinat kartesius (x, y) dan r adalah radius lingkaran. Dalam bentuk trigonometri persamaan dapat ditulismenjadi :

$$x = a + r\cos\theta$$

$$y = b + r\sin\theta$$

…… (6)

Proses pendeteksian lingkaran dengan Hough Circle Transform yang dilakukan pertama kali adalah dengan mendeteksi tepi lingkaran terlebih dahulu, dapat menggunakan metode deteksi tepi seperti *Canny* atau *Sobel*. Pada setiap tepicitra digambarkan sebuah lingkaran dengan pusat pada titik radius yang diinginkan,setiap koordinat yang dilalui garis keliling lingkaran akan masuk kedalam matriks akumulator. Matriks akumulator ini berisi jumlah garis keliling lingkaran yang melewati suatu koordinat. Matriks akumulator berisi jumlah garis keliling lingkaranyang melewati suatu koordinat. Nilai terbesar dari matriks akumulator merupakan titik tengah dari lingkaran yang terdeteksi.




Gambar 8. Ruang Geometri dan Ruang Parameter Lingkaran

# BAB III
# PERANCANGAN

## 4.5. Perancangan umum (*Quick Design*)

Kebutuhan secara keseluruhan dalam melaksanakan pembangunan sistem yaitu deskripsi kebutuhan perangkat keras (*hardware*) dan deskripsi kebutuhan perangkat lunak (*software*).

### *4.5.1.* Deskripsi Kebutuhan Pembangunan *Hardware*

Berdasarkan standar dari *website NetBeans*, spesifikasi minimum *hardware*

komputer yang disarankan untuk menginstal *NetBeans* adalah sebagai berikut :

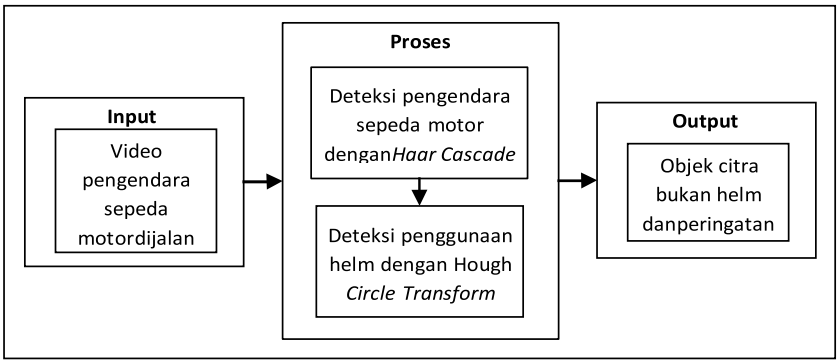
- *Processor:* 500 *MHz Intel Pentium III workstation* atau setaranya
- *Memory:* 384 *megabytes*
- *Disk space:* 125 *megabytes of free disk space*

### *4.5.2.* Deskripsi Kebutuhan Pembangunan *Software*

Kebutuhan perangkat lunak yang diperlukan dalam pembangunan aplikasi ini mencakup sistem operasi dan aplikasi. Aplikasi yang digunakan pada pembangunan sistem ini adalah *NetBeans IDE* 8.0.2 dengan *packages* sesuaidengan fitur yang dibutuhkan. Serta minimum Sistem Operasi yang digunakan padapembangunan sistem ini adalah *Windows XP Professional SP3*.

## 4.6. Perancangan umum (*Quick Design*)

Pembuatan perancangan umum untuk membentuk sistem berdasarkan tinjauan pustaka yang digunakan. Berikut ini merupakan blok diagram dari dari carakerja sistem yang dibangun untuk melakukan identifikasi penggunaan helm pada pengendara sepeda motor. Masukan diperoleh dari *file* berupa video pengendara sepeda motor di jalan yang kemudian akan diproses oleh sistem sehingga mendapatkan hasil deteksi objek helm pada pengendara sepeda motor. Blok diagram perancangan umum sistem adalah sebagai berikut :



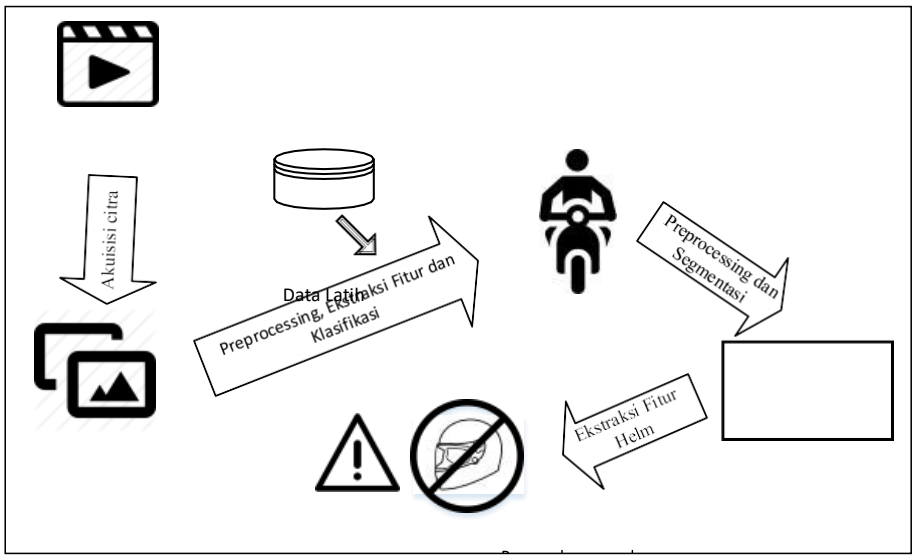

Gambar 10*. Blok Diagram* Quick Design

### 3.3. Membangun *Prototype* (*Building Prototype*)

Pada tahap ini dibuat perancangan sementara yang dilakukan dengan mewakili semua aspek perangkat keras dan lunak yang dibutuhkan dan menjadi dasar pembuatan sistem identifikasi penggunaan helm pada pengendara sepeda motor. Pada tahap ini dilakukan hingga tahap implementasi dan pengujian sistem.

#### 3.3.1 Pemodelan Sistem

Pada subbab ini menjelaskan prinsip kerja dari keseluruhan sistem, dimulaidari bagaimana proses input data sampai identifikasi penggunaan helm pada pengendara sepeda motor.



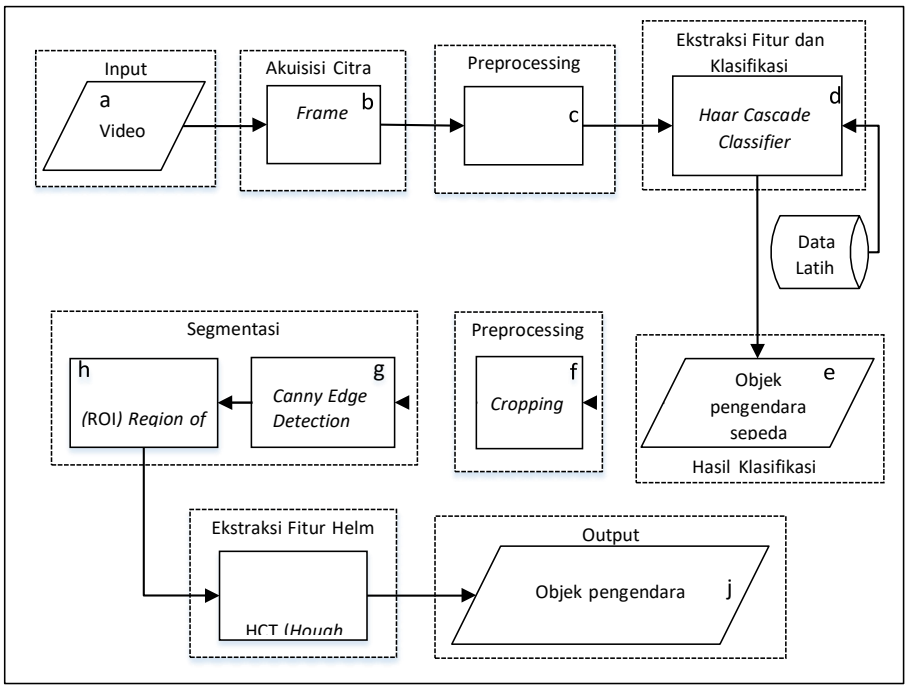

Gambar 11. Prinsip Kerja Keseluruhan Sistem

Gambar 11. merupakan gambaran prinsip kerja dari sistem identifikasi penggunaan helm pada pengendara sepeda motor. Sistem akan diberikan input berupa file rekaman video. Video tersebut akan dilakukan tahap akuisi citra untuk melakukan ekstrak dari video menjadi *frame-frame* citra. Selanjutnya dilakukan proses ekstraksi fitur dan klasifikasi untuk mendeteksi pengendara sepeda motor dari setiap frame. Setelah itu, dilakukan preprocessing dan segmentasi yang akan menghasilkan citra tepi objek pengendara sepeda motor. Selanjutnya dilakukan tahap ekstraksi fitur helm untuk mendeteksi penggunaan helm pada pengendara

sepeda motor dengan begitu dapat pula diketahui pengendara sepeda sepeda yang tidak menggunakan helm. Keluaran dari sistem yaitu objek pengendara sepeda motor tanpa helm dan peringatan.

### 3.3.2 Blok Diagram

Pada tahap ini diberikan blok diagram yang mempresentasikan proses dari sistem dengan menampilkan garis besar gambaran sistem yang dibuat, dimulai darimasukan hingga keluaran dari sistem yang dibuat. Blok diagram sistem yang dibuatadalah sebagai berikut :



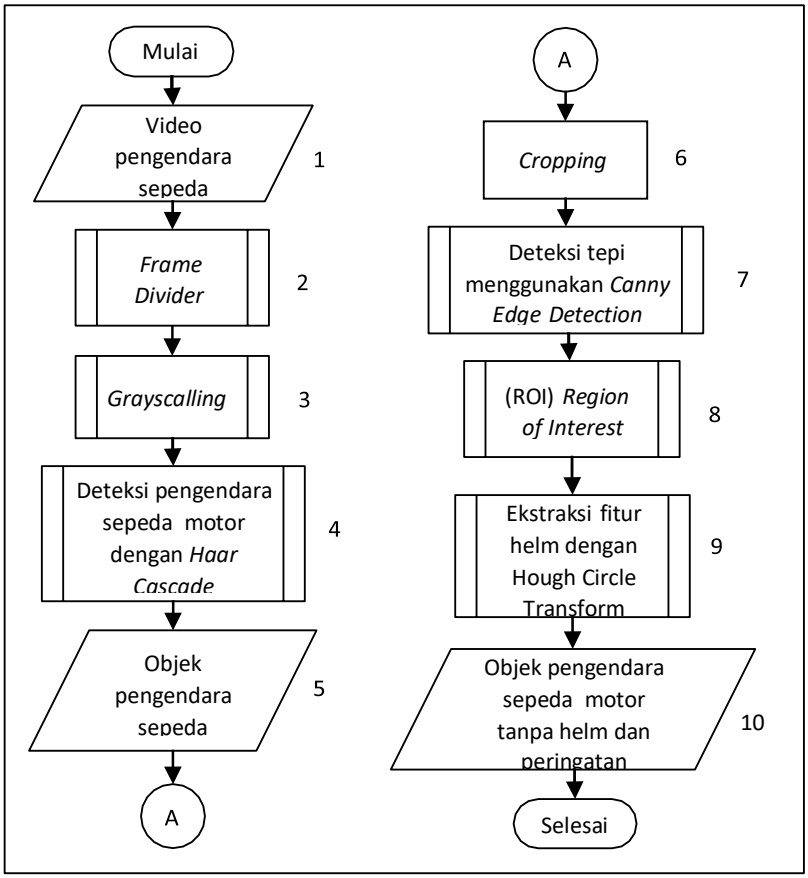

Gambar 12. Blok Diagram Sistem

Gambar 12 merupakan blok diagram dari sistem identifikasi pengendarasepeda motor tanpa helm.

1. Proses pertama merupakan akuisisi citra yaitu pengambilan data yangberupa *file* rekaman video yang juga merupakan input.
2. *Frame Divider* merupakan tahapan proses persiapan awal sistem dimanadilakukan ekstrak atau mengubah video menjadi *frame-frame* citra.
3. *Grayscalling* dilakukan untuk melakukan konversi dari citra RGB menjadi citra *grayscale* (keabuan).
4. Tahap selanjutnya ekstraksi fitur dan klasifikasi pengendara sepeda motor menggunakan algoritma Haar Cascade dimana pada tahap ini dilakukan proses pelatihan data.
5. Hasil keluaran dari tahapan sebelumnya yaitu identifikasi objek pengendarasepeda motor yang akan dijadikan masukan untuk tahapan proses selanjutnya.
6. Dilakukan *cropping* dari *frame* yang telah diketahui terdapat objek pengendara sepeda motor untuk memperjelas objek citra dan mempermudahproses selanjutnya.
7. Tahap selanjutnya merupakan tahap segmentasi citra dengan menggunakanoperator *Canny Edge Detection* untuk memperjelas atau mempertajam garistepi pada objek citra.
8. Dilakukan proses ROI (*Region of Interest*) untuk menandai suatu tempat area identifikasi sehingga didapatkan area bagian atas pengendara sepeda motor.
9. Objek citra yang telah dilakukan pendeteksian tepi, selanjutnya dilakukan tahap ekstraksi fitur dengan HCT (*Hough Circle Transform*) untuk mendeteksi helm pada pengendara sepeda motor sehingga dapat diketahui juga pengendara sepeda motor yang tidak menggunakan helm.
10. Keluaran yang dihasilkan yaitu berupa objek pengendara sepeda motor tanpa helm dan peringatan.

### 3.3.3 Flowchart

*Flowchart* berikut akan mempresentasikan gambaran cara kerja keseluruhan sistem yang dibuat yaitu sebagai berikut :




Gambar 13. Flowchart Sistem

Berdasarkan Gambar 13. yang merupakan diagram alir sistem akan dijelaskan setiap prosesnya sebagai berikut :

1. Pengambilan data berupa rekaman video pengendara sepeda motor di jalanyang menjadi masukan untuk sistem identifikasi penggunaan helm pada pengendara sepeda motor.
2. Dari input tersebut akan dilakukan akuisisi citra dengan proses *frame divider* untuk melakukan ekstrak video menjadi *frame-frame* citra yang akan dijelaskan pada subbab 3.3.3.1.
3. Proses *Grayscalling* dilakukan untuk melakukan konversi citra pengendara sepeda motor dalam bentuk citra RGB menjadi citra *grayscale* yang akan dijelaskan pada subbab 3.3.3.2.
4. Dilakukan proses ekstraksi fitur dan klasifikasi dengan menggunakan algoritma *Haar Cascade Classifier* sehingga akan mendapatkan hasil

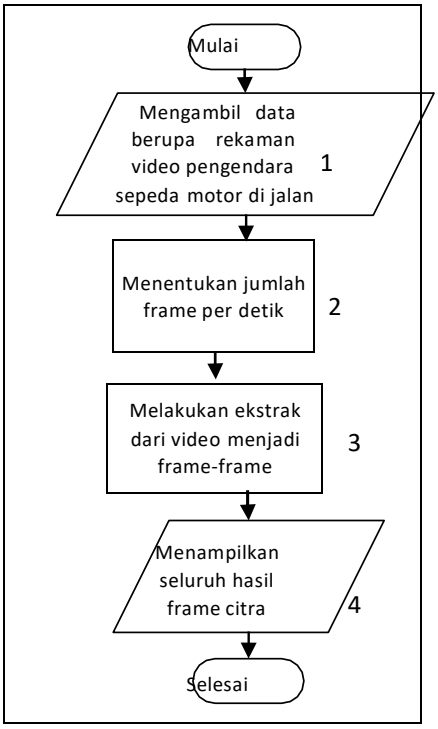
deteksi berupa citra objek pengendara sepeda motor yang akan dijelaskan pada subbab 3.3.3.3.

5. Sistem mendapatkan hasil deteksi objek pengedara sepeda motor.
6. Setelah didapatkatkan haasil deteksi yaitu citra objek pengendara sepeda motor maka selanjutnya dilkaukan proses *cropping* untuk memisahkan objek pengendara sepeda motor yang satu dengan yang lainnya dan untuk lebih memfokuskan area yang akan dilakukan tahapan selanjutnya.
7. Dilakukan tahap segmentasi citra yaitu deteksi tepi dengan *Canny Edge Detection* untuk mendapatkan garis tepi dari objek pengendara sepeda motor yang akan dijelaskan pada subbab 3.3.3.4.
8. Kemudian dilakukan proses ROI (*Region Of Interest*) pada area objek bagian atas pengendara sepeda motor agar dapat memfokuskan pada area tersebut yang akan dilakukan tahapan selanjutnya yang akan dijelaskan padasubbab 3.3.3.5.
9. Tahap selanjutnya yaitu ekstraksi fitur helm dengan menggunakan HCT (*Hough Circle Transform*) untuk mendeteksi penggunaan helm pada pengendara sepeda motor dengan begitu dapat diketahui juga pengendara sepeda motor yang tidak menggunakan helm yang akan dijelaskan pada subbab 3.3.3.6.
10. Hasil keluaran dari sistem yaitu pengendara sepeda motor tanpa helm dan peringatan.

Berdasarkan *flowchart* sistem dan uraian yang telah dijelaskan diatas terdapat beberapa subproses. Subbab berikut ini akan menjelaskan lebih rinci dari setiap subproses.

#### 3.3.3.1 Frame Divider

Pada Gambar 14, akan ditunjukan alur kerja subproses frame divider darihasil pengambilan video secara offline time.




Gambar 14. Flowchart Sitem

Berdasarkan Gambar 14, ditunjukkan alur proses *frame divider* yang berfungsi untuk mendapatkan hasil *frame-frame* citra dari video yang berisi objek pengendara sepeda motor yang akan digunakan pada proses selanjutnya. Alur dariproses frame video sebagai berikut :

1. Data set video pengendara sepeda motor di jalan yang telah diambil sebelumnya secara *offline time* yang akan digunakan sebagai masukan. Rekaman video yang diambil berdurasi selama sekitar 3-5 menit.
2. Menentukan berapa detik untuk pengambilan tiap frame dalam video.Dalam sistem yang digunakan 0.5 detik untuk konversi tiap frame.

   Jadi dapat diketahui jumlah frame, jika diambil salah satu contoh rekaman video berdurasi 3 menit yaitu :

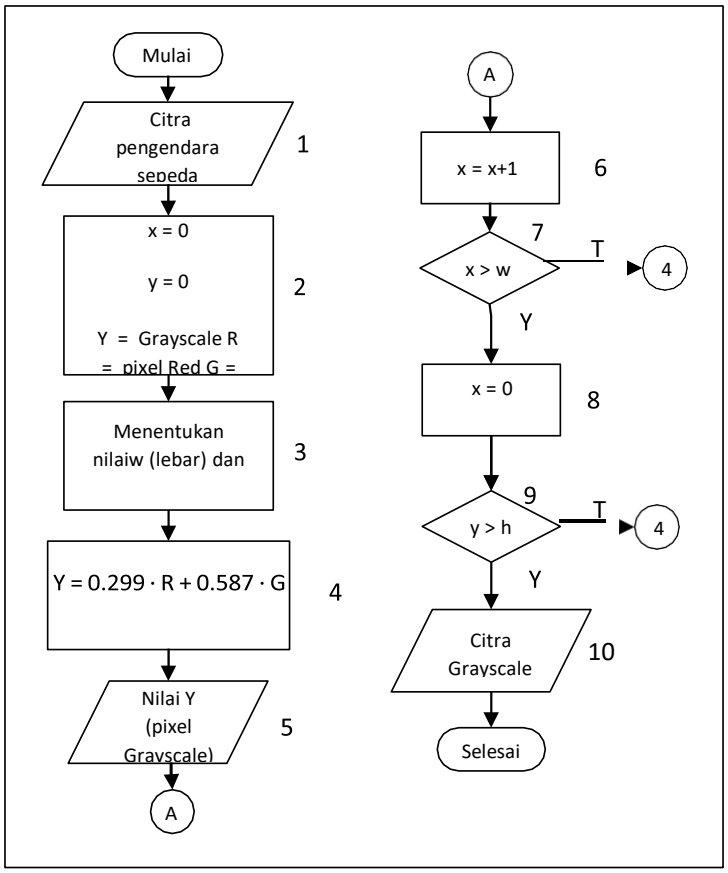
   1 $frame$ = 0.5 $detik$

   3 $menit$ = 180 $detik$

   $Jumlah\ frame$ = 180 : 0.5 = 360 $frame/detik$
3. Melakukan ekstrak dari video menjadi frame-frame.
4. Menampilkan seluruh frame yang telah dilakukan proses ekstrak dari video.

### 3.3.3.2. Grayscalling

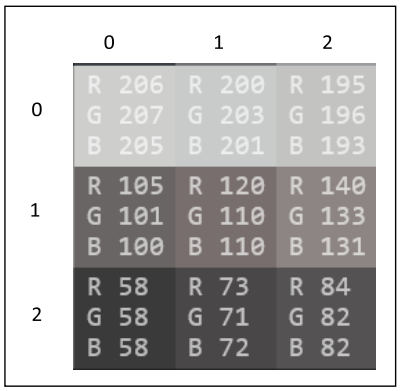
Proses *grayscalling* dilakukan sebagai tahap *preprocessing* pada proses *training* dan *testing*. Alur proses *grayscale* ditunjukkan pada Gambar 15.




Gambar 15. *Flowchart Grayscalling*

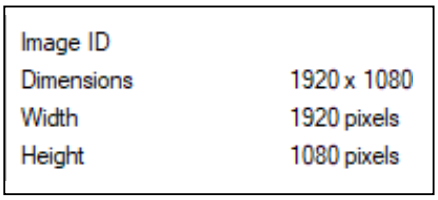
Terdapat beberapa tahap dalam melakukan proses *grayscalling*, yaitu:

1. Masukan berupa citra uji yaitu citra pengendara sepeda motor
2. Inisialisasi x dan y sebagai koordinat citra dengan koordinat awal (0,0), Y sebagai dan R, G, B sebagai nilai piksel merah, hijau, dan biru pada pikselkoordinat (x,y). Dapat dilihat pada Gambar 16 berikut merupakan sampel 3x3 pixel RGB pada citra pengendara sepeda motor.




Gambar 16. Sampel 3x3 *pixel* RGB citra uji

3. Proses pertama dari *Grayscalling* yaitu menentukan nilai panjang (*height*) dan lebar (*width*) dari citra agar proses *Grayscalling* dilakukan ke setiap pixel citra. Panjang dan lebar warna citra pengendara sepeda motor ditunjukkan pada Gambar 17.




Gambar 17.Dimensi Citra Uji *acne.jpg*

Berdasarkan Gambar 17diatas citra tersebut memiliki lebar 1920 piksel danpanjang 236 piksel. Setiap piksel memiliki jumlah 3 *channel* citra yaitu RGB yang akan dikonversi menjadi *grayscale*.

4. Setiap *pixel* RGB kemudian diubah menjadi *grayscale*. Nilai R dikalikan dengan 0.299, nilai G dikalikan dengan 0.587, dan nilai B dikalikan 0.114. Seluruh nilai hasil kali kemudian dijumlahkan menjadi satu ruang warna keabuan seperti pada perhitungan berikut untuk koordinat (1,1) *pixel* citra uji.

$$\textit{Grayscale}\ (Y) = (R \times 0.299) + (G \times 0.587) + (B \times 0.114)$$
$$= (206 \times 0.299) + (207 \times 0.587) + (205 \times 0.114)$$
$$= 61{,}594 + 121{,}509 + 23{,}37$$
$$= 206{,}473$$

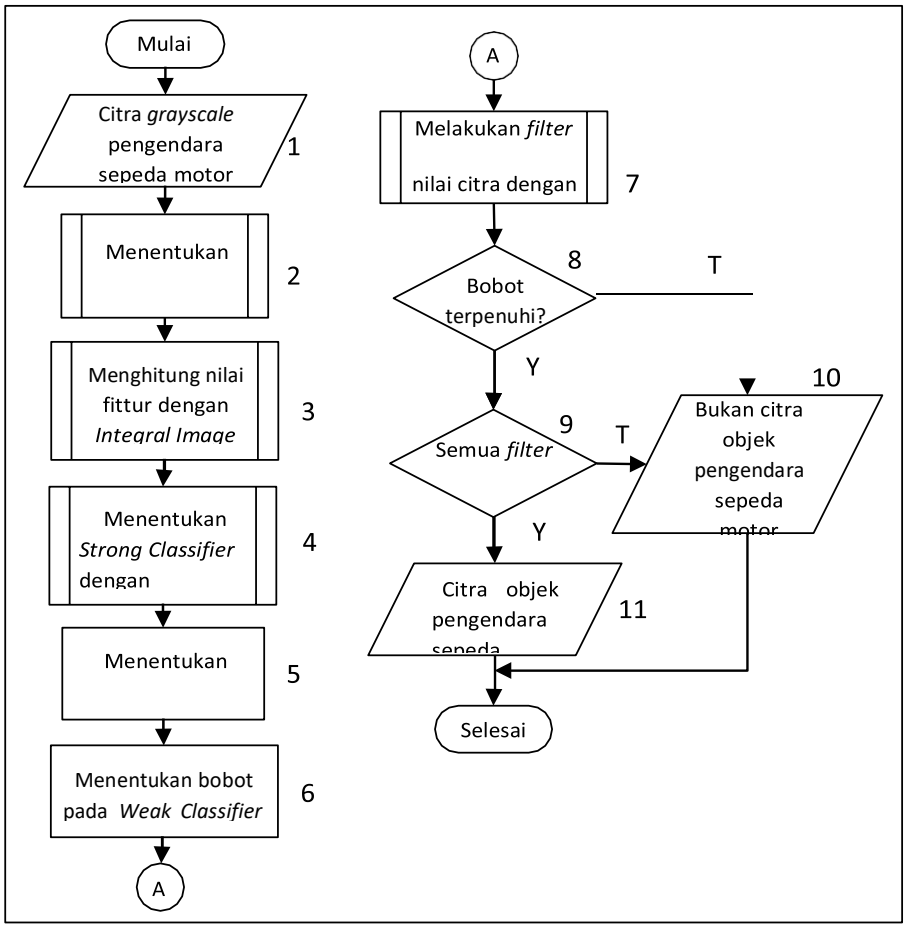
5. Diperoleh nilai piksel yang hanya memiliki satu kanal warna berdasarkan persamaan diatas. Hasil nilai perhitungan grayscale pada koordinat (1,1) dibulatkan menjadi 206.
6. Perhitungan *grayscale* kemudian dilakukan pada koordinat berikutnya yaitu (x+1, y) atau koordinat (1,2).
7. Dilakukan proses pengulangan perhitungan pada koordinat $x$ + 1 sampai koordinat $x$ pada $y$ = 0 akhir, atau nilai x sudah melebihi panjang citra.
8. Jika $x$ sudah melebihi panjang citra, maka nilai $x$ kembali ke 0 dan nilai y yang terus bertambah untuk dilakukan perhitungan.
9. Proses perhitungan dinyatakan selesai jika seluruh piksel citra uji telah dikonversikan menjadi piksel *grayscale*. Pada gambar 30ditunjukkan nilaicitra pengendara sepeda motor dalam bentuk citra grayscale untuk sampel 3x3.

Gambar 18. *Pixel* keabuan citra uji

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | R 206<br>G 206<br>B 206 | R 202<br>G 202<br>B 202 | R 196<br>G 196<br>B 196 |
| 1 | R 101<br>G 101<br>B 101 | R 112<br>G 112<br>B 112 | R 134<br>G 134<br>B 134 |
| 2 | R 58<br>G 58<br>B 58 | R 70<br>G 70<br>B 70 | R 81<br>G 81<br>B 81 |

### 3.3.3.3. Haar Cascade

Berdasarkan subbab sebelumnya, didapatkan hasil berupa frame-frame gambar yang akan diproses pada tahap selanjutnya yaitu proses *Haar Cascade* yangakan ditunjukkan dengan alur kerja pada Gambar 19.



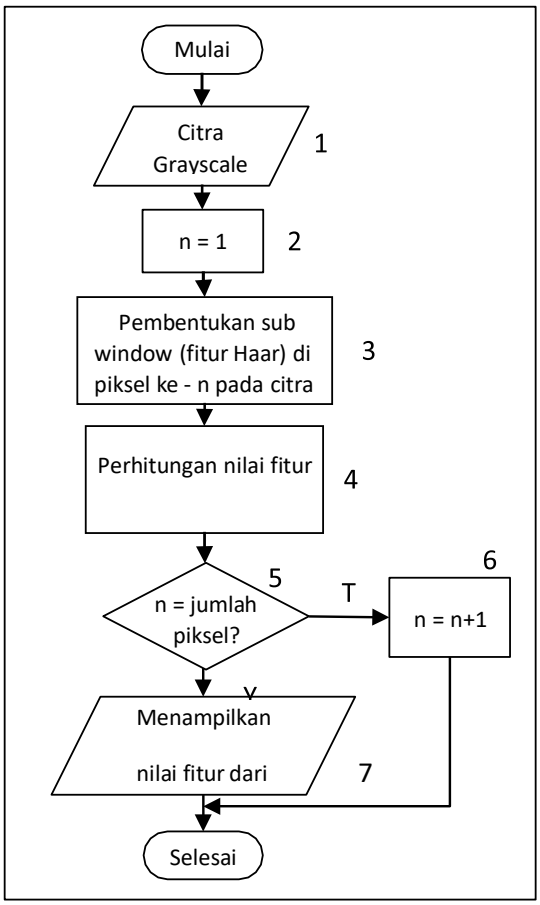

Gambar 19. *Flowchart* Haar Cascade

Berdasarkan Gambar 19, ditunjukkan alur proses *Haar Cascade* untuk mendapatkan hasil deteksi pengendara sepeda motor yang akan digunakan pada proses selanjutnya. Berdasarkan alur sistem diatas dapat juga dilihat bahwa terdapat empat proses utama dalam pendeteksian objek pengendara sepeda motor dengan menggunakan metode *Haar Cascade Classifier* antara lain: menentukan *Haar Feature*, menghitung nilai fitur dengan *Integral Image*, melakukan analisa algoritma *AdaBoost* untuk menentukan apakah fitur berisi objek pengendara sepedamotor dengan serangkaian *filter classifier* dan melakukan analisa *Cascade Classifier* untuk melakukan filter nilai citra dengan memberikan nilai bobot untuk melewati setiap filter sehingga dapat diketahui objek pengendara sepeda motor yang terdeteksi pada citra. Berikut ini merupakan penjelasan dari tahapan-tahapan

yang ada dalam proses pendeteksian objek pengendara sepeda motor dengan metode *Haar Cascade Classifier* tersebut:

**1. Haar Like Feature**

Proses pengklasifikasian citra dilakukan berdasarkan nilai dari sebuah fitur. Untuk dapat memahami bagaimana penentuan nilai fitur ini dapat dilihat pada gambar *flowchart* dari *Haar Like Feature* berikut ini :

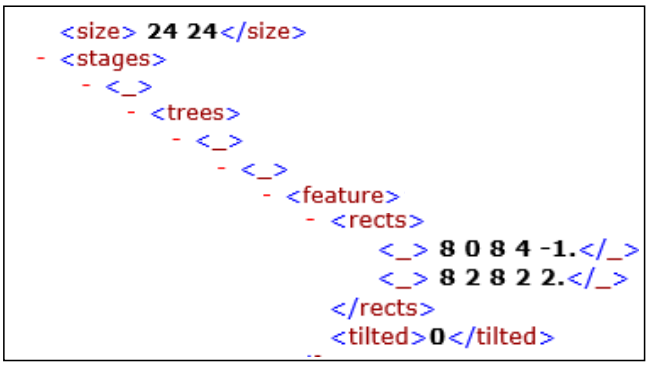



Gambar 20. Flowchart Haar Like Feature

Berdasarkan alur sistem diatas dapat diketahui masukan berupa citra dari hasil proses *grayscale* sebelumnya. Pada penelitian ini untuk menghitung nilai fitur Haar pada sebuah citra dan pada skala yang berbeda secara cepat menggunakan satu teknik yang disebut *Integral Image*. Secara umum *integrating* memiliki arti menggabungkan unit-unit kecil menjadi satu. Dalam hal ini unit-unit kecil adalah nilai piksel. Nilai integral suatu

piksel adalah jumlah dari semua piksel diatas dan di kiri piksel tersebut. Seluruh citra dapat digabungkan dengan operasi nilai yang lebih sedikit tiappikselnya. *Integral Image* sangat membantu dalam perhitungan fitur *Haar Like Feature*. Dengan menggunakan *Integral Image*, perhitungan *Haar LikeFeature* dapat dilakukan dengan sangat cepat.

Dilakukan pemindaian setiap pixel citra wajah dengan menggunakan Haar-Like feature yang terdiri dari nilai-nilai ambang dataset citra positif dan negatif seperti contoh dataset ekspresi stage 0 pada Gambar 21.




Gambar 21. Nilai Fitur Haar Stage Data Latih

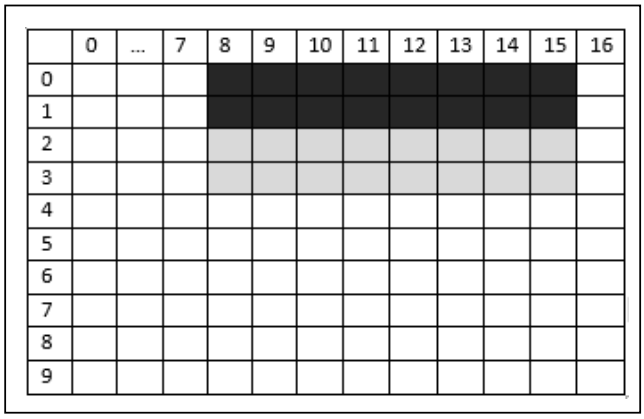
Berdasarkan Gambar 21 didapatkan informasi dari nilai-nilai objek data latih untuk deteksi objek pengendara sepeda motor yang ditunjukkan padaTabel 1.

Tabel 1. Informasi Nilai Objek Data Latih

| Tag | Angka | Informasi |
|---|---|---|
| <size> | 24, 24 | Besar area untuk hasil penandaan objek yang ditentukan. Pada dataset ekspresi, besar area yaitu<br>24 x 24 pixel |
| <!—stage --> | - | Menunjukkan informasi untuk stage ke-n, dimana stage yang ditentukan untuk dataset ekspresi sebanyak 10, sehingga stage dilakukan hingga stage<br>ke-9 |
| <rects> | 8 | Koordinat x fitur pertama classifier stage 0 yaitu x = 8 |
| <rects> | 0 | Koordinat y fitur pertama classifier stage 0 yaitu y = 0, maka koordinat (x,y) fitur pertama stage 0dimulai dari koordinat (8,0) |

| | | |
|---|---|---|
| <rects> | 8 | Menunjukkan panjang fitur pertama dalam pixel,maka fitur pertama stage 0 memiliki panjang 1<br>pixel |
| <rects> | 4 | Menunjukkan lebar fitur pertama dalam pixel, maka fitur pertama stage 0 memiliki lebar 4 pixel |
| <rects> | -1 | Nilai intensitas pixel, jika bernilai ≤ 0 maka fitur merupakan fitur gelap/hitam. Jadi, fitur pertama stage 0 merupakan fitur gelap/hitam |
| <rects> | 8, 2, 8, 2, 2 | Nilai fitur kedua pada stage 0 dengan informasi:<br>Koordinat (x,y) fitur kedua = (9,8)<br>Panjang fitur = 1 pixel<br>Lebar fitur = 8 pixel<br>Intensitas pixel = fitur terang/putih |
| <tilted> | 0 | Nilai kemiringan classifier stage 0, dimana classifier tidak memiliki nilai kemiringan |

Berdasarkan informasi tersebut, maka fitur haar stage 0 data latih yangterbentuk ditunjukkan pada Gambar 22.

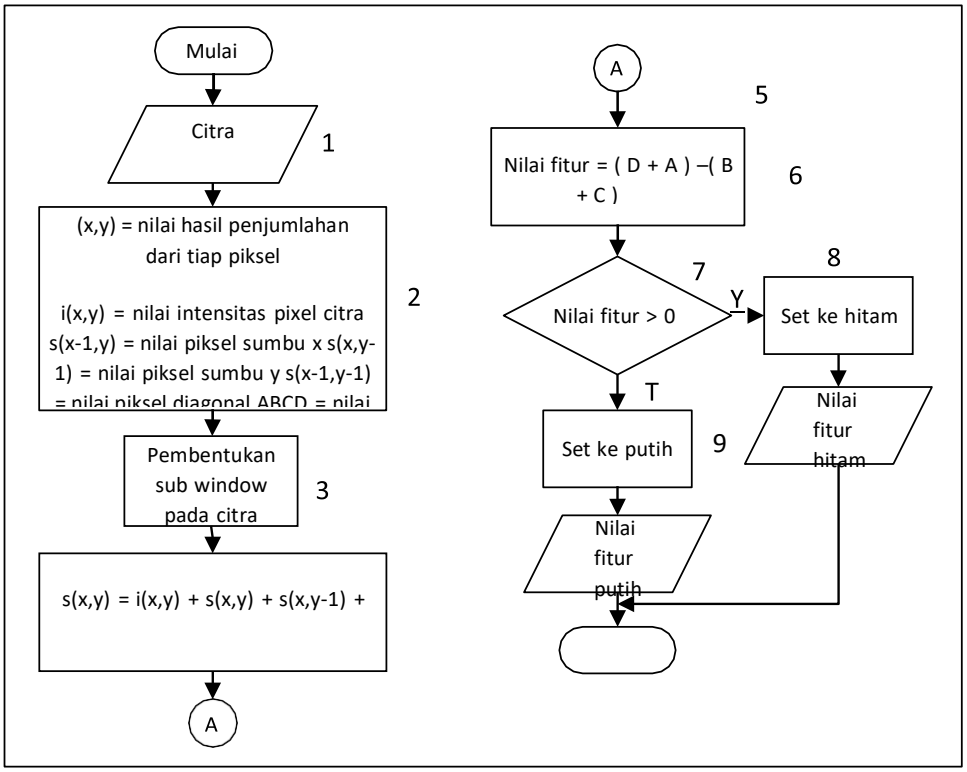



Gambar 22.Fitur Haar untuk Classifier

Gambar 23. Contoh Penerapan Fitur Haar pada Pixel Citra

## 2. Integral Image

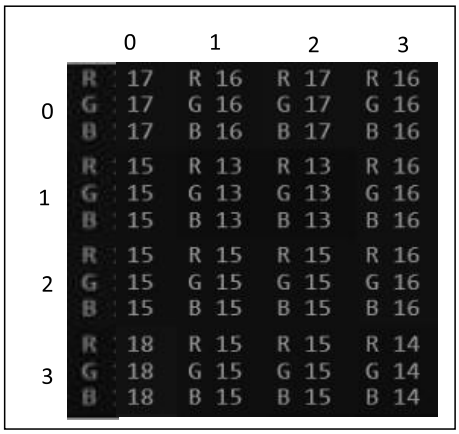
Secara umum integral mempunyai makna menambahkan bobot, bobot merupakan nilai-nilai piksel yang akan ditambahkan ke dalam gambar asli.Nilai integral dari setiap piksel merupakan jumlah dari semua piksel sebelah atasnya dan di sebelah kirinya. Keseluruhan gambar dapat diintegrasikan dengan operasi bilangan bulat tiap piksel. Alur proses dari *integral image* akan dijelaskan sesuai dengan tahapan yang terdapat dalam flowchart pada Gambar 24berikut.




Gambar 24. Flowchart *Integral Image*

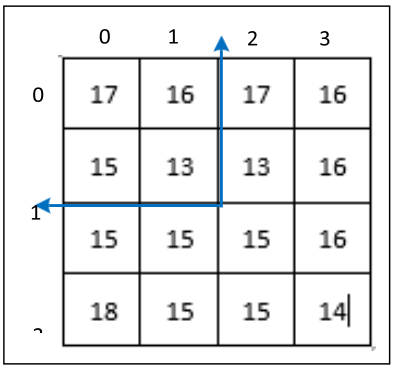
Beradasarkan gambar 3 terlihat tiap proses dari integral image dan akan dijelaskan lebih rinci antara lain :

1. Masuikan berupa citra dari hasil proses sebelumnya yaitu citra grayscale pengendara sepeda motor. Dari masukan citra tersebut dimabil sampel citraberukuran 4x4 piksel dapat dilihat pada Gambar 25berikut ini.




Gambar 25. Sampel Piksel Citra Grayscale

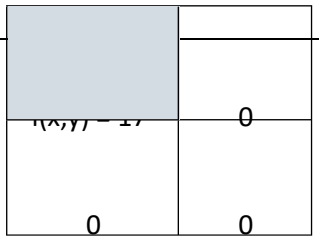
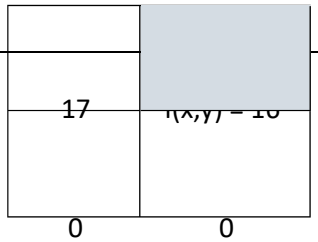
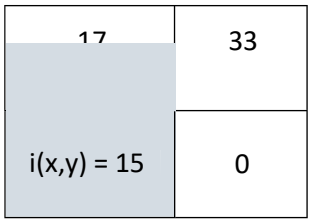
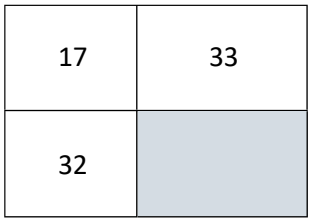
2. Mengambil nilai pixel pada koordinat (x,y). Pixel ini yang akan diproses menjadi pixel integral dan dilakukan secara menyeluruh terhadap seluruh pixel citra. Agar lebih jelas untuk nilai citra *Grayscale* yang terpindai fitur haar dapat dilihat pada Gambar 26 berikut.



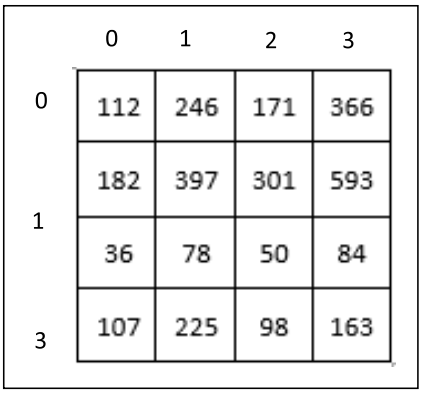

Gambar 26. Sampel Pixel Citra 4x4

3. Menghitung nilai S(x,y) sebagai nilai citra integral dari citra asli i(x,y). Sedangkan S(x,y) merupakan hasil penjumlahan nilai pixel yang sedang dihitung dengan nilai pixel pada koordinat x dan y sebelumnya. Hasil penjumlahan ini kemudian dikurangi dengan nilai pixel diagonal sebelumnya. Contoh perhitungan dilakukan pada pixel area 2x2 dari jumlaharea pixel 4x4 dan ditunjukkan pada Tabel 2 dengan menggunakan *sum area table*, atau matriks untuk menaruh piksel yang sedang dihitung, dimanapiksel yang belum dihitung sementara bernilai 0.

Tabel 2. Perhitungan Pixel Integral

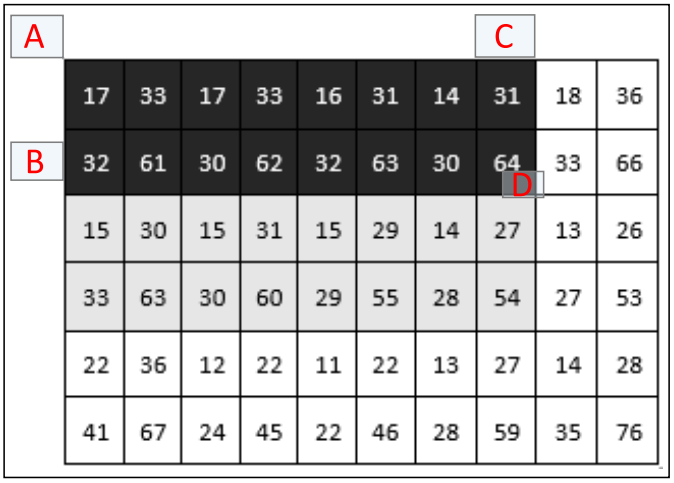
| | |
|---|---|
|   | i(x,y) = 17<br>s(x,y-1) di luar batas matriks<br>s(x-1,y) di luar batas matriks<br>s(x-1,y-1) di luar batas matriks<br>s(x,y) = 17 |
|   | i(x,y) = 16<br>s(x,y-1) di luar batas matriks<br>s(x-1,y) = 17<br>s(x-1,y-1) di luar batas matriks<br>s(x,y) = 16 + 17 = 33 |
|   | i(x,y) = 15<br>s(x,y-1) = 17<br>s(x-1,y) di luar batas matriks<br>s(x-1,y-1) di luar batas matriks<br>s(x,y) = 15 + 17 = 32 |
|   | i(x,y) = 13<br>s(x,y-1) = 33<br>s(x-1,y) = 32<br>s(x-1,y-1) = 17<br>s(x,y) = 13 + 33 + 32 – 17 = 61 |

Dilakukan proses perhitungan yang sama untuk semua piksel pada citra. Hasildari *Integral Image* yang didapatkan ditunjukkan pada Gambar 36.




Gambar 27.Hasil Perhitungan Integral Image

4. Setelah seluruh pixel yang telah dilakukan konversi menjadi pixel integral, maka dilakukan perhitungan fitur. Pada Gambar 28 ditunjukkan piksel integral pada area yang terpindai fitur Haar.

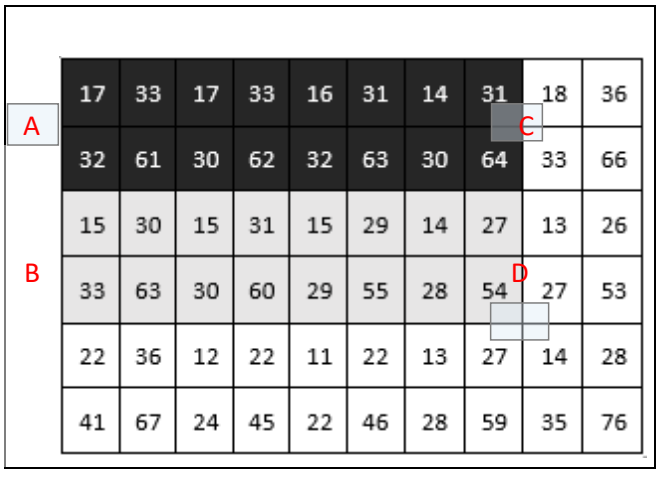



Gambar 28. Pencarian Nilai Fitur Hitam

Nilai fitur hitam = D + A – (B + C)

= 64

5. Melakukan perhitungan fitur area putih dengan cara yang sama seperti pada perhitungan sebelumnya.




Gambar 29. Pencarian Nilai Fitur Putih

Nilai fitur putih = D + A – (B + C)

= 54 – 31 = 23

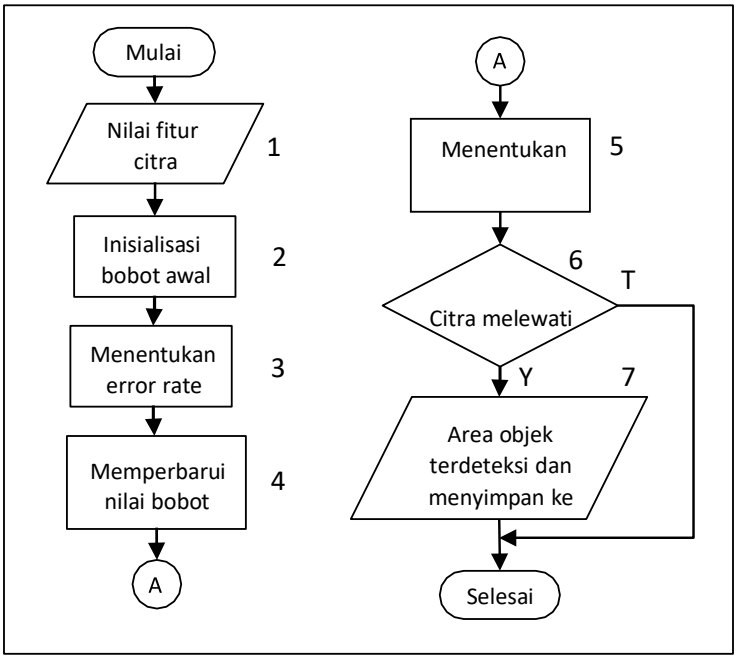
6. Dilakukan pengurangan antara nilai fitur area hitam dengan nilai fitur areaputih yang menghasilkan nilai *luminance* atau *haar like feature*.

Nilai *haar like feature* = nilai fitur hitam – nilai fitur putih

= 64 -23 = 41

**3. Adaptive Boosting (Adaboost)**

Pada penelitian ini diterapkan algoritma AdaBoost yang bertujuan mengkombinasikan banyak citra-citra yang kurang tajam (*weak classifiers*)untuk menjadi citra-citra yang lebih tajam (*strong classifiers*) dengan memberi bobot kepada citra weak classifiers. Proses tahapan algoritma *boosting* ini akan dijelaskan secara lebih terperinci berdasarkan gambar *flowchart* berikut ini:




Gambar 30. *Flowchart Adaptive Boosting*

Langkah-langkah dalam melakukan *Adaptive Boosting* yaitu:

1. Melakukan perhitungan bobot *weak classifier* dari dataset citra positif dan negatif. Contoh perhitungan dilakukan terhadap dataset ekspresi, dimana dataset memiliki 88 citra positif dan 250 citra negatif. Maka, L = 62 dan m

=

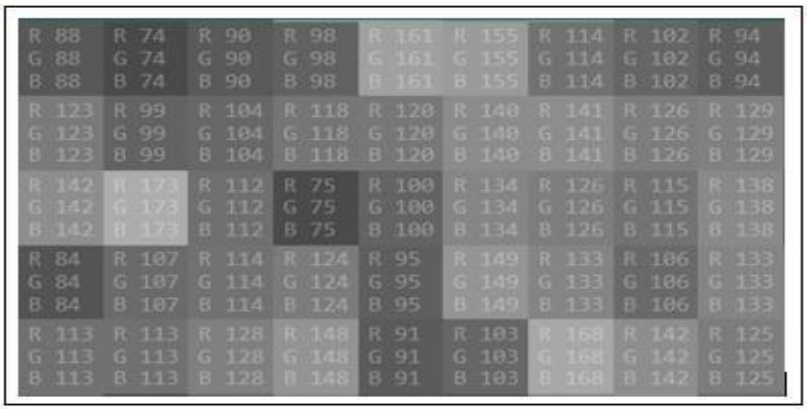
2. Menghitung bobot awal dengan contoh dataset ekspresi yang memiliki 88citra positif dan 250 citra negatif. Maka, L = 350 dan m = 88.

$$w1,0 = \frac{1}{2 \, x \, 250} = 0,002$$

$$w1,1 = \frac{1}{2 \, x \, 88} = 0,0005$$

3. Mencari nilai *error rate* terkecil pada citra negatif. Pencarian dilakukan dengan cara menghitung nilai fitur pada semua citra negatif untuk melakukan perhitungan *error rate*. Setelah itu dilakukan pengambilan nilai*error rate* yang terkecil di antara semua citra negatif. Berikut ini merupakancontoh perhitungan dilakukan pada tiga sampel citra negatif dari dataset. Perhitungan nilai fitur dilakukan dengan cara yang sama seperti pada *Integral Image*.

   Contoh perhitungan sampel citra negatif ke-1 ditunjukan pada Gambar 31 sebagai berikut :



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 88 | 74 | 90 | 98 | 161 | 155 | 114 | 102 | 94 |
| 123 | 99 | 104 | 118 | 120 | 140 | 141 | 126 | 129 |
| 142 | 173 | 112 | 75 | 100 | 134 | 126 | 115 | 138 |
| 84 | 107 | 114 | 124 | 95 | 149 | 133 | 106 | 133 |
| 113 | 113 | 128 | 148 | 91 | 103 | 168 | 142 | 125 |

Gambar 31. Sampel Piksel Citra Negatif Ke-1

Nilai fitur = | (total fitur putih) – (total fitur hitam) |hj (x)

= | (106 - 102) – 102 |

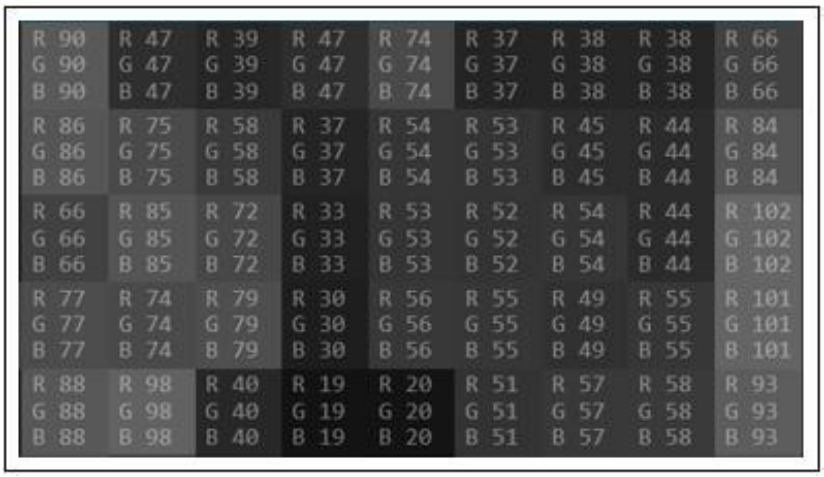
= 98

Maka nilai error rate yang didapatkan yaitu:

€j= (0,002) | 98 – 0 |

= 0,196

Contoh perhitungan sampel citra negatif ke-2 ditunjukan pada Gambar 32 sebagai berikut :




| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 90 | 47 | 39 | 47 | 74 | 37 | 38 | 38 | 66 |
| 86 | 75 | 58 | 37 | 54 | 53 | 45 | 44 | 84 |
| 66 | 85 | 72 | 33 | 53 | 52 | 54 | 44 | 102 |
| 77 | 74 | 79 | 30 | 56 | 55 | 49 | 55 | 101 |
| 88 | 98 | 40 | 19 | 20 | 51 | 57 | 58 | 93 |

Gambar 32. Sampel Piksel Pitra Negatif Ke-2

Nilai fitur = | (total fitur putih) – (total fitur hitam) |hj (x)

= | (55 - 38) – 38 |

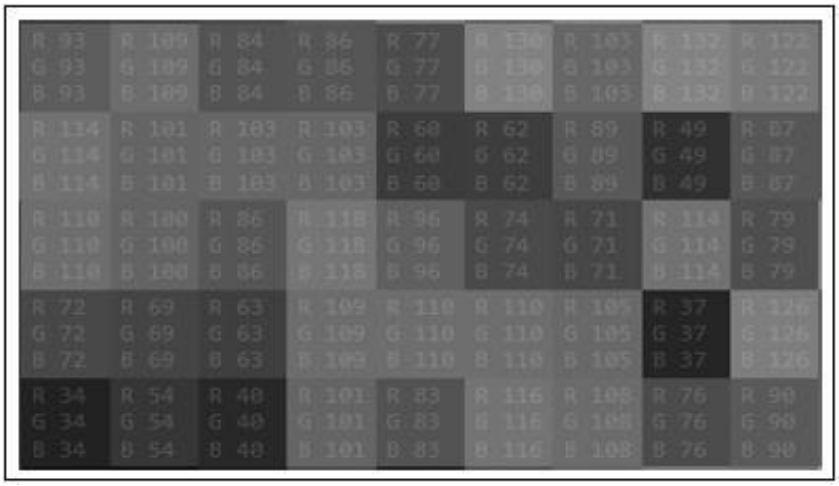
= 21

Maka nilai error rate yang didapatkan yaitu:

€j= (0,002) | 21 – 0 |

= 0,042

Contoh perhitungan sampel citra negatif ke-3 ditunjukan pada Gambar 33 sebagai berikut :



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 93 | 109 | 84 | 86 | 77 | 130 | 103 | 132 | 122 |
| 114 | 101 | 103 | 103 | 60 | 62 | 89 | 49 | 87 |
| 110 | 100 | 86 | 118 | 96 | 74 | 71 | 114 | 79 |
| 72 | 69 | 63 | 109 | 110 | 110 | 105 | 37 | 126 |
| 34 | 54 | 40 | 101 | 83 | 116 | 108 | 76 | 90 |

Gambar 33. Sample Piksel Citra Negatif Ke-3

Nilai fitur = | (total fitur putih) – (total fitur hitam) |hj (x) =

| (37 - 132) – 132 |

= 227

Maka nilai error rate yang didapatkan yaitu:

€j= (0,002) | 227 – 0 |

= 0,454

Setelah dilakukan perhitungan masing-masing *error rate* sampel citra negatif didapatkan nilai antara lain 0,196; 0,042 dan 0,454. Dalam pemilihan fitur terbaik digunakan klasifikasi data *training* menggunakan *error rate* terkecil yaitu 0,042.

4. Melakukan perhitungan bobot citra negatif. Perhitungan bobot citra negatifdilakukan dengan menggunakan nilai error rate terkecil yang telah

diperoleh. Perhitungan bobot ini dilakukan sampai mendapatkan bobot awalkurang dari 0.

$\beta j = \frac{€j}{1-€j}$

$= \frac{0,042}{1-(0,042)}$

= 0,439

Maka bobot classifier lemah setelah iterasi 1:

$wt + 1, i = wt, i \,.\, \beta j$

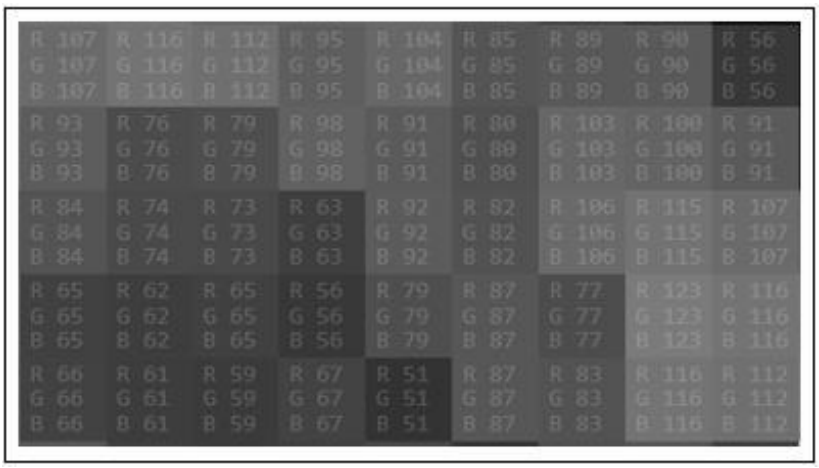
$w_{2,0}$ = 0,002 x (0,439)

= 0.000878

jika bobot awal setelah iterasi ke n jumlahnya < 0 maka iterasi berhenti. Maka, classifier negatif terbaik untuk fitur pertama classifier ekspresi memiliki bobot 0,439.

5. Melakukan pencarian nilai *error rate* terkecil pada citra positif. Sama halnyadengan proses pencarian nilai *error rate* pada citra negatif, proses yang pertama dilakukan yaitu mencari nilai fitur pada semua citra positif untuk melakukan perhitungan *error rate*. Setelah itu, proses pengambilan nilai *error rate* yang terkecil diantara keseluruhan citra positif. Berikut ini merupakan contoh perhitungan dilakukan pada tiga sampel citra positif dataset pengendara sepeda motor.

   Contoh perhitungan sampel citra positif ke-1 ditunjukan pada Gambar 34 sebagai berikut :



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 107 | 116 | 112 | 95 | 104 | 85 | 89 | 90 | 56 |
| 93 | 76 | 79 | 98 | 91 | 80 | 103 | 100 | 91 |
| 84 | 74 | 73 | 63 | 92 | 82 | 106 | 115 | 107 |
| 65 | 62 | 65 | 56 | 79 | 87 | 77 | 123 | 116 |
| 66 | 61 | 59 | 67 | 51 | 87 | 83 | 116 | 112 |

Gambar 34. Sampel Piksel Citra Positif Ke-1

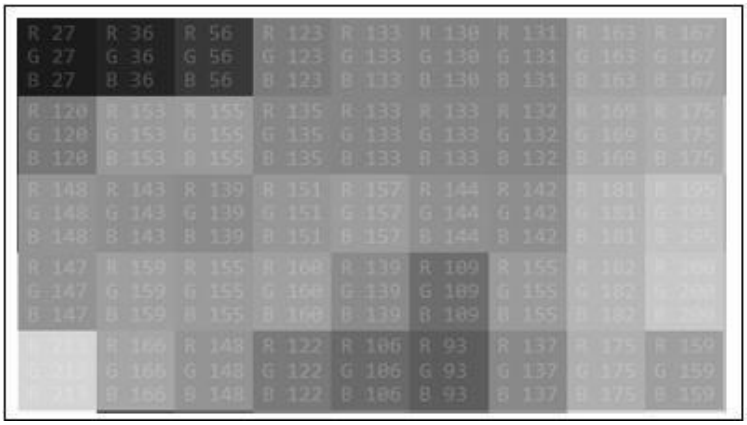
Nilai fitur = | (total fitur putih) – (total fitur hitam) | $h_t(x)$

= | (123 - 90) – 90 | = 57

Maka nilai error rate yang didapatkan yaitu:

$\epsilon_t$ = (0,0005) | 57 – 1 | = 0,028

Contoh perhitungan sampel citra positif ke-2 ditunjukan pada Gambar 35 sebagai berikut :



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 27 | 36 | 56 | 123 | 133 | 130 | 131 | 163 | 167 |
| 120 | 153 | 155 | 135 | 133 | 133 | 132 | 169 | 175 |
| 148 | 143 | 139 | 151 | 157 | 144 | 142 | 181 | 195 |
| 147 | 159 | 155 | 160 | 139 | 109 | 155 | 182 | 200 |
| 211 | 166 | 148 | 122 | 106 | 93 | 137 | 175 | 159 |

Gambar 35. Sampel Piksel Citra Positif Ke-2

Nilai fitur = | (total fitur putih) – (total fitur hitam) |ht (x)

= | (182 - 163) – 163 |

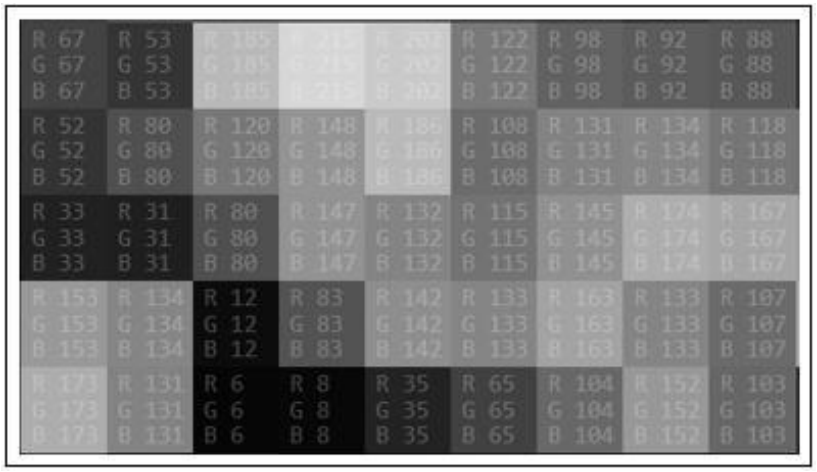
= 144

Maka nilai error rate yang didapatkan yaitu:

€t = (0,0005) | 144 – 1 |

= 0,0715

Contoh perhitungan sampel citra positif ke-3 ditunjukan pada Gambar 36 sebagai berikut :



| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 67 | 53 | 185 | 215 | 202 | 122 | 98 | 92 | 88 |
| 52 | 80 | 120 | 148 | 186 | 108 | 131 | 134 | 118 |
| 33 | 31 | 80 | 147 | 132 | 115 | 145 | 174 | 167 |
| 153 | 134 | 12 | 83 | 142 | 133 | 163 | 133 | 107 |
| 173 | 131 | 6 | 8 | 35 | 65 | 104 | 152 | 103 |

Gambar 36.Sampel Piksel Citra Positif Ke-3

Nilai fitur = | (total fitur putih) – (total fitur hitam) |ht (x)

= | (133 - 92) – 92 |

= 51

Maka nilai error rate yang didapatkan yaitu:

€t = (0,0005) | 51 – 1 | = 0,025

Setelah dilakukan proses perhitungan untuk masing-masing error rate sampel citra positif sehingga mendapatkan nilai antara lain 0,028; 0,0715 dan 0,025. Dalam pemilihan fitur terbaik digunakan klasifikasi data *training* menggunakan *error rate* terkecil yaitu 0,025.

6. Melakukan perhitungan bobot citra positif. Perhitungan bobot citra positif dilakukan dengan menggunakan nilai *error rate* terkecil yang telah diperoleh. Perhitungan bobot ini dilakukan sampai mendapatkan bobot awalkurang dari 0.

   $\beta t = \frac{€t}{1-€t}$

   $= \frac{0,025}{1-0,025}$

   = -0,256

   Maka bobot classifier lemah setelah iterasi 1:

   $w_{t+1,i} = w_{t,i} . \beta t$

   $w_{2,1}$ = 0,0005 x (-0,256)

   = -0,002

   jika bobot classifier lemah setelah iterasi ke n jumlahnya < 0 maka iterasiberhenti. Maka, classifier positif terbaik untuk fitur pertama classifier ekspresi memiliki bobot -0,256.

7. Melakukan pengecekan apakah fitur menghasilkan *strong classifier*. Pengecekan dilakukan dengan cara menghitung nilai H(x) seperti berikut ini.

   $$H(x) = \log \frac{1}{\beta j} \text{ x } h(x) \geq \frac{1}{2} \log \frac{1}{\beta t}$$

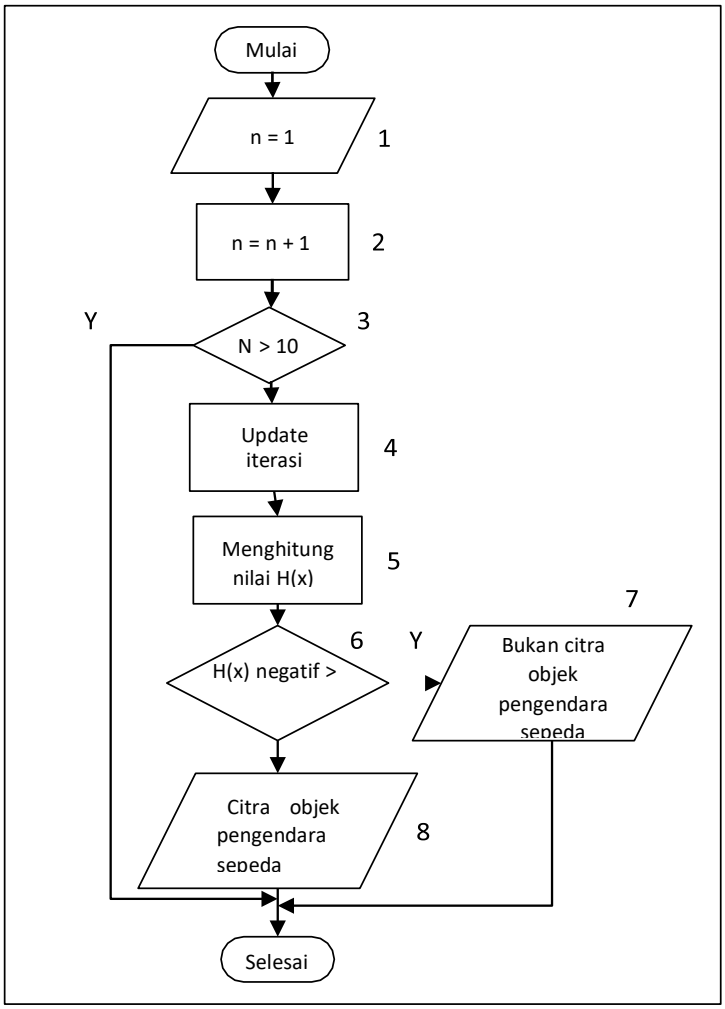
   $$= \log \frac{1}{0,439} \text{ x } 41 \geq \frac{1}{2} \log \frac{1}{0,256}$$

   **= 1,970 ≥ 0,296**

   Setelah dilakukan perhitungan didapatkan nilai H(x) yaitu 1,970 ≥ 0,296. Karena nilai 1,970 lebih besar dari 0,296 maka dapat dinyatakan fitur tersebut merupakan *strong classifier* yang selanjutnya akan diproses pada tahap *Cascade Classifier*.

**8. Cascade Classifier**

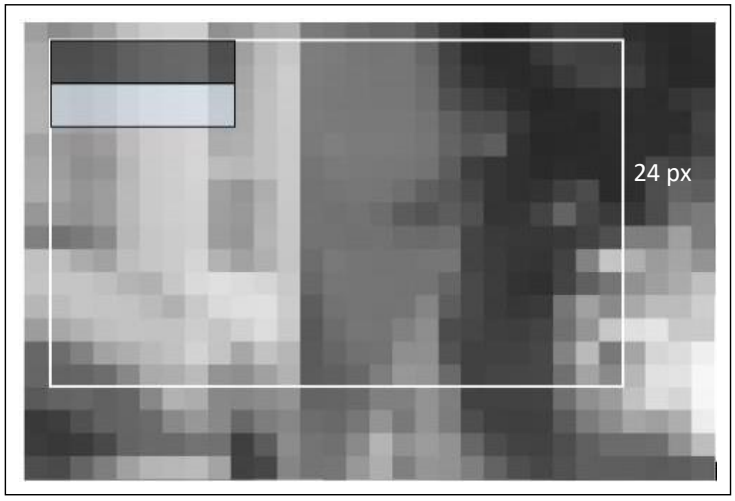
*Cascade classifier* merupakan suatu proses penyaringan bertingkatuntuk memastikan jika objek yang terdeteksi adalah objek pengendara sepeda motor menggunakan beberapa stage sejumlah 9 kali sesuai yang ditentukan pada saat proses training. Proses alur dari *Cascade Classifier* ditunjukkan pada Gambar .




Gambar 37. *Flowchart Cascade Classifier*

1. Melakukan inisialisasi variabel n sebagai *counter stage* dimulai dari *stage* ke-2. *Counter stage* dimulai dari *stage* ke-2 karena proses kalkulasi *stage* pertama telah dilakukan pada tahap *adaptive boosting* sebelumnya.

2. Masukan untuk tahap *cascade classifier* yaitu *strong classifier* berdasarkan perhitungan sebelumnya dari tahap *Adaboost*.
3. Melakukan pemindaian basis area. Proses ini dilakukan dengan membuat basis area atau area deteksi dengan ukuran yang telah ditentukan pada prosespelatihan untuk data *Haar Cascade*. Pada proses pelatihan dataset pengendara sepeda motor, basis area yang digunakan yaitu 24 x 24 pixel. Untuk setiap penambahan stage, panjang dan lebar basis area dikalikan dengan skala yang ditentukan pada proses testing. Skala yang digunakan yaitu 1.25 karena merupakan skala terbaik untuk *classifier* ekspresi yang diperoleh dengan cara *trial & error.* Maka pada stage ke-2, basis area yangdigunakan yaitu 25x25 pixel. Basis area ini selanjutnya dilakukan pemindaian dengan menggunakan fitur *haar* seperti pada Gambar 38.

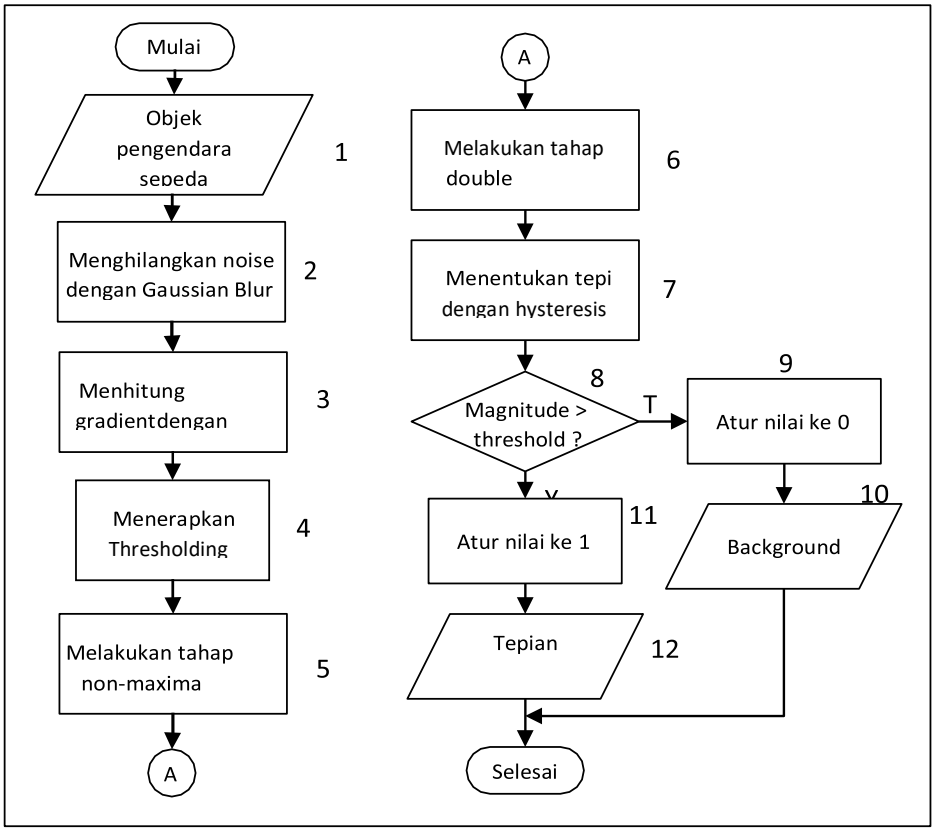



Gambar 38. Basis Area

4. Melakukan proses *adaptive boosting* kembali untuk *stage* ke-2 dengan basis area baru. Proses *adaboost* menyaring kembali *strong calssifier* pada *stage* 1 sehingga menghasilkan *classifier* yang terbaik.
5. Melakukan proses penambahan *counter stage* untuk melakukan *filter* pada *stage* berikutnya. *Stage* yang ditentukan pada proses pelatihan yaitu 10 *stage*, sehingga proses *filter* akan terus dilakukan hingga *counter stage* sudah melebihi 10.

**4. Canny Edge Detection**

Dari proses *Haar Cascade* yang dilakukan sebelumnya didapatkan hasil keluaran yaitu objek pengendara sepeda motor yang ditemukan dengan algoritma *Haar Cascade* dan proses *Cropping* dilakukan pada objek tersebut untuk lebih memfokuskan area yang akan dideteksi pada tahap pendeteksian tepi. Tahapan selanjutnya yaitu dilakukan segementasi citra yaitu deteksi tepi pada objek pengendara sepeda motor dengan menggunakan operasi *Canny Edge Detection* seperti pada Gambar 39. Alur dari deteksi tepi yaitu sebagai berikut :

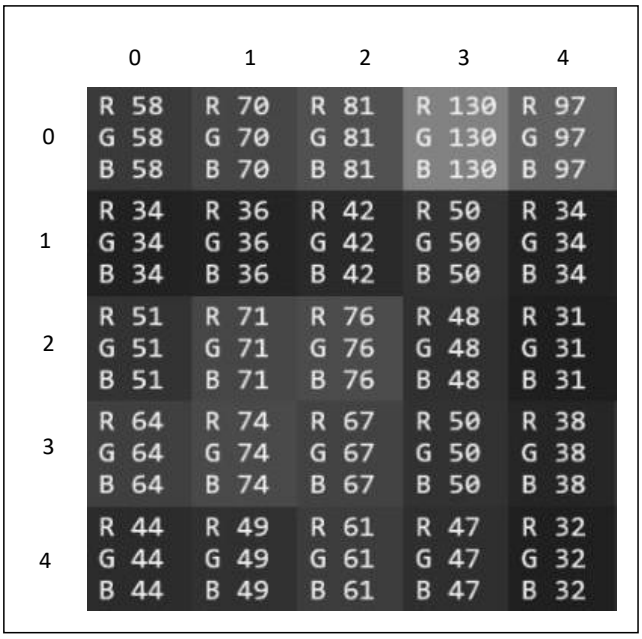



Gambar 39. Flowchart Deteksi Tepi dengan Canny

Berdasarkan gambar 6 diatas dapat dilihat bahwa *Canny Edge Detection* terdiri dari beberapa proses antara lain:

1. Masukan dari *Canny Edge Detection* yaitu citra *grayscale* dari proses hasil deteksi objek *Haar Cascade Classifier* dan proses *Cropping* pada citra yang terdapat objek pengendaara sepeda motor yang akan memproses citra yang

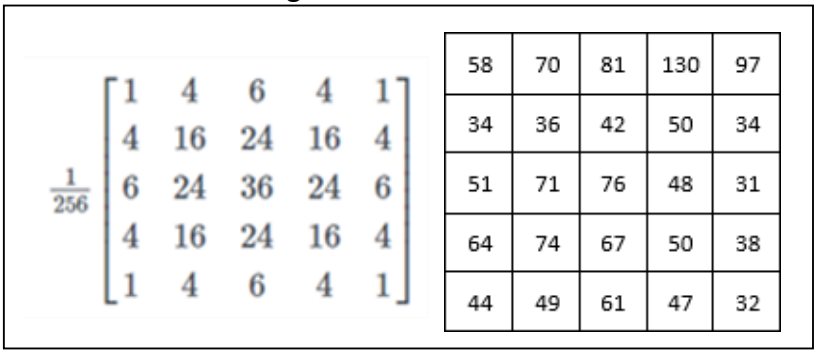
memiliki satu kanal warna yaitu keabuan. Berikut ini merupakan sampel piksel citra objek pengendara sepeda motor berukuran 5x5.



| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | R 58<br>G 58<br>B 58 | R 70<br>G 70<br>B 70 | R 81<br>G 81<br>B 81 | R 130<br>G 130<br>B 130 | R 97<br>G 97<br>B 97 |
| 1 | R 34<br>G 34<br>B 34 | R 36<br>G 36<br>B 36 | R 42<br>G 42<br>B 42 | R 50<br>G 50<br>B 50 | R 34<br>G 34<br>B 34 |
| 2 | R 51<br>G 51<br>B 51 | R 71<br>G 71<br>B 71 | R 76<br>G 76<br>B 76 | R 48<br>G 48<br>B 48 | R 31<br>G 31<br>B 31 |
| 3 | R 64<br>G 64<br>B 64 | R 74<br>G 74<br>B 74 | R 67<br>G 67<br>B 67 | R 50<br>G 50<br>B 50 | R 38<br>G 38<br>B 38 |
| 4 | R 44<br>G 44<br>B 44 | R 49<br>G 49<br>B 49 | R 61<br>G 61<br>B 61 | R 47<br>G 47<br>B 47 | R 32<br>G 32<br>B 32 |

Gambar 40. Sampel Piksel Citra Grayscale

2. Dilakukan penghilangan noise menggunakan *Gaussian Blur*. Perhitungan ini merubah nilai piksel pada bagian tengah pada kernel dengan melakukan konvolusi menggunakan kernel *gaussian* yang selanjutnya dilakukan perhitungan nilai rata-rata dari hasil konvolusi. Nilai rata-rata tersebut yang selanjutnya akan mengganti nilai tengah matriks piksel citra asli. Sebagai contoh digunakan citra sampel citra seperti pada Gambar 40 diatas yang akan dilakukan konvolusi dengan kernel Gaussian 5x5.



$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

| | | | | |
|---|---|---|---|---|
| 58 | 70 | 81 | 130 | 97 |
| 34 | 36 | 42 | 50 | 34 |
| 51 | 71 | 76 | 48 | 31 |
| 64 | 74 | 67 | 50 | 38 |
| 44 | 49 | 61 | 47 | 32 |

Gambar 41. Sampel Piksel Citra dan kernel *Gaussian*

Berikut ini merupakan piksel citra hasil konvolusi yang dapat dilihat pada Gambar 42 sebagai berikut :

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 2 | 2 | 0 |
| 1 | 2 | 4 | 3 | 1 |
| 1 | 7 | 11 | 5 | 1 |
| 1 | 5 | 6 | 3 | 1 |
| 0 | 1 | 1 | 1 | 1 |

Gambar 42. Piksel Citra Hasil Konvolusi

Dari keseluruhan piksel citra hasil konvolusi selanjutnya dilakukan perhitungan untuk mendapatkan nilai rata-rata. Hasil rata-rata yang didapatkan, digunakan untuk mengganti piksel yang berada di tengah pada piksel asli citra seperti pada Gambar 43.

$$F(1,1) = \frac{60}{25}$$

$$= 2{,}4 \approx 2$$

| | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| 0 | 58 | 70 | 81 | 130 | 97 |
| | 34 | 36 | 42 | 50 | 34 |
| 1 | 51 | 71 | 2 | 48 | 31 |
| | 64 | 74 | 67 | 50 | 38 |
| 2 | 44 | 49 | 61 | 47 | 32 |

Gambar 43. Piksel Citra Akhir Gaussian Blur

Berdasarkan perhitungan rata-rata kernel 5x5 pada bagian tengah atau koordinat (2,2) diperoleh hasil 2,4 atau jika dibulatkan menjadi 2.

3. Tahap perhitungan *Gradient*. Pada proses perhitungan gradien dilakukan untuk melakukan deteksi intensitas dan arah tepi menggunakan kernel *Sobel*.Konvolusi Sx dilakukan secara horizontal sedangkan Sy secara vertikal.

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \text{dan} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Gambar 44. Kernel Sobel

Sumber: (Wardanu, 2009)

Diambil sampel pixel citra dari berukuran 3x3 yang telah dilakukan proses sebelumnya yang ditujukan pada Gambar 45 berikut. Sampel piksel citra tersebut akan dilakukan konvolusi dengan kernel Sobel yang telah ditunjukan pada Gambar 45.

| | 0 | 1 | 2 |
|---|---|---|---|
| 0 | 58 | 70 | 81 |
| 1 | 34 | 58 | 42 |
| 2 | 51 | 71 | 2 |

Gambar 45. Sampel Piksel Citra 3x3

Berikut ini merupakan perhitungan sampel pixel citra yang dilakukan konvolusi dengan kernel Sobel.

$Sx = (-1\ x\ 58) + (1\ x\ 81) + (-2\ x\ 34) + (2\ x\ 42) + (-1\ x\ 51) + (1\ x\ 2)$

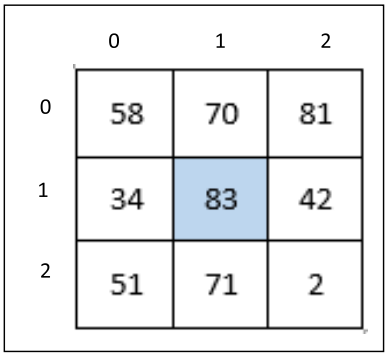
$= -10$

$Sy = (1\ x\ 58) + (-1\ x\ 51) + (2\ x\ 70) + (-2\ x\ 71) + (1\ x\ 81) + (-1\ x\ 2)$

$= 84$

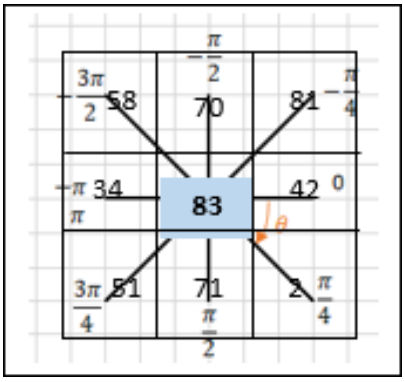
$|S| = \sqrt{Sx^2 - Sy^2} = \sqrt{(-10)^2 - 84^2} = 83{,}402 \approx 83$

Berdasarkan perhitungan yang telah dilakukan, maka nilai piksel yang gradien dapat dilihat pada Gambar 18 sebagai berikut.




Gambar 46. Piksel Citra Hasil Konvolusi Sobel

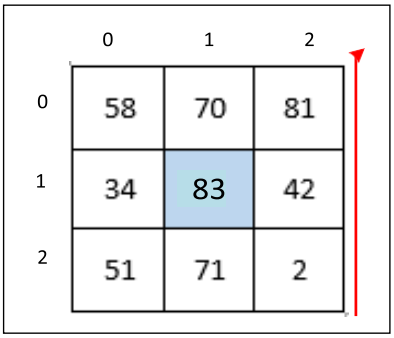
4. Tahap selanjutnya yaitu tahap *Non-Maximum Suppression* yang dilakukan untuk menipiskan garis dari hasil konvolusi *Sobel* untuk melakukan penghilangan nilai-nilai yang tidak maksimum. Jika piksel tersebut bukan merupakan nilai maksimal lokal pada arah tepi yang didapatkan dari konvolusi sobel maka tahap ini dilakukan dengan cara membuang potensi gradien di suatu piksel dari kandidat tepi. Pada Gambar 47 berikut ini merupakan contoh dari tahap *non-maximum suppression* pada sampel pixel dengan kernel 3x3 yang telah dilakukan tahap konvolusi *Sobel* sebelumnya.




Gambar 47. *Non-Maximum Suppression* pada Sampel Piksel Citra

Contoh dari proses *non-maximum suppression* dilakukan pada pixel yang ditandai kotak berwarna biru, yaitu pixel yang bernilai 83. Arah sudut dari

konvolusi *sobel* dibulatkan menjadi -40° sehingga *non-maximum suppression* yang dilakukan ke arah $-\frac{\pi}{4}$ seperti pada Gamba




Gambar 48. Arah *Non-Maximum Supression*

Jika terdapat piksel tetangga pada arah yang sama bernilai lebih besar dari piksel yang sedang diproses, maka piksel yang diproses tersebut dilakukan perubahan menjadi 0. Sedangkan, jika piksel tetangga pada arah yang sama bernilai lebih kecil dari piksel yang sedang diproses, maka nilai piksel yang diproses tersebut dipertahankan, sehingga pixel 83 tetap dipertahankan.

5. Tahap *double threshold* bertujuan mengklasifikasi dua buah nilai, yaitu *High-threshold* (T2) dan *Low-threshold* (T1). Perhitungan T1 dan T2 dilakukan pada sampel pixel citra 3x3 yang sebelumnya telah dilakukan tahap *non- maximum suppression,* dimana pada tahap tersebut didapatkan nilai nilai piksel citra terbesar yaitu 83. Nilai *high threshold* diperoleh melalui perhitungan berikut:

   T2 = nilai piksel terbesar x 0,09

   = 83 x 0,09

   = 7,47 ≈ 7

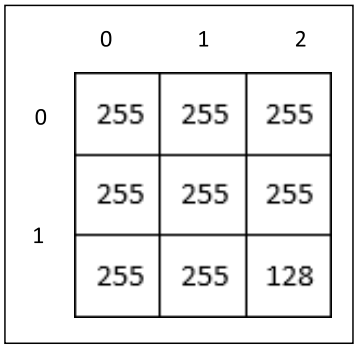
   Sedangkan nilai *low threshold* diperoleh melalui perhitungan berikut:T1 = T2 x 0,05

   = 7,47 x 0,05

   = 0,3735 ≈ 0

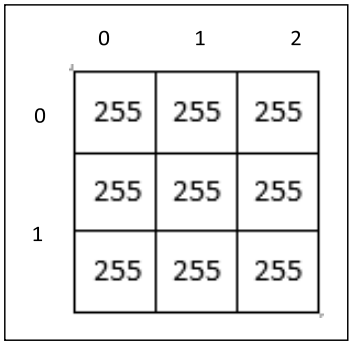
   Dari perhitungan yang telah dilakukan dapat diketahui bahwa nilai *High-Threshold* yaitu 7 dan nilai *Low-Threshold* yaitu 0. Apabila nilai piksel ≥ 7

maka nilai piksel tersebut diatur kedalam nilai 255, sedangkan apabila nilai piksel ≤ 0 maka piksel tersebut diatur kedalam nilai 0. Untuk piksel yang memiliki nilai antara 1 dan 20 disebut kandidat piksel tepi maka sementara diatur kedalam nilai 128. Beikut ini merupakan hasil nilai pixel yang diperoleh dari proses *double thresholding* yang ditunjukan pada Gambar 49.




Gambar 49. Piksel Citra Hasil *Double Threshold*

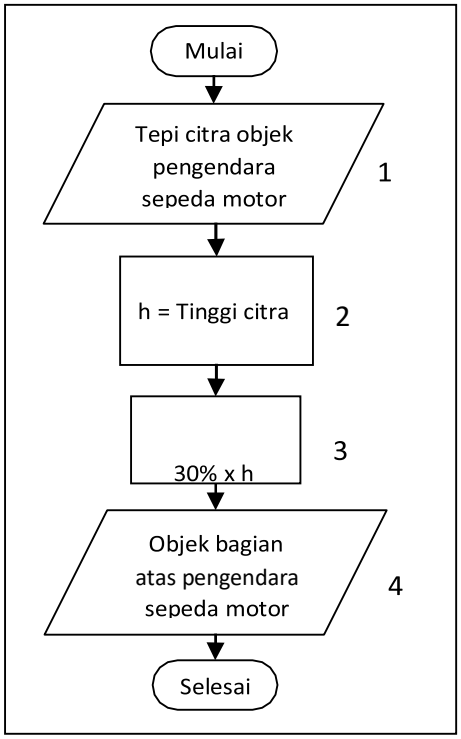
6. Tahap *Edge Tracking by Hysteresis* memiliki tujuan untuk mendapatkan tepian final dengan menekan semua sisi yang tidak terhubung pada tepian yang sangat kuat. Pada tahapan sebelumnya yaitu *Double Threshold* didapatkan suatu piksel yang menjadi piksel kandidat yang bernilai 128. Piksel yang menjadi kandidat tersebut selanjutnya dilakukan pengecekan pada piksel dari 8 arah tetangganya, sehingga piksel hanya bernilai 0 atau 255. Perubahan nilai 128 menjadi nilai 255 apabila salah satu dan semua pixel pada 8 arah tetangganya bernilai 255. Maka didapatkan hasil akhir sampel piksel citra yang dtunjukan pada Gambar 26 sebagai berikut :




Gambar 50. Piksel Citra Hasil *Edge Tracking by Hysteresis*

### 3.3.3.4. Region of Interest

Tahap *Region of Interest* atau biasa disingkat dengan ROI merupakan susatu proses untuk mendapatkan suatu area yang diamati atau area untuk lebih memfokuskan pada suatu area yang akan dilakukan tahapan berikutnya. Berdasarkan tahapan sebelumnya yaitu tahap deteksi tepi dengan algoritma Cannydidapatkan hasil tepi dari citra objek pengendara sepeda motor. Citra tepi bjek pengendara sepeda motor tersebut akan dilakukan proses perhitungan dengan*Region of Interest* atau area yang akan diamati yaitu bagian atas pengendara sepedamotor. Alur dari proses *Region of Interest* ditunjukan oleh Gambar 51 sebagai berikut :

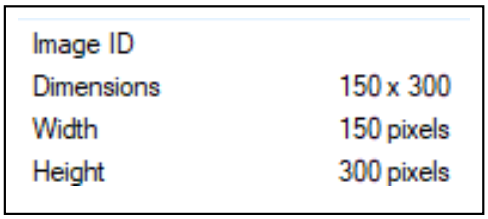



Gambar 51. Flowchart Pembuatan Region of Interest

Berdasarkan gambar *flowchart* diatas telah diketahui bagaimana proses dari ROI,berikut ini merupakan penjelasan lebih rinci dari alur kerja tersebut.

1. Masukan pada proses ROI ini merupakan objek pengendara sepeda motoryang telah dilakukan proses deteksi tepi.

2. Melakukan inisialisasi h sebagai tinggi dari citra objek pengendara sepeda motor. Untuk mendapatkan area bagian atas pengendara sepeda motor tentudiperlukan suatu informasi dari citra yang menjadi masukan. Berikut ini merupakan sampel informasi dari citra pengendara sepeda motor yang ditunjukan pada Gambar 52.




Gambar 52. Informasi Ukuran Citra

Berdasarkan Gambar 52 didapatkan informasi dari citra berupa tinggi (*height*) dan lebar (*width*) dari citra. Tinggi dari citra tersebut yaitu 300 piksel.

3. Melakukan perhitungan ROI dengan mengambil 30% dari ketinggian citra.Nilai 30% tersebut telah dilakukan pembuktian untuk menghasilkan area atas pengendara sepeda motor. Proses perhitungan dari ROI adalah sebagaiberikut:

$$ROI = 30\% \; x \; h$$

$$= 30\% \; x \; 300 = 90$$

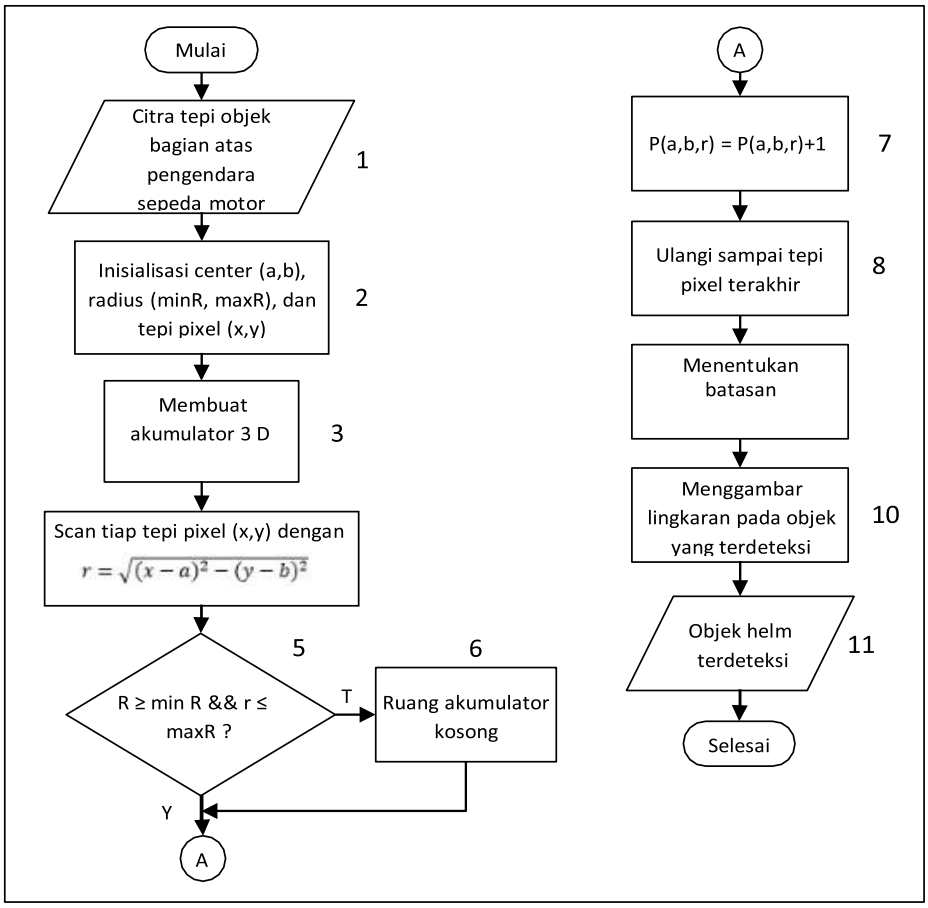
Dari hasil perhitungan tersebut dapat diketahui bahwa area bagian atas pengendara sepeda motor pada salah satu citra uji memiliki ketinggian sebesar 90 piksel.

4. Keluaran berupa tepi citra bagian atas pengendara sepeda motor.

#### 3.3.3.5. Hough Circle Transform

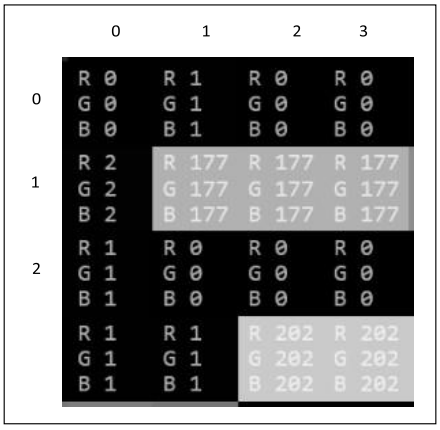
Setelah dilakukannya proses *Region of Interest* (ROI) untuk mendapatkan area atas pengedndara sepeda motor. Tahapan selanjutnya yaitu dilakukan proses ekstraksi fitur dengan menggunakan algoritma *Hough Circle Transform* yang akan mendeteksi bentuk lingkaran yang merupakan bentuk dari helm pada citra objek

bagian atas pengendara sepeda motor yang berbentuk tepian citra. Alur dari metode *Hough Circle Transform* ditunjukan ada Gambar 53sebagai berikut :



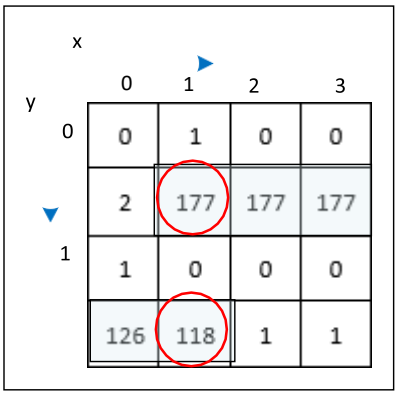

Gambar 53. Flowchart *Hough Circle Transform*

1. Masukan berupa tepi objek citra bagian atas pengendara sepeda motor yangditunjukan seperti pada Gambar berikut.




Gambar 54. Sampel Piksel Tepi Citra

2. Dilakukan proses pengambilan dua buah titik koordinat piksel ($x_i$ , $y_i$).secara acak seperti yang ditunjukan pada Gambar 55berikut.




Gambar 55. Contoh Pengambilan Titik koordinat

Berdasarkan Gambar 55 diketahui bahwa kotak yang berwarna biru merupakan piksel tepi dari citra. Diambil contoh titik koordinat dari pikseltepi citra secara acak antara lain titik koordinat (1,1) dan (1,3). Titik kooordinat tersebut akan dilakukan perhitungan untuk mendapatkan titik pusat (center a,b). Perhitungan untuk mendapat titik center adalah sebagai berikut :

($x_1$ , $y_1$) = (1,1)

($x_2$ , $y_2$) = (1,3)

$$a = \frac{(x1 + x2)}{2}, \quad b = \frac{(y1 + y2)}{2}$$

$$a = \frac{(1 + 1)}{2} = 1$$

$$b = \frac{(1 + 3)}{2} = 2$$

$(a, b)$ = (1,2)

Berdasarkan hasil perhitungan titik center didapatkan (a,b) yaitu titik (1,2). Selanjutnya dilakukan perhitungan untuk mendapatkan nilai radius yang merupakan jari-jari untuk membuat sebuah lingkaran.

$$r = \sqrt{(x_i - a)^2 + (y_i - b)^2}$$

$$r = \sqrt{(1 - 1)^2 + (1 - 2)^2} = 1$$

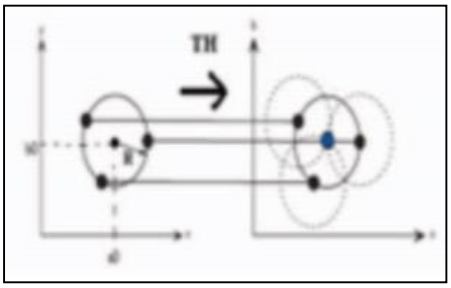
3. Selanjutnya dilakukan perhitungan untuk mencari jarak dari satu lingkaran dengan lingkaran lainnya.

$d = (x - a)^2 + (y - b)^2 - r^2$

$d = (1 - 1)^2 + (1 - 2)^2 - 1^2 = 0$

Berdasarkan perhitungan yang dilakukan didapatkan jarak untuk pembuatanlingkaran dengan jarak yang bernilai 0.

4. Menentukan nilai dari Rmax yaitu batas jari-jari lingkaran maksimum dan Rmin yaitu batas jari-jari lingkaran minum. Diambil contoh nilai dari Rminyaitu 0 dan dan Rmax yaitu 20.
5. Membuat ruang akumulator lingkaran P(a,b,r) berdasarkan nilai dari centerdan radius. Dari hasil kedua perhitungan diatas, didapatkan nilai dari titik center (a,b) yaitu (1,2) r yaitu 1. Karena nilai r lebih besar dari Rmin dan r lebih kecil dari nilai Rmax ($Rmin \leq r \leq Rmax$). Maka dibuatlah sebuah gradient lingkaran berdasarkan titik tengah dan jari-jari yang didapatkan danjarak antara lingkaran yang dibuat ini dengan yang dibuat selanjutnya berjarak 0.
6. Jika Karena nilai r lebih kecil dari Rmin dan r lebih besar dari nilai Rmax atau dengan kata lain nilai r tidak masuk kedalam *range* dari Rmax dan Rmin, maka ruang akumulator lingkaran kosong atau tidak dibuat.
7. Proses perhitungan titik center dan r serta penggambaran ini dilakukan sampai piksel terakhir dari citra.
8. Dari semua lingkaran yang dibuat pada ruang akumulator P(a,b,r) nilai a,b,ryang memiliki nilai maksimum merupakan lingkaran yang terbaik. Maka nilai dari P(a,b,r) maksimum ini yang akan digambarkan sebagai lingkaran.




Gambar 56. Proses Penggambaran LingkaranSumber :

(Pathasu & Klubsuwan, 2016)

9. Menampilkan hasi objek citra yang diberi gambar lingkaran pada bagian atas pengendara sepeda motor, hal tersebut menandakan bahwa objek yangterdeteksi adalah pengendara sepeda motor yang menggunakan helm.

### 3.3.4. Desain Layout

Berikut ini merupakan pembangunan tampilan program atau user interface untuk dijalankan pada sistem identifikasi penggunaan helm pada pengendara sepeda motor.

1. Layout Form Tampilan Awal

   Nama Dialog : Form Tampilan Awal

   Fungsi : Menampilkan halaman awal aplikasi

   Bentuk : <<Gambar 57>>

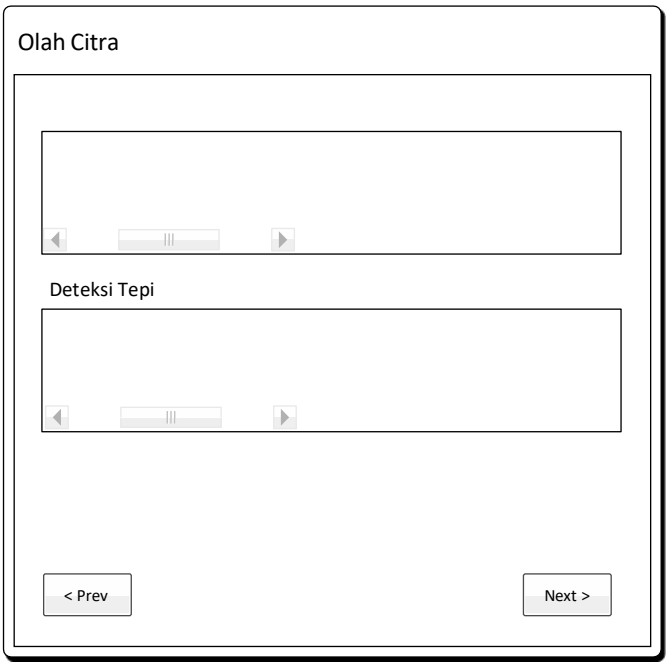

Gambar 57. *Layout Form* Tampilan Awal

2. Layout Form Tampilan Lanjutan Pertama

   Nama Dialog : Form Tampilan Lanjutan Pertama

   Fungsi : Menampilkan halaman lanjutan dari form tampilan awal

   Bentuk : <<Gambar 12>>



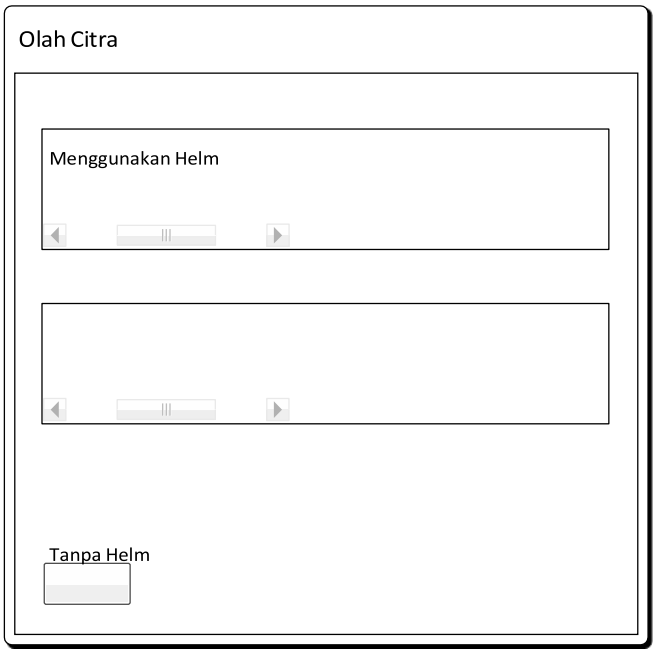

Gambar 58. *Layout Form Tampilan Lanjutan Pertama*

3. Layout Form Tampilan Lanjutan Pertama

Nama Dialog : Form Tampilan Lanjuan Pertama

Fungsi : Menampilkan halaman lanjutan dari form sebelumnya

Bentuk : <<Gambar 59>>




Gambar 59. *Layout Form Tampilan Lanjutan Kedua*

# BAB IV
# IMPLEMENTASI DAN PENGUJIAN

## 4.1. Implementasi

Pada subab ini menjelaskan implementasi dari hasil penelitian antara lain adalah perangkat yang digunakan, instalasi sistem, pengoperasian dan pengambilan data. Setelah dilakukan pembangunan sistem, kemudian dilakukan implementasi sistem dengan kata lain dilakukan pembuatan *script code* atau pemrograman pada sistem identifikasi pengendara sepeda motor tanpa helm. Pembuatan aplikasi ini menggunakan bahasa pemrograman Java, pada pembuatan apikasi ini menggunakan metode pengembangan sistem *prototype*.

## 4.2. Penyempurnaan Prototype (Refining Prototype)

Pada tahap ini dilakukan Penyempurnaan *prototype* jika perancangan ada yang kurang atau tidak tepat dari hasil evaluasi pelanggan, jika terjadi ketidaksesuaian perancangan dengan tujuan awal maka perancangan akandiperbaiki dan mengulang dari tahap desain cepat. Tahap pengembangan sistem diulang sampai hasil yang mendekati atau dinyatakan selesai. Berikut adalah hasil evaluasi yang telah dilakukan.

Penggunaan *packages* tambahan akan menambah performa dan keefektifansistem. Salah satunya yaitu OpenCV yang merupakan standard library dari pemrograman Java sehingga mendukung pembuatan dan jalannya sistem.

### 4.2.1 Perangkat yang digunakan

Untuk dapat membuat sistem aplikasi sistem identifikasi pengendara sepedamotor tanpa helm, maka beberapa proses dilakukan antara lain melalui tahapan pengolahan video dengan pengolahan citra digital untuk mendapatkan identifikasi pengendara sepeda motor yang setelah itu dapat dilakukan identifikasi penggunaanhelm.

Berikut ini adalah spesifikasi komputer yang digunakan untuk mendukung pembangunan sistem.

Sistem Operasi : *Windows 10 Enterprise 64 Bit*

Processor : *AMD FX-7500 R7, 10 Compute Cores 4C+6G*

RAM : 16 GB

### 4.2.2. Pembuatan Data Latih Haar Cascade

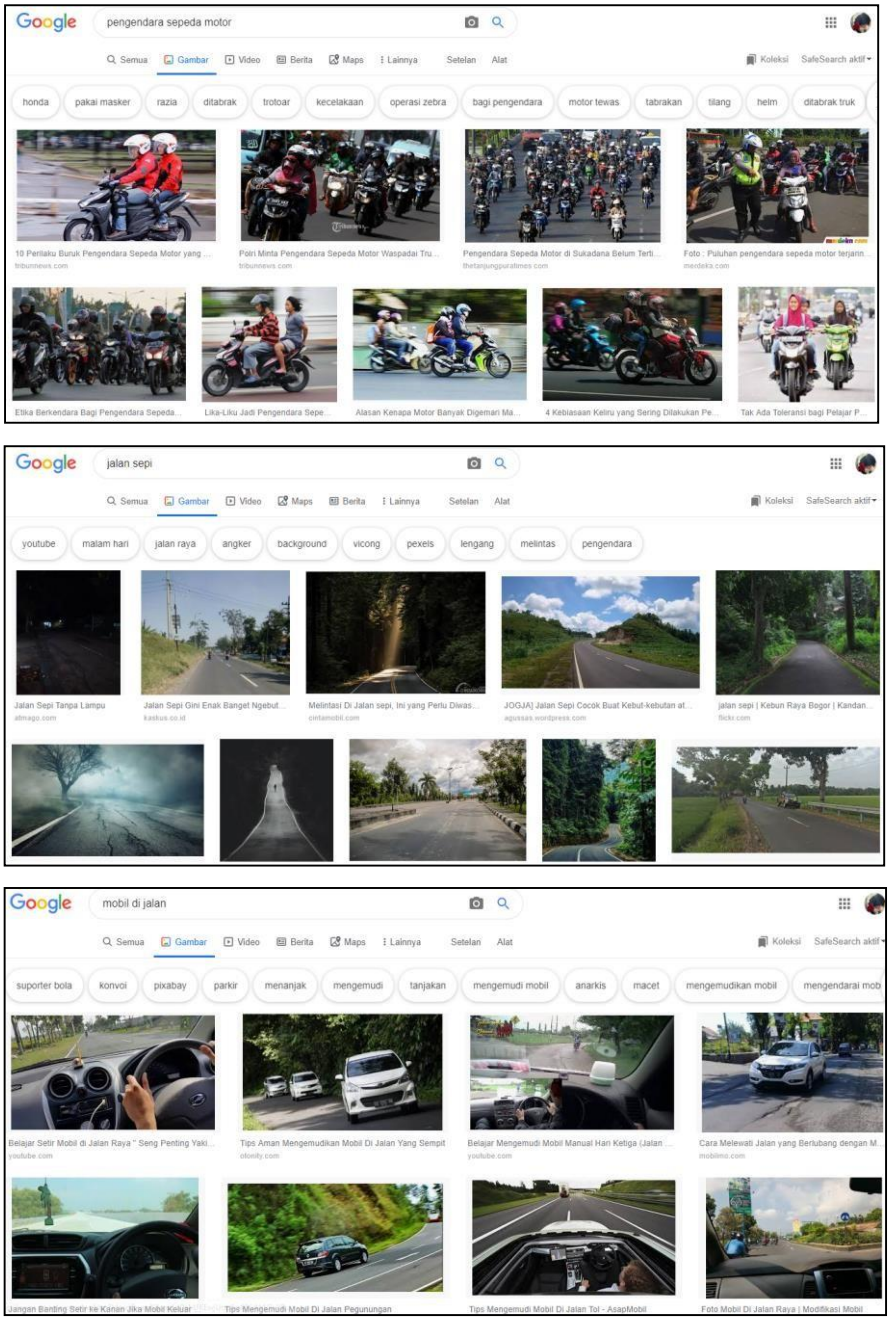
Sebelum masuk ke proses pendeteksian objek citra dengan menggunakan metode Haar Cascade, terlebih dahulu dilakukan pembuatan data latih dari mulai pengumpulan dataset dan proses training yang akan menghasilkan file .xml dimanafile tersebut merupakan data latih untuk tahapan Haar Cascade.

### 4.2.3. Pengumpulan Dataset

Pada tahap ini dilakukan proses pengumpulan dan pembuatan dataset berupa gambar yang di dapat dari *google image* yang digunakan untuk pembuatan data positif dan data negatif. Program javascript dan python untuk mengumpulkangambar dari *google image*. Pada program javascript ditujukan untuk mengambil URL gambar yang ada di google image kemudian program python yang akan melakukan eksekusi untuk mendownload gambar tersebut. Pengambilan gambar pada google image dilakukan dengan menggunakan program javascript library jQuery untuk mendapatkan URL image yang akan dijadikan dataset. Library jQuery merupakan library javascript yang cepat, kecil, dan terdapat banyak fitur lainnya

Adapun beberapa langkah dalam mendapatkan URL image menggunakan javascript library jQuery pada google chrome :

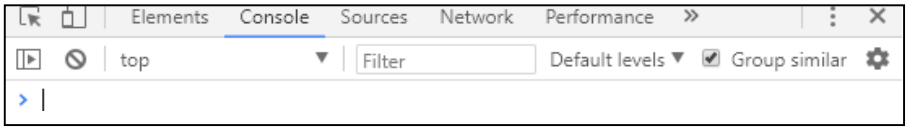
1. Memasukkan query pada google image dalam hal ini yang akan dipakai adalah istilah query “pengendara sepeda motor” untuk data positif, “jalan sepi” dan “mobil di jalan” untuk data negatif seperti berikut:




Gambar 60. Query Gambar Untuk Dataset

2. Mengumpulkan URL gambar yang akan di download menggunakan program python. Setelah melakukan search gambar sampai batas yang ditentukan dapat membuka *developer tools – console* dengan mengetikkan Ctrl+Shift+i pada keyboard.

Gambar 61. Developer tools – console




3. Memasukkan kode berikut pada console google chrome untuk mendapatkanURL gambar.

```
// pull down jquery into the JavaScript console
var script = document.createElement('script');
script.src = "https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js";
document.getElementsByTagName('head')[0].appendChild(script);

// grab the URLs
var urls = $('.rg_di .rg_meta').map(function() { return JSON.parse($(this).text()).ou; });
```

```
// write the URLs to file (one per line)
var textToSave = urls.toArray().join('\n');
var hiddenElement = document.createElement('a');
hiddenElement.href = 'data:attachment/text,' + encodeURI(textToSave);
hiddenElement.target = '_blank';
hiddenElement.download = 'urls.txt';
hiddenElement.click();
```

Gambar 62. Download URL pada Console

Berikut ini merupakan hasil download file URL bernama urls.txt yangberisikan link-link dari gambar yang akan didownload.


Gambar 63. Hasil download URL

4. Sebelumnya, telah dilakukan proses hasil download file urls melalui javascript. Program python digunakan untuk mendownload gambar dari masing-masing urls. Adapun file .py yang berisikan kode untuk mendownload seluruh gambar pada urls yang tersedia dengan nama download.py adalah sebagai berikut.

```python
from imutils import paths
import argparse
import requests
import cv2
import os

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-u", "--urls", required=True, help="path to file containing image URLs")
ap.add_argument("-o", "--output", required=True, help="path to output directory of images")
args = vars(ap.parse_args())

# grab the list of URLs from the input file, then initialize the
# total number of images downloaded thus far
rows = open(args["urls"]).read().strip().split("\n")
total = 0
```

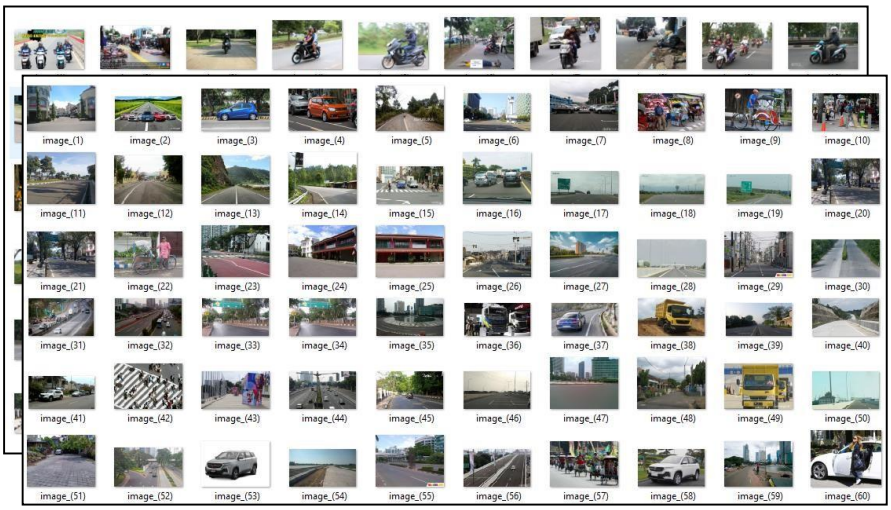
```python
# loop the URLs
for url in rows:
    try:
        # try to download the image
        r = requests.get(url, timeout=60)

        # save the image to disk
        p = os.path.sep.join([args["output"], "{}.jpg".format(
            str(total).zfill(8))])
        f = open(p, "wb")
        f.write(r.content)
        f.close()

        # update the counter
        print("[INFO] downloaded: {}".format(p))
        total += 1

    # handle if any exceptions are thrown during the download process
    except:
        print("[INFO] error downloading {}...skipping".format(p))
```

Gambar 64. Script Download URL pada Python

5. Berdasarkan script diatas, pada kode perulangan download URL digunakan untuk melakukan download dari sebuah URL yang telah didownload dan selanjutnya gambar-gambar pada URL tersebut akan disimpan pada sebuahdisk komputer secara bertahap. Setelah melakukan running script pada python maka akan terbentuk sebuah dataset berupa pengendara sepeda motor untuk data file positif dan dataset negatif berupa gambar backgroundatau gambar selain objek pengendara sepeda motor yang diambil dari hasil download melalui javascript seperti proses diatas dan penambahan gambar

lainnya dengan download manual. Berikut merupakan dataset yang terbentuk yaitu citra pengendara sepeda motor untuk data file positif yang berjumlah 500 dan citra background untuk data file negatif yang berjumlah250.

Gambar 65. Dataset File Positif

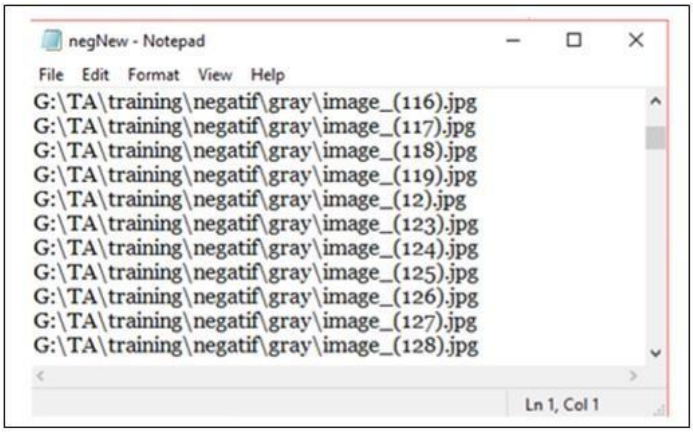


Gambar 66. Dataset File Negatif

### 4.2.4. Proses Pelatihan

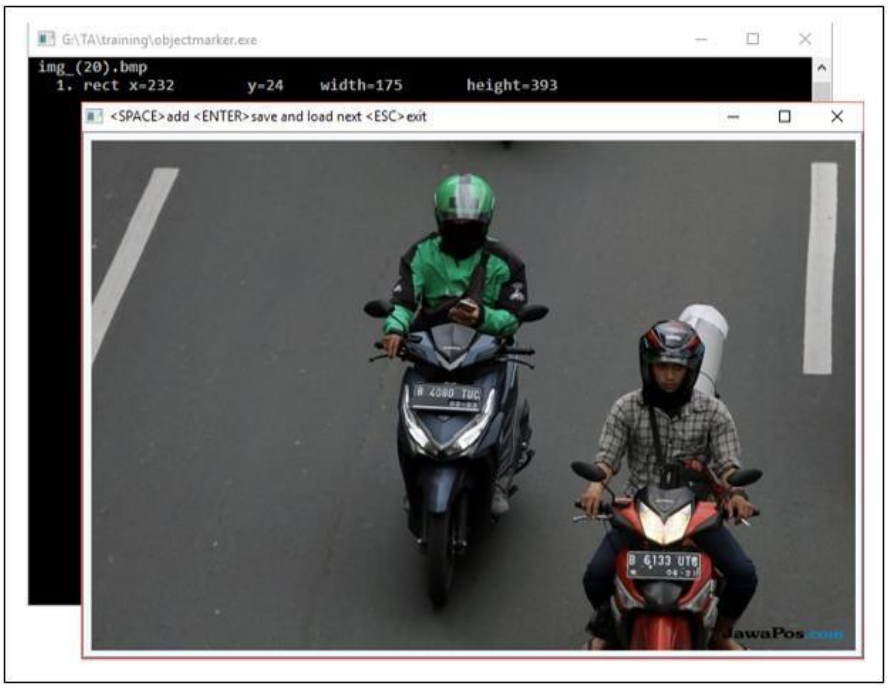
Setelah dilakukan pengumpulan dataset gambar untuk pembuatan file positif dan negatif, maka selanjutnya dilakukan proses pelatihan Haar Cascade. Tahapan-tahapan dalam proses pelatihan Haar Cascade adalah sebagai berikut :

1. Pembuatan *file* negatif dilakukan dengan melakukan *listing* untuk mendapatkan *fullpath* dari *file* ngatif.



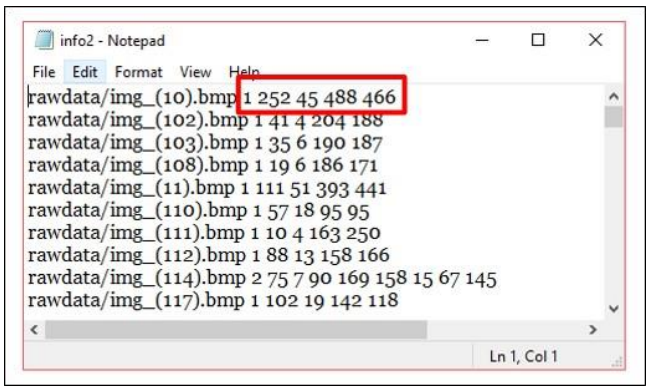

Gambar 67. Listing fullpath file negatif

2. Pembuatan file positif dilakukan dengan melakukan penandaan objek pada file positif dilakukan menggunakan aplikasi *objectmarker*. Aplikasi ini akan mengambil setiap data gambar berformat .bmp dalam folder bernama “rawdata” yang disimpan dalam direktori yang sama dengan aplikasi tersebut.




Gambar 68. Penandaan Objek Dataset Pengendara Sepeda Motor

Jika objek pada gambar sudah ditandai, selanjutnya ditekan tombol spasi pada *keyboard* untuk menambahkan data nilai dari objek tersebut secara otomatis. Adapun data nilai yang diperoleh yaitu koordinat (x,y), panjang, dan lebar dari objek. Jika ingin menandai objek pada gambar berikutnya, maka ditekan tombol *enter*. Jika seluruh gambar dalam folder sudah ditandai, maka aplikasi *objectmarker* akan keluar dengan sendirinya dan menyimpan data nilai-nilai objek dalam sebuah file teks bernama "info.txt". Dari semua dataset pengendara sepeda motor yang ada, pembuatan data file positif menjadi 88 file. Hal tersebut dikarenakan tidak semua posisi objek pengendara sepeda motor dalam posisi yang pas untuk dijadikan file positif.




Gambar 69. Data Nilai-Nilai Objek Pengendara Sepeda Motor

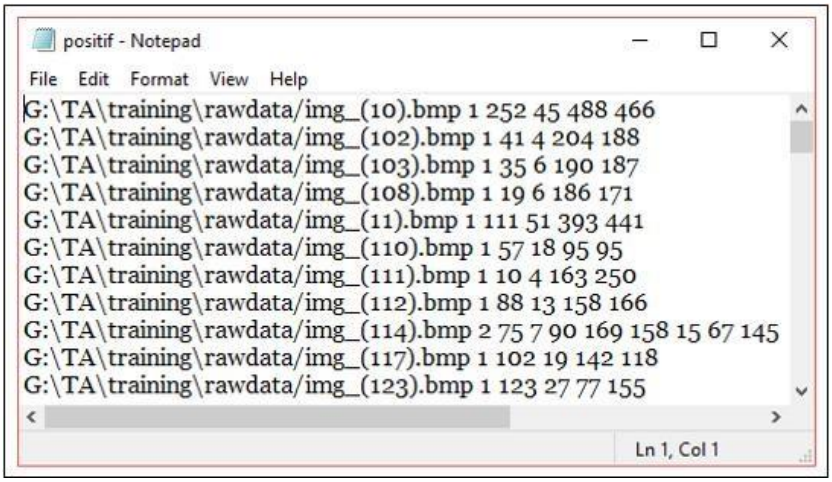
Informasi yang terdapat pada nilai objek dataset kantung mata ditunjukkanpada Tabel 3.

Tabel 3. Informasi Nilai Objek Dataset Pengendara Sepeda Motor

| Angka | Informasi |
|---|---|
| 1 | Jumlah objek yang ditandai dalam satu foto. Maka dalam foto kantung mata dark10.bmp terdapat 2 objek yang ditandai |
| 252 | Koordinat x objek pertama. Objek pertama dark10.bmp terletak pada koordinat x = 103 |
| 45 | Koordinat y objek pertama. Objek pertama dark10.bmp terletak pada koordinat y = 294. Maka, koordinat objek yaitu (103,294) |

| | |
|---|---|
| 488 | Menunjukkan panjang pixel, maka objek pertama dark10.bmp memiliki panjang 361 pixel |
| 466 | Menunjukkan lebar pixel, maka objek pertama dark10.bmp memiliki lebar 253 pixel |

3. Dilakukan *listing* untuk mendapatkan *fullpath* dari *file* positif (dengan nilaiobjek).



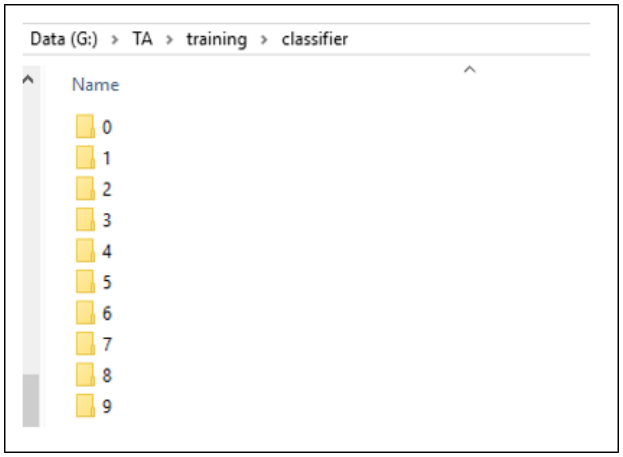

Gambar 70.*Listing Fullpath File* Negatif

4. File positif yang telah di-*list* kemudian diubah menjadi *file* vector untukmemperoleh piksel dari objek.

```
Info file name: positif.txt
Img file name: (NULL)
Vec file name: samples.vec
BG  file name: (NULL)
Num: 88
BG color: 0
BG threshold: 80
Invert: FALSE
Max intensity deviation: 40
Max x angle: 1.1
Max y angle: 1.1
Max z angle: 0.5
Show samples: FALSE
Original image will be scaled to:
        Width: $backgroundWidth / 24
        Height: $backgroundHeight / 24
Create training samples from images collection...
Done. Created 88 samples
```

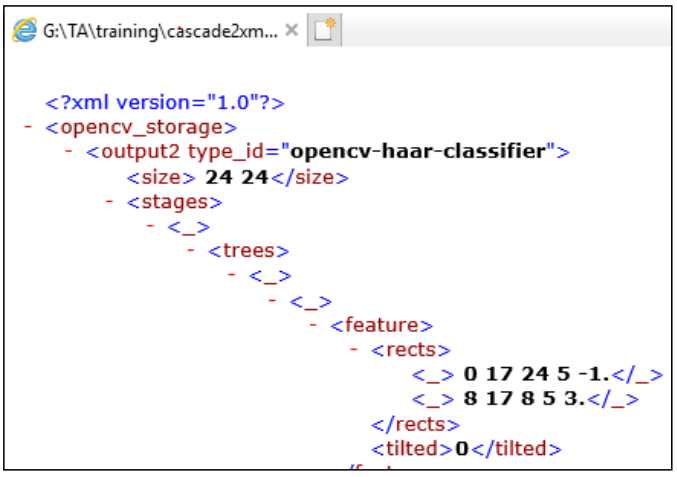
Gambar 71. Proses Pembuatan File Samples.vec

5. Dilakukan proses *haartraining* dari *file samples vector* dan *file negatif* untukmenghasilkan *file classifier* yang akan digunakan untuk deteksi objek.




Gambar 72. *File Classifier* dari Proses Haar Training

Proses ini menggunakan file samples.vec dan gambar-gambar pada listing file negNew.txt sebagai latar (bukan objek). Pada proses tertera bahwa panjang dan lebar area deteksi yaitu 24x24 *pixel* dan *stage* yang digunakan yaitu 10 *stage* sehingga *training* akan dilakukan 10 kali dimulaidari stage 0. Dalam proses tertera bahwa jumlah file positif yang digunakan yaitu 88 dan file negatif yang digunakan sebanyak 250. Proseshaartraining menghasilkan classifier yang siap digunakan untuk deteksi objek dalam bentuk file xml. Adapun hasil classifier tersebut ditunjukkanpada Gambar 73.




Gambar 73. File XML Classifier

### 4.3. Pengujian Sistem

Pada Subbab ini dijelaskan hasil pengujian fungsionalitas dengan menggunakan metode alpha test terhadap beberapa fungsi atau fasilitas yang ada pada sistem identifikasi penggunaan helm pada pengendara sepeda motor. Berikutini merupakan daftar pengujian yang dilakukan terlihat pada Tabel 4

Tabel 4. Daftar Pengujian Sistem

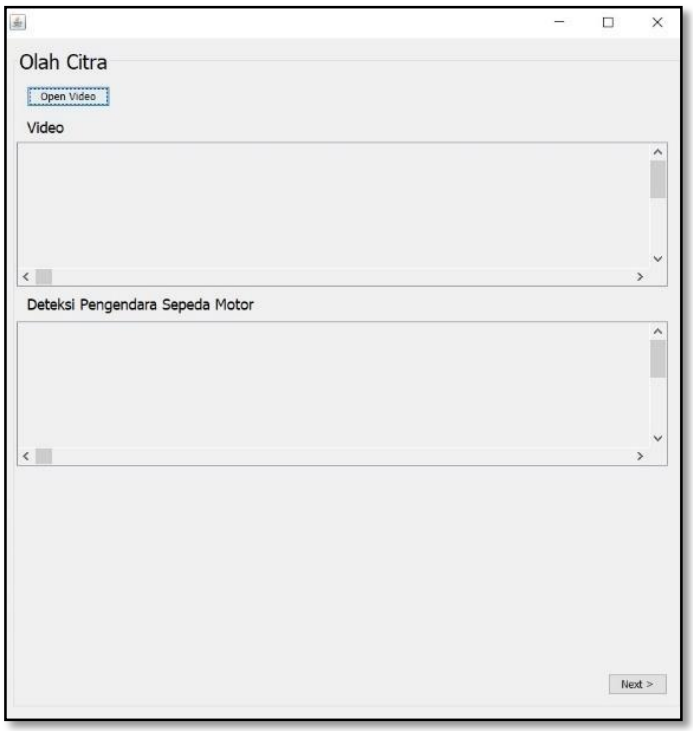
| No. Uji | Skenario Pengujian | Hasil yang Diharapkan |
|---|---|---|
| Uji-01 | Melakukan proses frame divider untuk mengekstrak video menjadi frame-frame citra | Video yang berupa masukan sistem akan diekstrak menjadi frame-frame |
| Uji-02 | Melakukan tahapan pengolahan citra dengan menggunakan Haar Cascade | Frame-frame akan dilakukan proses Haar Cascade sehingga objek pengendara sepeda motor dapat terdeteksi |
| Uji-03 | Melakukan tahapan pengolahan citra dengan menggunakan Cropping | Objek pengendara sepeda motor yang telah terdeteksi pada frame dapat dilakukan pemotongan citra |
| Uji-04 | Melakukan tahapan pengolahan citra dengan menggunakan ROI | Citra pengendara sepeda motor dilakukan perhitungan untuk menentukan bagian atas pengendara sepeda motor yang kemudian dilakukan pemotongan citra |
| Uji-05 | Melakukan tahapan pengolahan citra dengan menggunakan Canny Edge Detection | Citra bagian atas pengendara sepeda motor dilakukan proses deteksi tepi dengan Canny yang diantaranya proses tersebut anata lain Grayscale, Gaussian Blur dan Thresholding sehingga didapatkan tepian dari citra |
| Uji-06 | Melakukan tahapan penolahan citra dengan menggunakan HCT | Citra hasil dari proses sebelumnya dilakukan perhitungan untuk pencarian lingkaran yang merupakan bentuk dari helm sehingga didapatkan objek citra berupa helm dan bukan helm |

Berdasarkan Tabel 4, ditunjukkan daftar fungsionalitas sistem identifikasi penggunaan helm pada pengendara sepeda motor yang akan diuji tingkat keberhasilan fungsi tersebut dengan metode alpha-test.

### 4.3.1. Implementasi

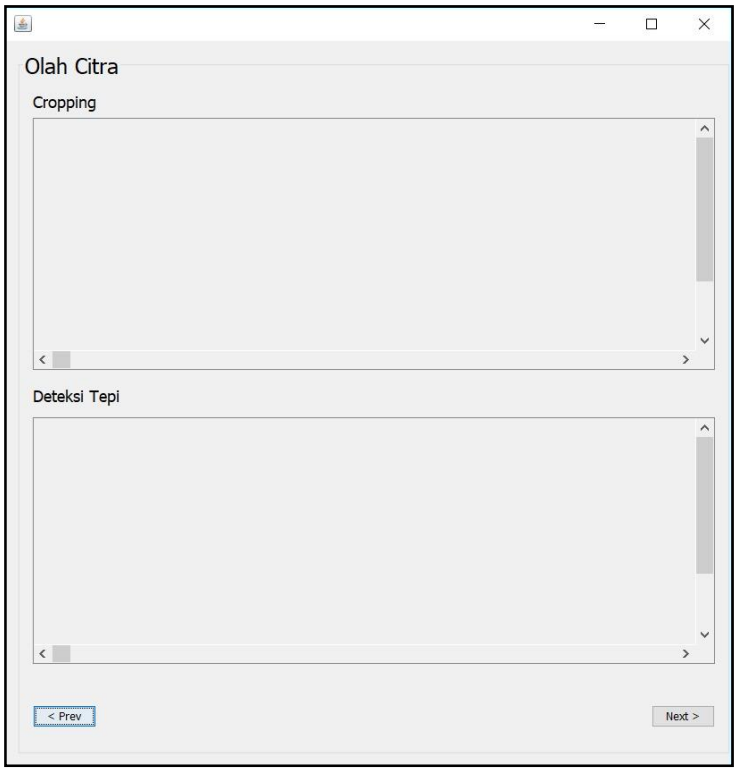
Implementasi pada sistem dilakukan dengan tujuan untuk dapat melakukantahapan awal pada aplikasi sistem identifikasi penggunaan helm pada pengendara sepeda motor. Tahapan pada aplikasi sistem identifikasi penggunaan helm pada pengendara sepeda motor ini ditampilkan pada form-form aplikasi. Dalam sistem identifikasi ini terdapat tahapan inti dalam melakukan pengolahan citra digital sebanyak 6 tahapan yaitu tahapan *Frame Divider*, *Graysacale, Haar Cascade*, *Cropping*, *Canny Edge Detection, Regoin of Interest* (ROI) dan *Hough Circle Transform* (HCT).

Berikut ini merupakan implementasi dari form form pada aplikasi yang dibuat berdasarkan desain layout yang telah dibuat sebelumnya pada Bab III Subbab 3.3.4. Pada Gambarditunjukan tampilan form ini dapat dilihat terdapat duabuah tombol yaitu “Open Video” yang digunakan untuk membuka file video yangmenjadi masukan sistem dan tombol “Next” untuk ke tampilan form berikutnya. Selanjutnya terdapat dua buah panel untuk menampilkan hasil video yang diubah ke dalam frame citra dan hasil deteksi  pengendara sepeda motor.



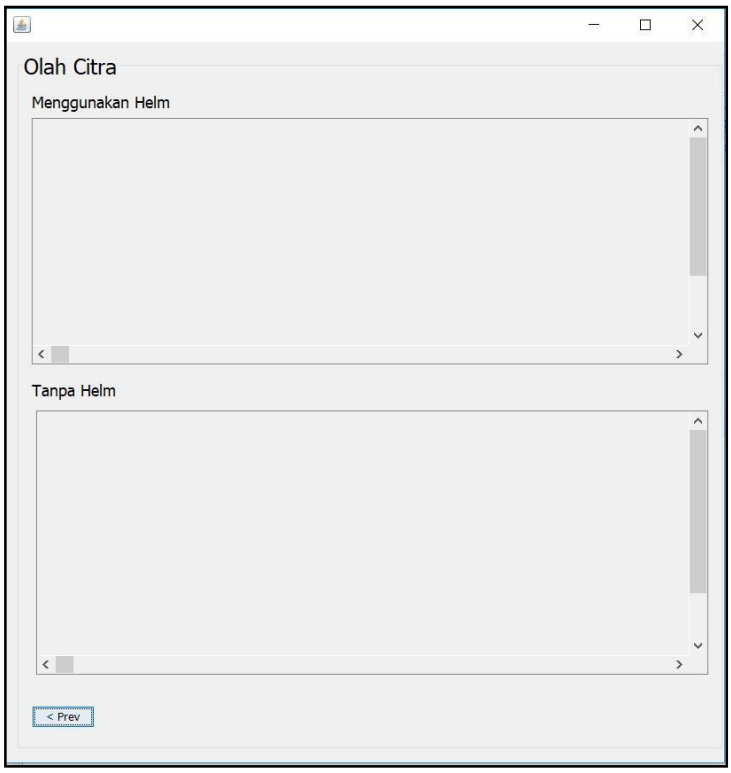

Gambar 74. Tampilan *Form* Awal Aplikasi

Tampilan form ke-2 pada aplikasi memilki dua buah panel yang untuk menampilkan hasil dari *cropping* dan hasil deteksi tepi, serta terdapat dua buah tombol yaitiu “Next” untuk pergi melihat ke form selanjutnya dan “Prev” untuk pergi melihat ke form sebelumnya. Tampilan form ke-2 dapat ditujukan pada Gambar75.




Gambar 75. Tampilan *Form* Ke-2 Aplikasi

Berikut ini merupakan tampilan form ke-3 dari aplikasi yang merupakan *form* terakhir. Pada *form* tersebut memiliki satu buah tombol “Prev” untuk kembalike *form* sebelumnya. Pada *form* ke-3 ini pun memeliki dua buah panel yang bergunauntuk menampilkan hasil proses dari citra objek pengendara sepeda motor yang menggunakan helm dan untuk menampilkan citra objek pengendara yang tidak menggunakan helm. Tampilan *form* ke-3 pada aplikasi ditunjukan pada Gambar 76sebagai berikut :

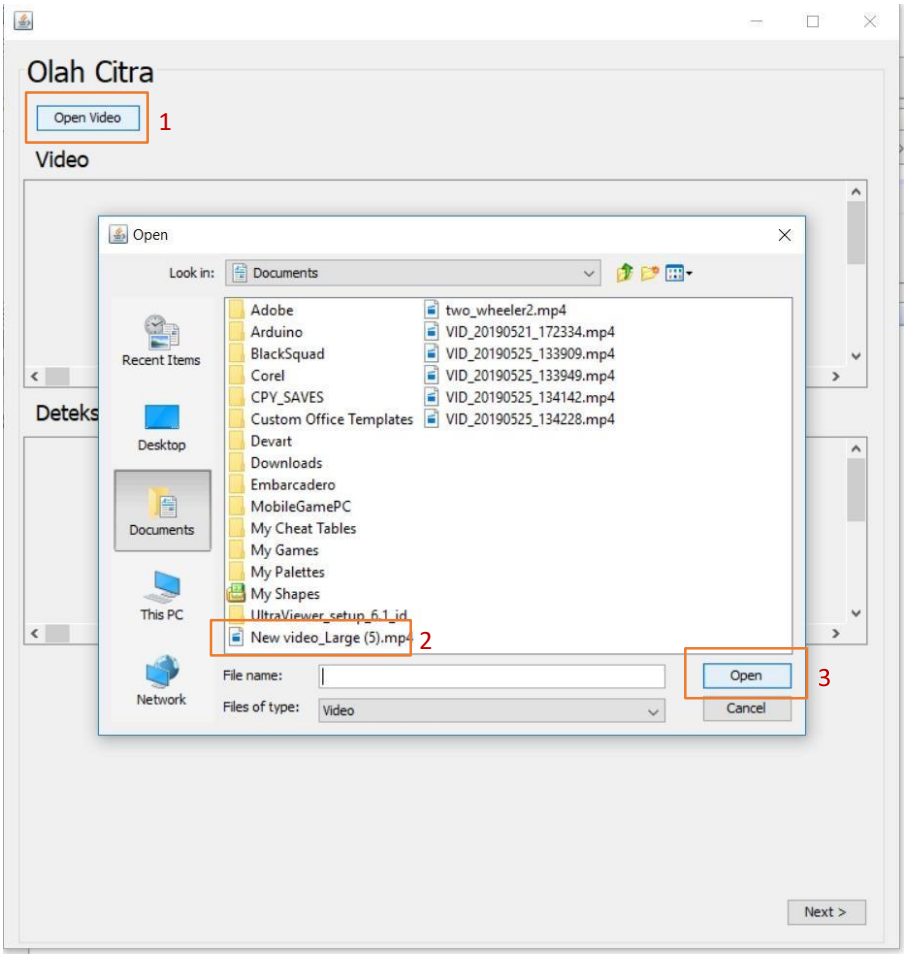



Gambar 76. Tampilan Form Ke-3 Aplikasi

Berdasarkan gambar-gambar diatas yang merupakan tampilan-tampilan form pada aplikasi ditunjukkan hasil implementasi dari aplikasi sistem identifikasi penggunaan helm pada pengendara sepeda motor sesuai dengan dari *design layout*yang telah dirancang sebelumnya. Adapun fungsi dari setiap tahapan sistem aplikasipengolahan citra digital yang terdapat pada form awal tersebut dengan alur sebagaiberikut :

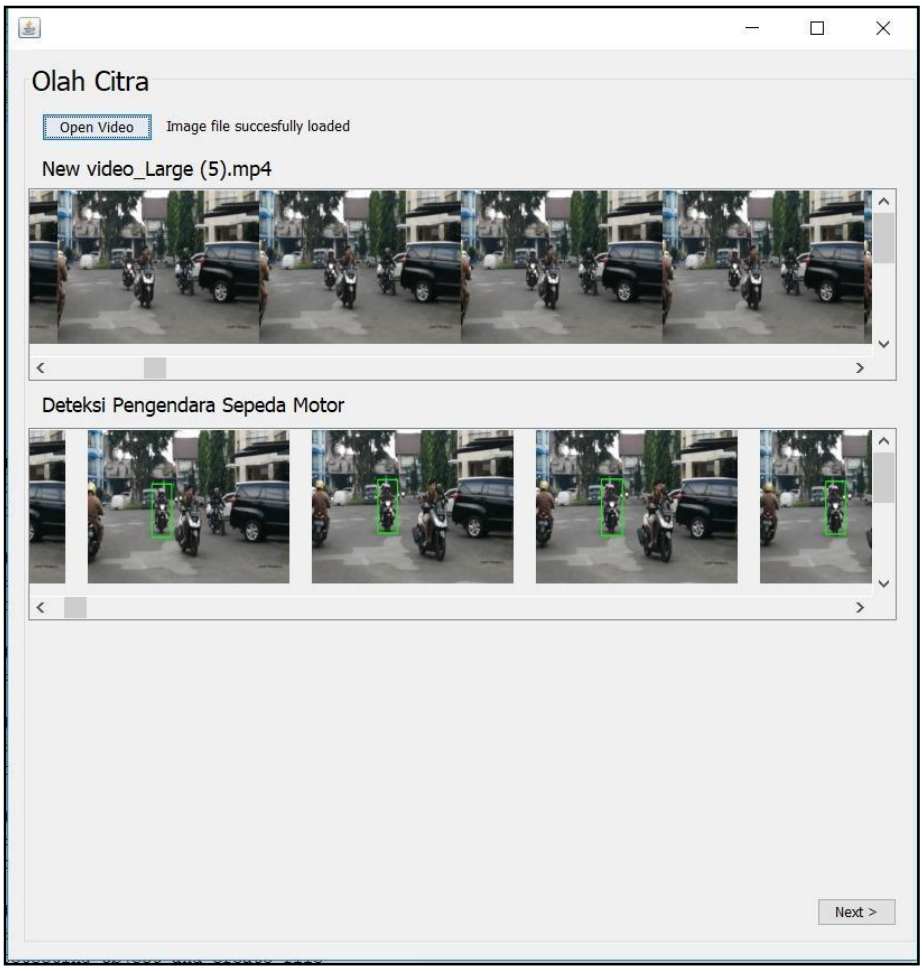
1. User dapat melakukan pemilihan data video yang akan diuji oleh sistem.

   Proses user dapat menekan tombol “Open Video” dan selanjutnya sistem akan menampilkan *path* yang dimana merupakan tempat file video disimpan. Kemudian user dapat memilih file video yang akan dijadikan sebagai masukan atau data uji pada aplikasi. Proses *open file* dan *load file* video pada aplikasi ditunjukkan oleh Gambar 77 sebagai berikut.



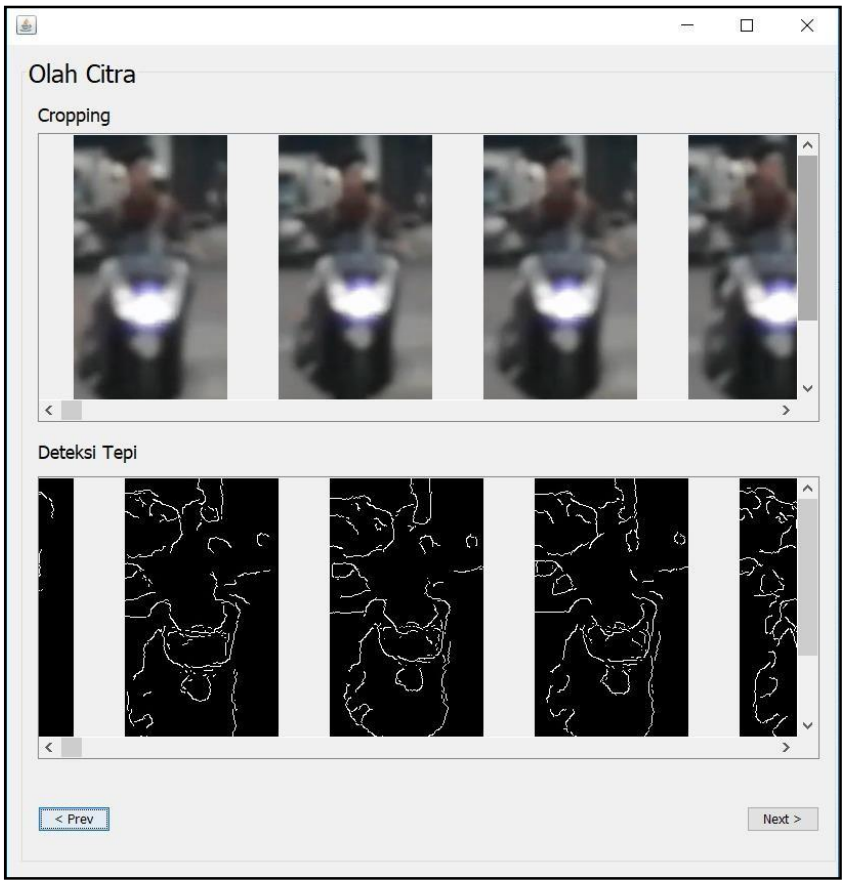

Gambar 77. Tampilan *Open* dan *Load File Video*

2. Sistem menampilkan *file name* video yang menjadi masukan pada aplikasi.Video yang menjadi data uji tersebut kemudia dilakukan proses *Frame Divider* atau melakukan eksktrak dari video menjadi frame-frame citra danakan menampilkan hasil tersebut pada panel. *Frame-frame* citra tersebut kemudia dilakukan proses *grayscalling* agar citra dari RGB dapat dikonversi menjadi citra keabuan, dimana hasil citra tersebut yang menjadi masukan dari proses selanjutnya. Proses kedua dilakukan yaitu melakukan proses deteksi objek pengendara sepeda motor dengan menggunakan Haar Cascade. Objek pengendara sepeda motor yang terdeteksi pada citra diberi tanda dengan bentuk persegi berwarna hijau yang ditampilkan pada panel deteksi pengendara sepeda motor. Berikut ini merupakan tampilan dari proses pada form pertama aplikasi ditunjukkan pada Gambar 78.




Gambar 78. Tampilan *Proses Frame Divider dan Haar Cascade*

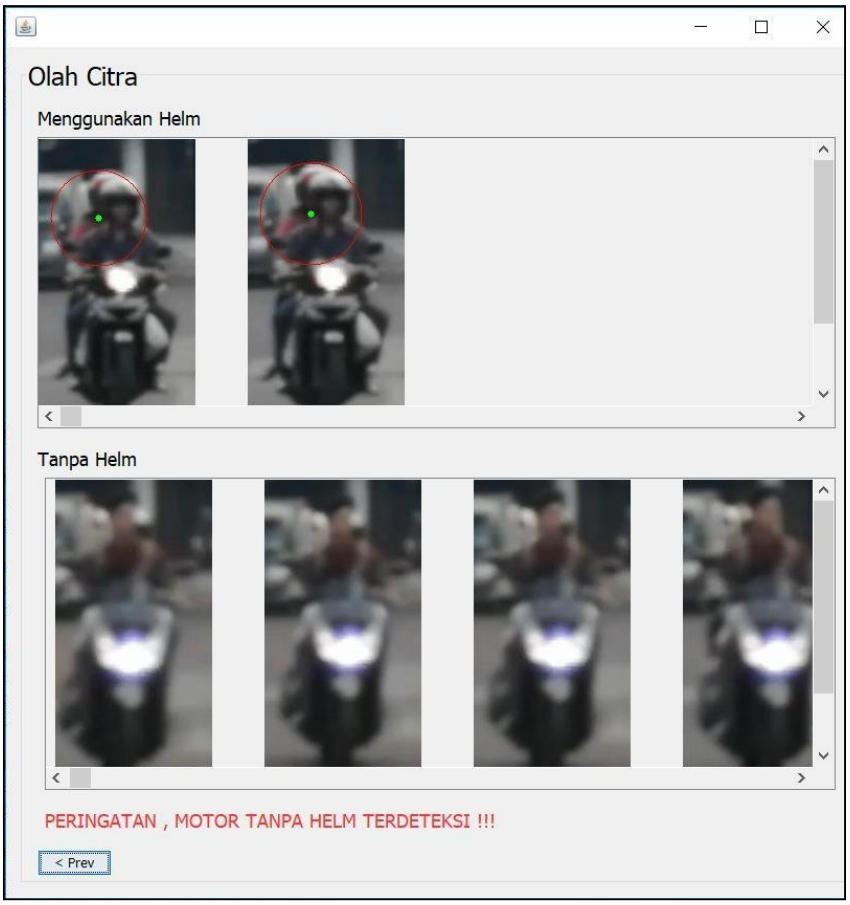
3. Berdasarkan tampilan *form* diatas terdapat tombol untuk menuju ke form selanjutnya yaitu *form* ke-2 aplikasi. Pada form ke-2 ini dilakukan dua proses yaitu proses *Cropping* untuk memotong bagian hanya objek pengendara sepeda motor pada citra dan akan menampilkan hasil tersebut pada panel *Cropping*. Proses selanjutnya yaitu dilakukan proses *Canny Edge Detection* untuk mendapatkan tepian dari objek citra dan akan menampilkan hasil proses tersebut pada panel deteksi tepi. Berikut ini merupakan tampilan dari proses pada form pertama aplikasi ditunjukkan pada Gambar 79.




Gambar 79. Tampilan Proses *Cropping* dan *Canny*

4. Berdasarkan tampilan *form* diatas terdapat dua buah tombol untuk melihat ke form sebelumnya dan ke form selanjutnya. Pada *form* ke-3 atau terakhirini dilakukan proses ROI untuk memfokuskan area hanya bagian atas pengendara sepeda motor yang diamati, dimana hasil dari proses ini akan dijadikan masukan pada proses selanjutnya yaitu *Hough Circle Transform* (HCT) untuk mendeteksi penggunaan helm. Pada panel pertama ditampilkan objek pengendara sepeda motor yang terdeteksi menggunakan helm dan ditandai dengan titik center lingkaran berwarna hijau dan lingkaran berwarna merah. Dengan terdeteksinya pengendara sepeda motoryang menggunakan helm, maka dapat diketahui mana saja pengendara yang tidak menggunkan helm dan akan menampilkannya pada panel kedua yang kemudian akan muncul peringatan karena terdapat pengendara sepeda

motor yang tidak mengenakan helm. Berikut ini merupakan tampilan dari proses pada form ke-3 aplikasi ditunjukkan pada Gambar 80.




Gambar 80. Tampilan Proses Deteksi Penggunaan Helm

### *4.3.2.* Pengujian *Alpha Frame Divider*

Proses pengujian alpha untuk fungsionalitas Frame Divider akan dijelaskanpada Tabel 2.

Tabel 5.Pengujian Alpha Frame Divider

| IDENTIFIKASI | |
|---|---|
| Nomor Uji | Uji-01 |
| Nama Butir Uji | Frame Divider |
| Tujuan | Melakukan ektraksi video menjadi frame-frame citra |
| Deskripsi | Sistem akan melakukan ektrak video yang berupa masukan menjadi frame-frame |

<table>
<tr><td>Kondisi Awal</td><td colspan="3">Masukan berupa video dalam format .mp4</td></tr>
<tr><td colspan="4">PENGUJIAN</td></tr>
<tr><td colspan="4">Skenario Uji</td></tr>
<tr><td colspan="4">1. Sistem menampilkan tampilan gui halaman awal yang terdapat tombol openvideo<br>2. Sistem menampilkan tampilan dialog box yang berisi path untuk pemilihandata yang akan diuji<br>3. Sistem melakukan proses frame divider<br>4. Sistem menyimpan frame-frame citra ke path dan menampilkannya pada panel</td></tr>
<tr><td colspan="4">Kriteria Evaluasi Uji</td></tr>
<tr><td colspan="4">Sistem melakukan proses Frame Divider dan menampilkan pada panel</td></tr>
<tr><td colspan="4">Kasus dan Hasil Uji</td></tr>
<tr><td>Masukan</td><td>Harapan</td><td>Pengamatan</td><td>Kesimpulan</td></tr>
<tr><td>Video</td><td>1. Sistem mengambil video dari path<br>2. Sietem melakukan ekstraksi video menjadi frame<br>3. Sistem menampilkan seluruh frame hasil ektraksi pada panel</td><td>1. Sistem mengambil videodari path<br>2. Sistem melakukan ekstraksi video menjadi frame<br>3. Sistem menampilkan seluruh frame hasil ektraksi pada panel</td><td>[X] Berhasil</td></tr>
<tr><td colspan="4">Source Code</td></tr>
</table>

```
// TODO add your handling code here:
int returnValue = openFileChooser.showOpenDialog(this);

if (returnValue == JFileChooser.APPROVE_OPTION){
    try{
        originalBI = ImageIO.read(openFileChooser.getSelectedFile());
        lb_namafile.setText(openFileChooser.getSelectedFile().getName());
        lbl_load.setText("Image file succesfully loaded");
        FrameDevider fd = new FrameDevider();
        fd.setPath(openFileChooser.getSelectedFile().getPath());
        fd.setMain(this);
        fd.start();
    } catch (IOException ioe){
        lbl_load.setText("Failed to load file");
    }
}
else {
    lbl_load.setText("No file choosen");
}
```

Output

### *4.3.3.* Pengujian *Alpha Haar Cascade*

Proses pengujian alpha untuk fungsionalitas Haar Cascade akan dijelaskanpada Tabel 3.

Tabel 6.Pengujian Alpha Haar Cascade

| IDENTIFIKASI | |
|---|---|
| Nomor Uji | Uji-02 |
| Nama Butir Uji | Haar Cascade |
| Tujuan | Melakukan proses Haar Cascade pada frame |
| Deskripsi | Sistem melakukan proses deteksi objek pengendara sepeda motor pada citra |
| Kondisi Awal | Masukan berupa citra *grayscale* pengendara sepeda motor di jalan |
| PENGUJIAN | |
| Skenario Uji | |
| 1. Sistem membaca path tempat frame-frame citra disimpan | |

2. Sistem melakukan konversi citra RGB menjadi citra *grayscale*
3. Sistem melakukan proses Haar Cascade
4. Sistem menggambar rectangle pada objek pengendara sepeda motor yang terdeteksi
5. Sistem menyimpan citra hasil deteksi objek pengendara sepeda motor ke path dan menampilkannya pada panel

Kriteria Evaluasi Uji

Sistem melakukan proses Haar Cascade dan menampilkan citra hasil pada panel

Kasus dan Hasil Uji

| Masukan | Harapan | Pengamatan | Kesimpulan |
|---|---|---|---|
| Citra *grayscale* pengendara sepeda motor | 1. Sistem melakukan proses Haar Cascade<br>2. Sistem menampilkan citra hasil deteksipada panel | 1. Sistem melakukan proses Haar Cascade<br>2. Sistem menampilkan citra hasil deteksi pada panel | [X] Berhasil |

Source Code

```
if(x == 50){
    Mat frame = new Mat();
    if (capturre.read(frame)){
        System.out.println("get frame");
    } else {
        System.out.println("get frame done");
        Capture = false;
        HaarCascade hc = new HaarCascade();
        if (!hc.isAlive()){
            hc.setMain(main);
            hc.setPath("hasil/");
            hc.setWritePath("OutputHaarCascade/");
            hc.setJumFile(y-1);
            hc.start();
        }
    }
    Imgcodecs.imwrite("hasil/DetectedObject"+y+".jpg",frame);
    ImageIcon image = new ImageIcon("hasil/DetectedObject"+y+".jpg");
    Image imagel = image.getImage();
    Image newImage = imagel.getScaledInstance(180, 137, java.awt.Image.SCALE_SMOOTH);
    image = new ImageIcon(newImage);
    main.updateUI(image,position);
```

Output

### 4.3.4. Pengujian Alpha Cropping

Proses pengujian alpha untuk fungsionalitas cropping akan dijelaskan pada

Tabel 4.

Tabel 7.Pengujian Alpha Cropping

| IDENTIFIKASI | | | |
|---|---|---|---|
| Nomor Uji | Uji-03 | | |
| Nama Butir Uji | Cropping | | |
| Tujuan | Melakukan proses Cropping pada citra hasil deteksi objek | | |
| Deskripsi | Sistem melakukan proses pemotongan citra hasil deteksi objek pengendara sepeda motor | | |
| Kondisi Awal | Masukan berupa citra hasil deteksi objek pengendara sepeda motor | | |
| PENGUJIAN | | | |
| Skenario Uji | | | |
| 1. Sistem membaca path tempat citra hasil Haar Cascade disimpan<br>2. Sistem melakukan proses Cropping pada citra<br>3. Sistem menyimpan citra Cropping ke path dan menampilkannya pada panel | | | |
| Kriteria Evaluasi Uji | | | |
| Sistem melakukan proses Cropping dan menampilkan citra hasil pada panel | | | |
| Kasus dan Hasil Uji | | | |
| Masukan | Harapan | Pengamatan | Kesimpulan |
| Citra hasil deteksi objek pengendara sepeda motor | 1. Sistem melakukan proses Cropping<br>2. Sistem menampilkan citra hasil pemotongan pada panel | 1. Sistem melakukan proses Cropping<br>2. Sistem menampilkan citra hasil pemotongan pada panel | [X] Berhasil |
| Source Code | | | |

```
int dataSegmentasi = 1;
for (Rect rect : imgDetections.toArray())
{
    Imgproc.rectangle(image, new Point(rect.x, rect.y),
     new Point(rect.x + rect.width, rect.y + rect.height),
                                    new Scalar(0, 255, 0),10);

    System.out.println(" xyz : "+rect.x +"-"+ rect.y+"-"+ rect.width+"-"+ rect.height);
    //imageSeg = grabCut2(x, imageSeg, rect.x, rect.y, rect.width, rect.height);
    if (y > 0){
        ImageIcon ImageIconSegmen = grabCut((dataSegmentasi), image, rect.x,rect.y,rect.width,rect.height, objekGambar);
        Image imagelSeg = ImageIconSegmen.getImage();
        Image newImageSeg = imagelSeg.getScaledInstance(180, 137, java.awt.Image.SCALE_SMOOTH);
        ImageIconSegmen = new ImageIcon(newImageSeg);
```

Output

### *4.3.5.* Pengujian *Alpha Canny Edge Detection*

Proses pengujian alpha untuk fungsionalitas Canny Edge Detection akandijelaskan pada Tabel 6.

Tabel 8.Pengujian Alpha Canny Edge Detection

| IDENTIFIKASI | |
|---|---|
| Nomor Uji | Uji-05 |
| Nama Butir Uji | Canny Edge Detection |
| Tujuan | Melakukan proses Canny Edge Detection yang dilakukan beberapa tahap |
| Deskripsi | Sistem melakukan proses Canny Edge Detection diawali dengan proses Gaussian Blur dan Thresholding |
| Kondisi Awal | Masukan berupa citra objek pengendara sepeda motor |
| PENGUJIAN | |
| Skenario Uji | |

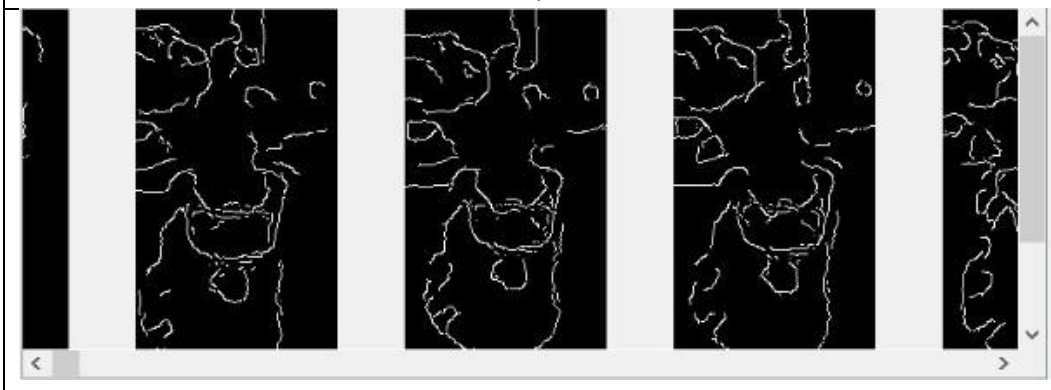
1. Sistem membaca path tempat citra objek bagian atas pengendara sepeda motor disimpan
2. Sistem melakukan proses Gaussian Blur
3. Sistem melakukan proses Thresholding
4. Sistem menyimpan citra hasil deteksi tepi ke path dan menampilkannya padapanel

| Kriteria Evaluasi Uji | | | |
|---|---|---|---|
| Sistem dapat melakukan proses Canny Edge Detection dan menampilkan citra hasil pada panel | | | |
| Kasus dan Hasil Uji | | | |
| Masukan | Harapan | Pengamatan | Kesimpulan |
| Citra objek pengendara sepeda motor | 1. Sistem dapat melakukan proses Canny Edge Detection<br>2. Sistem dapat menampilkan citra hasil pada panel | 1. Sistem dapat melakukan proses Canny Edge Detection<br>2. Sistem dapat menampilkan citra hasil pada panel | [X] Berhasil |

Source Code

```
Mat edges = new Mat();
int lowThreshold = 40;
int ratio = 2;
Imgproc.Canny(gray, edges, lowThreshold, lowThreshold * ratio);

Imgcodecs.imwrite("OutputDeteksiTepi/DetekTepi"+urutan+".jpg", edges);
```

Output



### *4.3.6.* Pengujian *Alpha Hough Circle Transform*

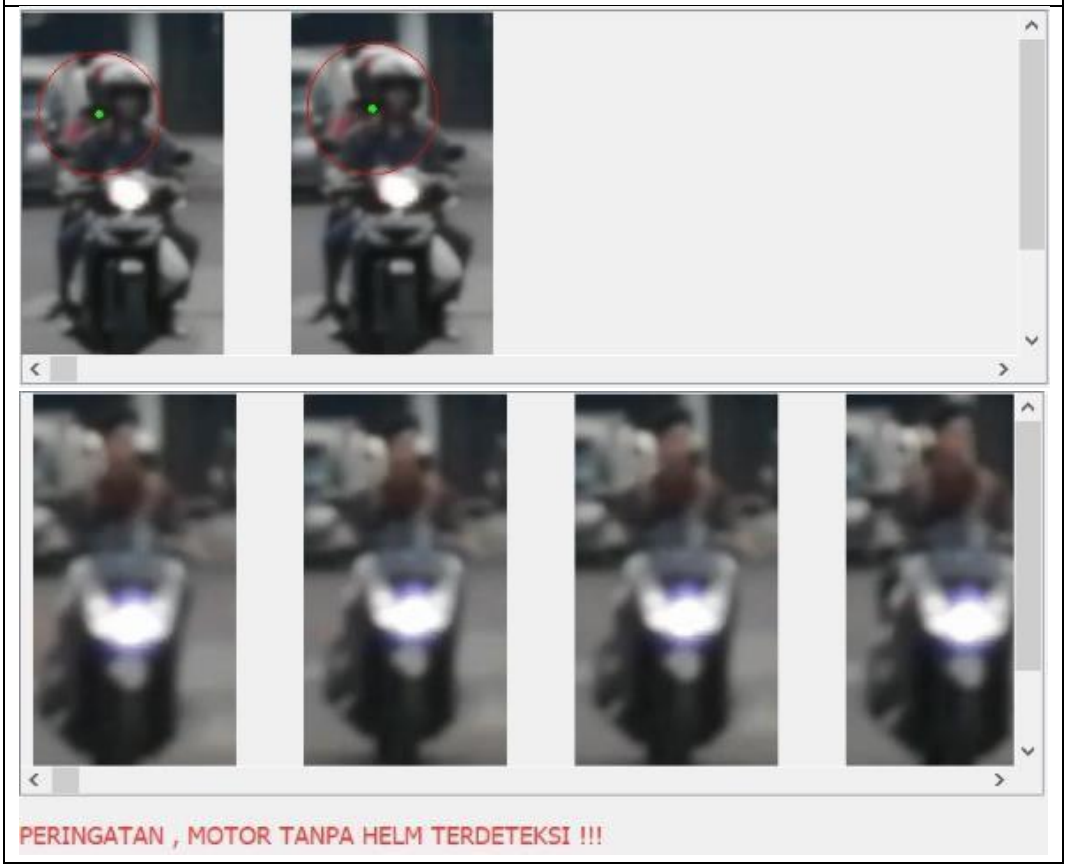
Proses pengujian alpha untuk fungsionalitas Hough Circle Transform akandijelaskan pada Tabel 7.

Tabel 9.Pengujian Alpha Hough Circle Transform

<table>
<tr><td colspan="4">IDENTIFIKASI</td></tr>
<tr><td>Nomor Uji</td><td colspan="3">Uji-06</td></tr>
<tr><td>Nama Butir Uji</td><td colspan="3">Hough Circle Transform</td></tr>
<tr><td>Tujuan</td><td colspan="3">Melakukan proses Hough Circle Transform pada citra</td></tr>
<tr><td>Deskripsi</td><td colspan="3">Melakukan proses deteksi lingkaran yang berupa bentuk dari helm</td></tr>
<tr><td>Kondisi Awal</td><td colspan="3">Masukan berupa citra hasil dari Region of Interest (ROI) yaitu tepi citra objek bagian atas pengendara sepeda motor pengendara sepeda motor</td></tr>
<tr><td colspan="4">PENGUJIAN</td></tr>
<tr><td colspan="4">Skenario Uji</td></tr>
<tr><td colspan="4">1. Sistem membaca path tempat citra hasil deteksi tepi disimpan<br>2. Sistem melakukan proses Hough Circle Transform<br>3. Sietem menggambar titik center dan lingkaran pada objek yang terdeteksi<br>4. Sistem menyimpan citra hasil objek hasil deteksi ke path dan menampilkannya pada panel</td></tr>
<tr><td colspan="4">Kriteria Evaluasi Uji</td></tr>
<tr><td colspan="4">Sistem dapat melakukan proses Hough Transfprm dan menampilkan cira hasil deteksi pada panel</td></tr>
<tr><td colspan="4">Kasus dan Hasil Uji</td></tr>
<tr><td>Masukan</td><td>Harapan</td><td>Pengamatan</td><td>Kesimpulan</td></tr>
<tr><td>Tepi citra objek bagian atas pengendara sepeda motor pengendara sepeda motor</td><td>1. Sistem dapat melakukan proses Hough Circle Transforn<br>2. Sistem dapat menampilkan citra hasil deteksi pada panel</td><td>1. Sistem dapat melakukan proses Hough Circle Transforn<br>2. Sistem dapat menampilkan citra hasil deteksi pada panel</td><td>[X] Berhasil</td></tr>
<tr><td colspan="4">Source Code</td></tr>
</table>

```
for( int i = 0; i < circles.rows(); i++ )
{
  double[] data = circles.get(i, 0);
  for(int j = 0 ; j < data.length ; j++){
       x = data[0];
       y = data[1];
       r = (int) data[2];
  }
  Point center = new Point(x,y);
  // circle center
  Imgproc.circle( img, center, 3, new Scalar(0,255,0), -1);
  // circle outline
  Imgproc.circle( img, center, r, new Scalar(0,0,255), 1);

    Imgcodecs.imwrite("OutputCircle/WearHelmet"+urutan+".jpg", img);
    ImageIcon imgHough = new ImageIcon("OutputCircle/WearHelmet"+urutan+".jpg");
    //ImageIcon imgCrop2 = new ImageIcon("OutputUpperObject/ObjekUpper"+y+".jpg");
    Image iCrop3 = imgHough.getImage();
    Image newICrop3 = iCrop3.getScaledInstance(180, 137, java.awt.Image.SCALE_SMOOTH);
    imgHough= new ImageIcon(newICrop3);
```

Output



### 4.4. Pengujian Akurasi Sistem

Pengujian yang dilakukan adalah dengan menghitung performansi aplikasi berdasarkan beberapa parameter dan tingkat akurasi sistem yang dirancang. Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan. Uji coba dilakukan untuk mengenali penggunaan helm melalui input video. Akurasi dari setiap pengujian yang dilakukan pada sistem identifikasi penggunaan helm pada pengendara sepeda motor ini dihitung menggunakan persamaan sebagai berikut :

$$Akurasi = \frac{Total\ Benar}{Total\ Uji}\ x\ 100\ \% \quad \text{.....(7)}$$

Keterangan :

Total Benar : Total dari data yang benar Total

Uji : Total dari gambar yang diujikan

#### 4.4.1. Skenario Pengujian 1

Pada skenario ini dilakukan pengujian untuk mengetahui tingkat akurasi dari sistem berdasarkan pengaruh waktu pengambilan data pengujian dilakukan yaitu pada waktu pagi, siang dan sore, serta diberikan dua kondisi pada helm yaituuntuk helm *full helmet* dan *half helmet* yang keduanya merupakan helm ber SNI (Sandar Nasional Indonesia). Pengujian yang dilakukan untuk mengetahui pada waktu kapan pengujian sistem ini dapat maksimal menemukan objek helm pada pengendara sepeda motor.

Tabel 10. Pengujian Akurasi Pengaruh Waktu

| Kondisi | Pengujian | Hasil yang Diharapkan | Waktu | | |
|---|---|---|---|---|---|
| | | | Pagi | Siang | Sore |
| Full Helmet | 1 | √ | √ | X | √ |
| | 2 | √ | √ | √ | X |
| | 3 | √ | √ | √ | X |
| | 4 | √ | X | X | X |
| | 5 | √ | √ | √ | √ |
| Akurasi | | | $\frac{4}{5}x100\%$ = 80% | $\frac{3}{5}x100\%$ = 60% | $\frac{2}{5}x100\%$ = 40% |

| | | | | | |
|---|---|---|---|---|---|
| Half Helmet | 1 | √ | √ | X | √ |
| | 2 | √ | X | √ | X |
| | 3 | √ | √ | √ | √ |
| | 4 | √ | √ | √ | √ |
| | 5 | √ | √ | X | X |
| Akurasi | | | $\frac{4}{5}x100\%$ = 80% | $\frac{3}{5}x100\%$ = 60% | $\frac{3}{5}x100\%$ = 60% |

Keterangan :

√ = Terdeteksi

X = Tidak terdeteksi

Berdasarkan data hasil pengujian yang ditujukan pada tabel, dapat diketahuinilai keberhasilan dari terdeteksinya helm SNI dengan bentuk *full helmet* pada kondisi pagi, siang dan sore adalah 80%, 60% dan 40%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm SNI bentuk *full helmet*, maka didapatkan presentase kegagalannya masing-masing sebesar 20%, 40% dan 60%. Hal ini dapat disebabkan oleh setiap kondisi waktu memiliki pencahayaan yang berbeda dan hal tersebut mempengaruhi kelima data yang akan diuji.

Sedangkan nilai keberhasilan dari terdeteksinya helm SNI dengan bentuk *half helmet* pada kondisi pagi, siang dan sore adalah 80%, 60% dan 60%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm SNI bentuk *full helmet*, maka didapatkan presentase kegagalannya masing-masing sebesar 20%, 40% dan 40%. Sama halnya seperti penjelasan sebelumnya, hal ini dapat disebabkan oleh setiap kondisi waktu memiliki pencahayaan yang berbeda dan haltersebut mempengaruhi kelima data yang akan diuji.

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untukkondisi pagi hari adalah $\frac{8}{10}x100\%$ = 80%.

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untukkondisi siang hari adalah $\frac{6}{10}x100\%$ = 60%.

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untukkondisi malam hari adalah $\frac{5}{10}x100\%$ = 20%.

### 4.4.2. Skenario Pengujian 2

Pada skenario ini dilakukan pengujian untuk mengetahui tingkat akurasi dari sistem berdasarkan pengaruh sudut pengambilan data pengujian dilakukan yaitu pada dudut 0°, 45° dan 90°, serta diberikan dua kondisi pada pada pengendarasepeda motor yaitu untuk tunggal dan berboncengan. Pengujian yang dilakukan untuk mengetahui pada sudut berapa pengujian sistem ini dapat maksimal menemukan objek helm pada pengendara sepeda motor.

Tabel 11. Pengujian Akurasi Pengaruh Sudut

| Kondisi | Pengujian | Hasil yang Diharapkan | Sudut | | |
|---|---|---|---|---|---|
| | | | 0° | 45° | 90° |
| Tunggal | 1 | √ | √ | X | X |
| | 2 | √ | X | √ | X |
| | 3 | √ | √ | X | X |
| | 4 | √ | √ | √ | X |
| | 5 | √ | √ | X | X |
| Akurasi | | | $\frac{4}{5}x100\%$ = 80% | $\frac{2}{5}x100\%$ = 40% | $\frac{0}{5}x100\%$ =0% |
| Berboncengan | 1 | √ | X | X | X |
| | 2 | √ | √ | X | X |
| | 3 | √ | √ | √ | X |
| | 4 | √ | X | X | X |
| | 5 | √ | √ | X | X |
| Akurasi | | | $\frac{3}{5}x100\%$ = 60% | $\frac{1}{5}x100\%$ = 20% | $\frac{0}{5}x100\%$ = 0% |

Keterangan :

√ = Terdeteksi

X = Tidak terdeteksi

Berdasarkan data hasil pengujian yang ditujukan pada tabel, dapat diketahuinilai keberhasilan dari terdeteksinya helm pada pengendara sepeda motor yang tunggal berdasarkan sudut pengambilan data adalah 80%, 40% dan 0%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm SNI bentuk *full helmet*, maka didapatkan presentase kegagalannya masing-masing sebesar 20%, 60% dan 100%. Hal ini dapat disebabkan pada tahap deteksi pengendara sepeda motor dengan *Haar Cascade*. Objek pengendara sepeda motor tidak dapat dikenali

maka sistem tidak dapat melanjutkan tahap deteksi helm pada pengendara sepeda motor. Hal ini pun dapat disebabkan karena kurangnya data latih yang dibuat untuk tahap *Haar Cascade* terutama pada sudut 90° atau pengambilan data dari samping objek. Selain itu, dapat juga dikarenakan pada sudut tertentu akan mempengaruhi pendeteksian helm itu sendiri karena bentuk helm yang tidak memiliki bentuk lingkaran yang diinginkan jadi sistem tidak dapat mendeteksinya sebagi objek helm.

Sedngkan nilai keberhasilan dari terdeteksinya helm pada pengendara sepeda motor yang berboncengan berdasarkan sudut pengambilan data adalah 60%,20% dan 0%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm SNI bentuk *full helmet*, maka didapatkan presentase kegagalannya masing-masing sebesar 40%, 80% dan 100%. Sama halnya seperti penjelasan sebelumnya dapat disebabkan pada tahap deteksi pengendara sepeda motor dengan *Haar Cascade*. Objek pengendara sepeda motor tidak dapat dikenali maka sistem tidak dapat melanjutkan tahap deteksi helm pada pengendara sepeda motor. Hal ini pun dapat disebabkan karena kurangnya data latih yang dibuat untuk tahap *Haar Cascade* terutama pada sudut 90° atau pengambilan data dari samping objek. Selain itu, dapat jugadikarenakan pada sudut tertentu akan mempengaruhi pendeteksian helm itu sendiri karenabentuk helm yang tidak memiliki bentuk lingkaran yang diinginkan jadi sistem tidak dapatmendeteksinya sebagi objek helm

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untuk sudut 0° adalah $\frac{7}{10} x100\% = 70\%$.

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untuksudut 45° adalah $\frac{3}{10} x100\% = 30\%$.

Jadi, dapat diambil kesimpulan bahwa persentase keseluruhan pengujian untuksudut 90° adalah $\frac{0}{10} x100\% = 0\%$.

### 4.4.3. Skenario Pengujian 3

Pada skenario ini dilakukan pengujian untuk mengetahui tingkat akurasi dari sistem dengan kondisi yang diberikan yaitu helm yang bukan SNI dan objek bukan helm. Pengujian yang dilakukan untuk mengetahui apa dengan diberikannya

kondisi tersebut sistem ini dapat maksimal menemukan objek helm bukan SNI atauobjek bukan helm pada pengendara sepeda motor.

Tabel 12. Pengujian Akurasi Bukan Helm SNI dan Kepala

| Kondisi | Pengujian | Hasil yang DIharapkan | Hasil | Akurasi |
|---|---|---|---|---|
| Helm Bukan SNI | 1 | X | X | $\frac{3}{5}x100\%$= 60% |
| | 2 | X | X | |
| | 3 | X | √ | |
| | 4 | X | √ | |
| | 5 | X | X | |
| Kepala | 1 | X | X | $\frac{4}{5}x80\%$= 80% – |
| | 2 | X | X | |
| | 3 | X | √ | |
| | 4 | X | X | |
| | 5 | X | X | |

Keterangan :

√ = Terdeteksi

X = Tidak terdeteksi

Berdasarkan data hasil pengujian yang ditujukan pada tabel, dapat diketahuinilai keberhasilan dari tidak terdeteksinya helm yang bukan SNI adalah 60%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm bukan SNI sebesar 60%, maka didapatkan presentase kegagalannya sebesar 40%. Hal ini dapatdisebabkan oleh dua dari lima citra helm data uji yang merupakan bukan helm SNI,memiliki dasar bentuk lingkaran.

Sedangkan data hasil pengujian yang ditujukan pada tabel, dapat diketahui nilai keberhasilan dari tidak terdeteksinya bukan helm atau kepala adalah sebagai berikut 100%. Apabila dilihat dari presentase keberhasilan tidak terdeteksinya helm(kepala) yaitu sebesar 100%, maka tidak didapatkan presentase kegagalannya. Halini ddikarenakan kepala yang tidak berbentuk sebagai lingkaan atau tidak memilikiunsur bentuk lingkaran.

# BAB V
# PENUTUP

## 5.1 Kesimpulan

Berdasarkan hasil pengujian sistem identifikasi penggunaan helm pada pengendara sepeda motor, diperoleh kesimpulan sebagai berikut :

1. Beradasarkan penelitian dan pengujian yang telah dilakukan, dapat diambil kesimpulan bahwa pengujian alpha pada aplikasi dengan tahapan-tahapan yaitu *Frame Divider*, *Grayscalling*, *Haar Cascade Classifier*, *Cropping*, *Canny Edge Detection*, *Region of Interest* dan *Hough Circle Transform* berhasil diterapkan dan dijalankan oleh sistem.
2. Dari hasil pengujian akurasi pada kondisi waktu pagi hari, siang hari, sore hari didapatkan persentase keselurahan tingkat nilai akurasi sebesar 80%, 60% dan 20% . Berdasarkan persentase tersebut, aplikasi lebih optimal mendeteksi pada waktu pagi hari.
3. Dari hasil pengujian akurasi sistem pada sudut sudut 0°, sudut 45° dan sudut sudut 90° didapatkan persentase keselurahan tingkat nilai akurasi sebesar 70%, 30% dan 0% . Berdasarkan persentase tersebut, aplikasi lebihoptimal mendeteksi pada sudut 0° atau dari depan.
4. Dari hasil pengujian akurasi sistem untuk mendeteksi bukan helm SNI mendapatkan persentase sebesar 80% dan bukan helm (kepala)

# DAFTAR PUSTAKA


Andrew, Buliali, J. L., & Wijaya, A. Y. (2017). Deteksi Kecepatan Kendaraan Berjalan di Jalan Menggunakan OpenCV . *Jurnal Teknik ITS.*

Auliannisa, R. D., Suratman S.T., M. F., & Rizal ST., M. (2017). Deteksi Katarak Menggunakan Metode Transformasi Hough Berbasis Android . *e- Proceeding of Engineering*, 3310.

Christina, M. (2017, November 3). *Tak Pakai Helm, Pelajar di Merangin Dihukum Baca Pancasila*. Retrieved from iNews.id: https://www.inews.id/daerah/regional/1348/tak-pakai-helm-pelajar-di-merangin-dihukum-baca-pancasila

Herlambang, A. S., Nurhayati, O. D., & Martono, K. T. (2016). Sistem Pendeteksi Kualitas Daging Dengan Ekualisasi Histogram Dan Thresholding Berbasis Android. *Jurnal Teknologi dan Sistem Komputer*.

Jandrisno, I. (2017). *Pemanfaatan Background Substraction Untuk Mendeteksi Kepadatan Lalu Lintas Berbasis Raspberry Pi.* Padang: Politeknik Negeri Padang.

Khotimah, C., & Juniati, D. (2017). Pengenalan Iris Mata Menggunakan Ekstraksi Fitur Dimensi Fraktal Box Counting. *MATHunesa*.

Lazaro, A., Buliali, J. L., & Amaliah, B. (2016). Deteksi Jenis Kendaraandi Jalan Menggunakan OpenCV. *JURNAL TEKNIK ITS Vol. 6 No. 2*, 2337-3520 .

Lestari, Q. A., Gerhana, Y. A., & Wahana, A. (2017). Implementasi Metode Circle Hough Transform pada Aplikasi Pendamping Belajar Huruf Braille.*ISSN*, 222-227 .

Li, K., & Zhao, X. (2017). Deteksi Penggunaan Helm Pengaman Berbasis Pengolahan Citra Dan Machine Learning. *IEEE*.

Mufrod. (2018, Mei 16). *5 Ribu Pengendara Motor Tewas Tak Gunakan Helm, Masih Mau Nekat?* Retrieved from Okezone.com: https://news.okezone.com/read/2018/05/15/15/1898745/5-ribu-pengendara-motor-tewas-tak-gunakan-helm-masih-mau-nekat

Mustafid, A., & 'Uyun, S. (2017). Segmentasi Citra Sapi Berbasis Deteksi Tepi Menggunakan Algoritma Canny Edge Detection. *Jurnal Buana Informatika*.

Naufal, M. A. (2017). *Implementasi Metode Klasifikasi K-Nearest Neighbour (K-NN) Untuk Pengenalan Pola Batik Motif Lampung.* Bandar Lampung: Universitas Lampung.

Palit, A. D. (2017). Mengenali Rambu Lalu Lintas Menggunakan Metode HOG dan KNN. *Jurnal Ilmiah Teknologi Informatika Terapan (JITTER)*.

Pammungkas, E. M., Sumbodo, B. A., & Candradewi, I. (2017). Sistem Pendeteksi dan Pelacakan Bola dengan Metode Hough Circle Transform, Blob Detection, dan Camshift Menggunakan AR.Drone . *IJEIS*, 1-12.

Pathasu, D., & Klubsuwan, K. (2016). Deteksi Penggunaan Helm Setengah dan Penuh di Thailand Menggunakan Haar Like Feature dan Circle Hough Transform pada Pengolahan Citra. *IEEE International Conference on Computer and Information Technology*.

Samir, M. I., Zuraiyah, M.Kom, T. A., & Aryani, M.Cs, A. S. (2017). Penerapan Algoritma Background Substraction Untuk Tracking dan Klasifikasi Kendaraan .

Sangeetha, D., & Deepa, P. (2016). Implementasi FPGA dari Algoritma Robust Canny Edge Detection yang Hemat Biaya. *Springer*.

Sarikan, S. S., Ozbayoglu, A. M., & Zilci, O. (2017). Automatisasi Klasifikasi Kendaraan Dengan Pengolahan Citra Dan Computational Intelligent. *Procedia Computer Science*, 515-522.

Septadina, I. S. (2015). Identifikasi Individu dan Jenis Kelamin Berdasarkan Pola Sidik Bibir.

Setiawan, P. R., Setyawan, F. A., Alam, S., & Sulistiyanti, S. R. (2018). Rancang Bangun Model Deteksi Pelanggaran Zebra Cross Pada Traffic Light Menggunakan Metode Adaptive Background Substraction.

Silva , R. R., Aires, K. R., & Veras, R. d. (2017). Deteksi Helm pada Pengendara Sepeda Motor. *Spinger*.

Undang-Undang Lalu Lintas Angkutan Jalan Tahun 2009, No. 22 Pasal 106 Ayat 8.