# LAPORAN SISTEM PENDETEKSI DAN MENGHITUNG KENDARAAN YANG LEWAT MENGGUNAKAN ALGORITMA HAAR CASCADE CLASSIFIER

## BAB I (Pendahuluan)

Pertumbuhan populasi dan urbanisasi yang pesat di seluruh dunia telah menyebabkan peningkatan lalu lintas kendaraan yang signifikan. Hal ini menimbulkan tantangan dalam pengaturan lalu lintas, keamanan jalan, dan pengumpulan data transportasi. Untuk mengatasi tantangan ini, penggunaan sistem pendeteksi dan penghitung kendaraan menjadi sangat penting. Sistem semacam itu dapat memberikan informasi yang berharga kepada pihak berwenang, membantu mengoptimalkan pengaturan lalu lintas, serta memberikan pemahaman yang lebih baik tentang pola pergerakan kendaraan.

Algoritma Haar Cascade Classifier adalah salah satu metode populer yang digunakan dalam sistem pendeteksi objek. Algoritma ini telah terbukti efektif dalam mendeteksi berbagai objek, termasuk kendaraan, dalam gambar dan video dengan tingkat akurasi yang tinggi. Dalam laporan ini, kami akan menjelaskan latar belakang dan implementasi sistem pendeteksi dan penghitung kendaraan menggunakan algoritma Haar Cascade Classifier.

Pendekatan menggunakan algoritma Haar Cascade Classifier ini memiliki keunggulan dalam hal kecepatan pemrosesan dan kinerja yang baik dalam mendeteksi objek di berbagai kondisi pencahayaan. Metode ini memungkinkan deteksi kendaraan secara real-time, yang sangat diperlukan dalam aplikasi yang membutuhkan respons cepat, seperti sistem pengaturan lalu lintas dan pemantauan jalan raya.

Dalam laporan ini, kami akan membahas secara rinci tentang algoritma Haar Cascade Classifier dan langkah-langkah yang diperlukan untuk mengimplementasikan sistem pendeteksi dan penghitung kendaraan. Kami juga akan menjelaskan teknik penghitungan jumlah kendaraan yang melewati suatu titik atau wilayah tertentu setelah pendeteksian dilakukan. Selain itu, kami akan membahas proses pelatihan model Haar Cascade Classifier menggunakan dataset yang relevan untuk memastikan akurasi deteksi yang optimal.

Diharapkan laporan ini dapat memberikan pemahaman yang komprehensif tentang penggunaan algoritma Haar Cascade Classifier dalam sistem pendeteksi dan penghitung kendaraan. Sistem ini memiliki potensi besar untuk meningkatkan keamanan jalan, efisiensi lalu lintas, dan pemantauan transportasi secara keseluruhan.

## BAB II (Methode Yang Digunakan)

Algoritma Haar Cascade Classifier adalah algoritma yang digunakan untuk pendeteksian objek dalam citra atau video. Algoritma ini dikembangkan berdasarkan metode Viola-Jones pada tahun 2001 oleh Paul Viola dan Michael Jones. Algoritma Haar Cascade Classifier terkenal karena kecepatan eksekusinya yang tinggi dan kinerja pendeteksian yang baik.

Prinsip dasar dari algoritma Haar Cascade Classifier adalah penggunaan fitur-fitur Haar untuk mengidentifikasi objek dalam citra. Fitur-fitur Haar adalah fitur visual yang diekstraksi dari citra berdasarkan perbedaan intensitas piksel pada berbagai area dalam citra. Fitur-fitur Haar ini berbentuk kotak-kotak dengan pola piksel gelap dan terang yang menunjukkan perbedaan intensitas.

Proses pendeteksian menggunakan algoritma Haar Cascade Classifier terdiri dari beberapa tahapan:

1. Ekstraksi Fitur Haar: Pada tahap ini, fitur-fitur Haar diekstraksi dari citra. Fitur-fitur Haar ini berbentuk kotak-kotak dengan pola piksel gelap dan terang. Untuk setiap fitur Haar, dilakukan perhitungan yang melibatkan operasi penjumlahan dan pengurangan intensitas piksel di dalam kotak. Ekstraksi fitur ini memungkinkan algoritma untuk mempelajari pola-pola yang khas dari objek yang ingin dideteksi.
2. Pelatihan dengan Metode Boosting: Algoritma Haar Cascade Classifier menggunakan metode Boosting, seperti Adaboost, untuk melatih model klasifikasi. Pada tahap pelatihan, algoritma melakukan iterasi melalui dataset pelatihan yang terdiri dari citra objek positif dan citra latar belakang negatif. Setiap iterasi, algoritma mengubah bobot data latih untuk memfokuskan pada citra yang masih sulit dideteksi. Selanjutnya, algoritma membangun kumpulan klasifier yang secara bertahap memperbaiki performa pendeteksian.
3. Pembuatan Cascade: Setelah dilakukan pelatihan dengan metode Boosting, algoritma membentuk "cascade" dari klasifier. Cascade adalah serangkaian klasifier yang diterapkan secara berurutan. Setiap tahap cascade memiliki beberapa klasifier dan setiap klasifier memiliki tingkat kompleksitas yang meningkat. Cascade digunakan untuk mengurangi jumlah daerah yang harus diperiksa dalam citra, sehingga mempercepat proses pendeteksian.
4. Deteksi Objek: Pada tahap ini, algoritma menerapkan cascade klasifier pada citra atau frame video yang ingin dideteksi objeknya. Proses deteksi dilakukan dengan menjalankan serangkaian filter yang diterapkan pada setiap bagian citra. Jika suatu bagian citra cocok dengan kriteria filter, maka bagian tersebut dianggap sebagai objek yang dideteksi.

Algoritma Haar Cascade Classifier memiliki beberapa kelebihan, antara lain:

- Kecepatan eksekusi yang tinggi: Algoritma ini dapat bekerja secara cepat dan efisien, sehingga cocok digunakan dalam aplikasi real-time yang membutuhkan deteksi objek dalam waktu nyata.
- Kinerja pendeteksian yang baik: Algoritma ini memiliki kinerja yang baik dalam mendeteksi objek, termasuk kendaraan, wajah, atau objek lainnya dalam citra atau video.
- Toleransi terhadap variasi pencahayaan: Algoritma ini cukup toleran terhadap variasi pencahayaan dalam citra, sehingga tetap mampu mendeteksi objek dengan baik dalam berbagai kondisi pencahayaan.

Namun, algoritma Haar Cascade Classifier juga memiliki beberapa keterbatasan, seperti rentan terhadap perubahan posisi dan rotasi objek, serta ketergantungan pada kualitas dan variasi data latih.

**BAB III (Implementasi Algoritma)**

Pada bab ini, akan dijelaskan mengenai implementasi algoritma dalam Sistem Menghitung Kendaraan yang Lewat menggunakan algoritma Haar Cascade Classifier. Implementasi ini menggunakan bahasa pemrograman Python dan library OpenCV.

A. Persiapan Implementasi
   1. Import Library: Mulai dengan mengimport library yang diperlukan, yaitu cv2 (OpenCV) untuk pengolahan citra dan video, serta sleep dari library time untuk mengatur delay antar frame.

```
import cv2
from time import sleep
```

   2. Inisialisasi Variabel: Tentukan variabel yang akan digunakan, seperti lokasi file cascade_xml (cascade_src) yang berisi model Haar Cascade Classifier untuk deteksi kendaraan, dan lokasi video yang akan diproses (video_src). Juga, tentukan variabel untuk mengatur delay antara frame (delay), posisi garis batas untuk menghitung kendaraan (pos_line), jarak toleransi dari garis batas (offset), dan variabel untuk menghitung jumlah kendaraan (car).

```
cascade_src = 'cars.xml'
video_src = 'video.mp4'

delay= 2000
detec = []
pos_line=600
offset=10
car= 0
```

3. Definisi Fungsi: Buat fungsi *center object* untuk menghitung titik tengah objek deteksi kendaraan berdasarkan koordinat dan dimensi kotak pembatas (x, y, w, h).

```
def center_object(x, y, w, h):
    x1 = int(w / 2)
    y1 = int(h / 2)
    cx = x + x1
    cy = y + y1
    return cx,cy
```

B. Pembacaan Video dan Deteksi Kendaraan

1. Baca Video: Gunakan cv2.VideoCapture() untuk membaca video dari lokasi yang ditentukan oleh video_src.

```
cap = cv2.VideoCapture(video_src)
```

2. Deteksi Kendaraan: Gunakan cv2.CascadeClassifier() untuk membuat objek cascade classifier dengan model yang diambil dari cascade_src. Kemudian, gunakan loop while untuk membaca setiap frame video. Terapkan delay antara frame menggunakan sleep.

```
car_cascade = cv2.CascadeClassifier(cascade_src)

while True:
    ret, img = cap.read()
    time = float(1/delay)
    sleep(time)
```

3. Konversi dan Deteksi: Konversi setiap frame menjadi citra grayscale menggunakan cv2.cvtColor(). Gunakan metode detectMultiScale dari cascade classifier untuk mendeteksi kendaraan dalam citra grayscale. Hasil deteksi akan berupa array yang berisi koordinat dan dimensi kotak pembatas untuk setiap kendaraan.

```
    if (type(img) == type(None)):
        break

    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    cars = car_cascade.detectMultiScale(gray, 1.1, 1)
```

C. Penghitungan Kendaraan dan Tampilan Visual

1. Garis Batas dan Kotak Pembatas: Gambar garis batas pada frame menggunakan cv2.line(). Loop melalui setiap kendaraan yang terdeteksi dan gambar kotak pembatas menggunakan cv2.rectangle(). Hitung titik tengah objek dan simpan dalam array detect. Gambar titik tengah menggunakan cv2.circle().

```
for (x,y,w,h) in cars:
    cv2.rectangle(img,(x,y),(x+w,y+h),(0,255,0),2)
    center = center_object(x, y, w, h)
    detec.append(center)
    cv2.circle(img, center, 4, (0, 0,255), -1)
```

2. Penghitungan Kendaraan: Periksa apakah titik tengah kendaraan berada di kisaran offset dari garis batas. Jika iya, tambahkan jumlah kendaraan (car) dan gambar garis batas dengan warna yang berbeda.

```
if center[1]<(pos_line+offset) and center[1]>(pos_line-offset):
    car+=1
    cv2.line(img, (25, pos_line), (1200, pos_line), (0,127,255), 3)
```

3. Tampilan dan Teks: Tambahkan teks pada frame yang menampilkan jumlah kendaraan menggunakan cv2.putText(). Tampilkan frame video yang telah diolah menggunakan cv2.imshow().

```
cv2.putText(img, "Kendaraan Lewat : "+str(car), (450, 70), cv2.FONT_HERSHEY_SIMPLEX, 2, (0, 0, 255),5)
cv2.imshow('video', img)
```

4. Loop dan Interaksi Pengguna: Perbarui frame berikutnya dengan membaca frame baru menggunakan cap.read(). Loop akan berjalan hingga pengguna menekan tombol 'q' pada keyboard untuk keluar.

```
if cv2.waitKey(1)  & 0xFF == ord('q'):
    break
```

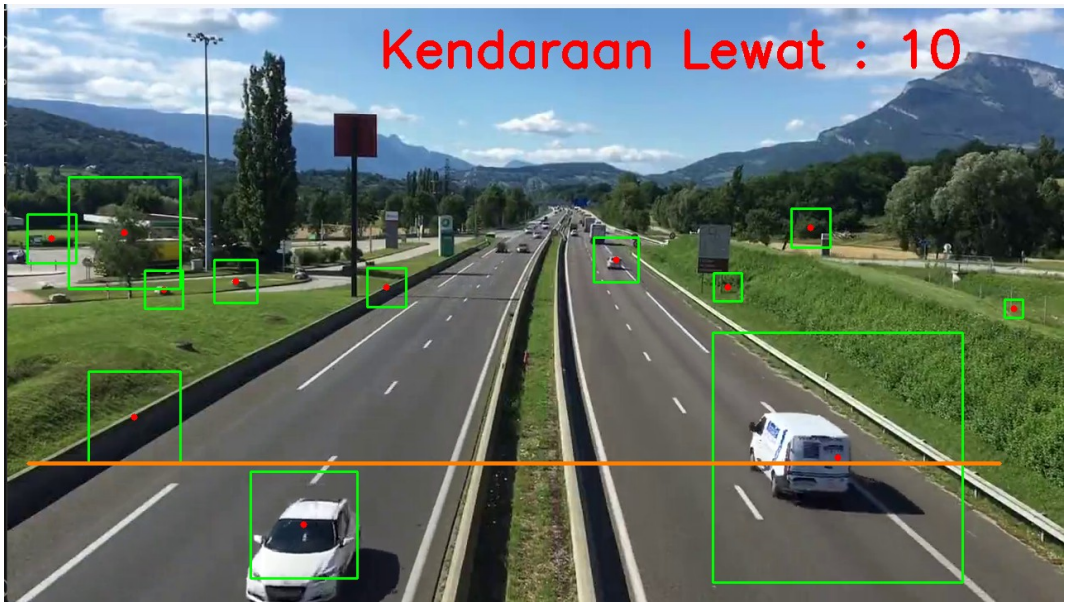
D. Penutup Implementasi

Setelah loop selesai, gunakan cv2.destroyAllWindows() untuk menutup semua jendela tampilan yang terbuka.

```
cv2.destroyAllWindows()
```

E. Mock up Aplikasi

Tampilan ketika sistem dibuka




**BAB IV (Evaluasi)**

Sejauh ini Pengimplementasian project di atas hanya melalui video yang di ambil dari youtube belum bisa dilakukan secara real menggunakan kamera cctv secara langsung karena keterbatasan perangkat.

Hasil evaluasi tentang Sistem Menghitung Kendaraan Yang Lewat Menggunakan Algoritma Haar Cascade Classifier adalah sebagai berikut:

1. Akurasi Penghitungan: Evaluasi dilakukan untuk mengukur seberapa akurat sistem dalam menghitung jumlah kendaraan yang melewati suatu garis. Hal ini dapat dilakukan dengan membandingkan hasil penghitungan sistem dengan jumlah kendaraan yang tercatat secara manual. Hasil akurasinya adalah dari 60 kendaraan yang lewat terdeteksi 54 atau akurasinya sekitar 90%.

2. Deteksi dan Identifikasi Jenis Kendaraan: Selain menghitung jumlah kendaraan, evaluasi juga dapat melibatkan kemampuan sistem dalam mendeteksi dan mengidentifikasi jenis kendaraan yang melewati area atau garis. Sistem ini berhasil mengenali berbagai jenis kendaraan, seperti mobil, motor, truk, dan lain-lain. Tetapi, untuk benda yang menyerupai kendaraan masih terdeteksi oleh sistem.

3. Kesalahan Penghitungan: Evaluasi juga harus memperhitungkan tingkat kesalahan sistem dalam menghitung kendaraan. Kesalahan penghitungan dapat berupa kesalahan over-count (menghitung kendaraan yang sebenarnya tidak melewati area) atau kesalahan under-count (kendaraan yang tidak terhitung). Kesalahan dari sistem ini terjadi karena posisi video/ kamera yang tidak stabil dan adanya objek yang terdeteksi menyerupai kendaraan menyentuh garis penghitung atau mobil berada di samping truk sehingga tidak terlihat/ terhitung oleh sistem.