

A Differential Evolution Based Multiclass Vehicle Detector and Classifier for Urban Environments

*Deepak Dawar, Department of Computer Science, North Dakota State University, USA
Simone A. Ludwig, Department of Computer Science, North Dakota State University, USA*

ABSTRACT

Video analytics is emerging as a high potential area supplementing intelligent transportation systems (ITSs) with wide ranging applications from traffic flow analysis to surveillance. Object detection and classification, as a sub part of a video analytical system, could potentially help transportation agencies to analyze and respond to traffic incidents in real time, plan for possible future cascading events, or use the classification data to design better roads. This work presents a specialized vehicle classification system for urban environments. The system is targeted at the analysis of vehicles, especially trucks, in urban two lane traffic, to empower local transportation agencies to decide on the road width and thickness. We present a hybrid appearance model specifically designed for speedy foreground extraction in the given context. We use a simple motion cue based tracking algorithm, and stay clear of using probabilistic trackers. The main thrust is on the accurate detection and classification of the detected objects using an evolutionary algorithm. The detector is backed by a differential evolution (DE) based discrete parameter optimizer. We show that, though employing DE proves expensive in terms of computational cycles, it measurably improves the accuracy of the detection and hence the classification system. The system was tested on multiple real video footage during varied weather conditions from a camera mounted in urban areas achieving a peak classification accuracy of approximately 90%.

Keywords: Video analytics, vehicle classification, differential evolution, Hough transform

INTRODUCTION

Vehicle detection and classification is currently a hot focus area in the myriad web of intelligent transportation systems (ITSs) with immense potential for traffic flow control, security, and surveillance to name a few. ITSs are embracing data driven techniques (Zhang et al., 2011) wherein the data related to traffic density, incidents, etc. are relayed back to users of the traffic systems thereby empowering them to make real-time decisions about routes thereby promoting efficiency. Road side regulations, increasing density of vehicles on roads, and costs of overlaying the roads are some of the supplementary and rather critical reasons calling for ever more efficient utilization of our transportation networks. Robust vehicle classification systems that are able to compute the number and type of vehicles plying on a particular road or highway, provide a part of the solution.

Various vehicle detection and classification systems such as digital wave radars (Sharma et al., 2008), amplitude modulated laser radars (Mao et al., 2012), lidars (Levinson et al., 2011), magnetometers (Canoga, 2003) etc., have been proposed and some have been in continual use commercially with their inherent advantages and disadvantages. These systems can be broadly classified depending upon the type of sensors they use or on the basis of their installation vis a vis intrusive and non-intrusive. Most commonly used systems, use one or a combination of laser, piezoelectric, microwave, or video cameras. For example, inductive loop detectors are one of the most accurate and widely used systems for vehicle detection. But due to their intrusive nature, high equipment and installation costs, they are not often being applied (Middleton, Chara, & Longmire, 2009).

Video based vehicle classification has recently emerged as a low cost alternative to conventional intrusive systems primarily owing to their low cost, non-intrusive nature of installation, and operation. For example, there has been extensive use of video based vehicle detection systems in surveillance (Tian et al., 2015), (Sivaraman & Trivedi, 2013), (Wang, 2013). Other benefits include (Linda & Volling, 2003):

- Freedom from extensive sawing residue and extensive cleaning after installation and continual maintenance.
- Installation can be done year round.
- No need for road closure for installation and maintenance thereby reducing the impact on traffic flow.
- Ease of operation.
- Provides rich data through which additional information and context can be approximated.
- Vision based systems integrate well with other object detection systems.

Their advantages notwithstanding, video based detection and classification systems have their own challenges, and pose many difficulties for researchers. Some of them may be categorized as (Tian et al., 2015):

- Occlusion: Vehicles and other objects on roads can block each other in the camera view leading to a false count.
- Rapid illumination changes: Weather changes may range from periods of very high brightness to very low intensities posing a problem for detection algorithms.
- Vehicle edge/contour deformation: Due to various continual changes in scene, the vehicle may not be detected with a perfect boundary leading to misclassifications.
- Multiplicity of vehicle types: There are highly varied types of vehicles ranging from a mid-sized sedan to transport trucks with different lengths, heights, and axles in between. Extracting individual features and correctly classifying vehicles becomes a challenge in this scenario. Add to that, pedestrian, bicycles and other objects, the problem becomes even more difficult.

There are other barrages of problems that specific stages of the classification systems try to solve, and will be discussed later in this work.

For solving complex real world problems, researchers have been looking into optimization techniques inspired by natural processes such as Darwinian evolution, social group behavior and foraging strategies. Nature-inspired search and optimization algorithms have seen a

significant growth in the past few decades. There are two main categories of such algorithms: evolutionary computing methods and swarm intelligence algorithms. Evolutionary computing methods employ a population of individual solutions and uses an iterative process to improve/evolve the solutions to achieve a certain goal under given constraints. Swarm intelligence on the other hand is characterized by the collective behavior of decentralized and self-organized systems, representing a population of simple entities that interact locally with one another and with their environment.

We propose a novel DE based vehicle detector and classifier (DEVEC) capable of classifying vehicles on highway and urban areas. DEVEC has a conventional vehicle detection architecture with the difference that an evolutionary algorithm (DE) is used for classification of vehicles using multiple cues including the axle count. We use Hough Transform, a parameter based feature detection method, to detect the axles. The quality of the detected circles is sensitive to appropriate settings of these parameters. Since the process is time consuming, it is not viable to adjust these parameters manually every time, thus, there is always a motivation to do a parameter search by attaching a machine learning algorithm to discover an optimized set.

Differential Evolution (DE), first proposed in the work (Storn & Price, 1995), is a robust real parameter optimizer in the family of evolutionary algorithms. DE has become quite popular lately and has been subjected to rigorous analysis in the past decade. It has been applied to a multitude of benchmark problems to ascertain its efficacy, and at the same time has proved quite effective in solving a broad range of real life scientific and engineering problems (Das & Suganthan, 2011). DE secured first position among evolutionary algorithms at the First International Contest on Evolutionary Optimization in May 1996 (Storn & Price, 1996). One of the major reasons for its popularity lies in its simplicity as it works with a few control parameters namely the scaling factor (F), the crossover rate (Cr), and the population size (NP).

We employ a modified DE algorithm as a discrete parameter optimizer to find the best suited parameters for accurate axle detection, which is a crucial part of our vehicle classification system. We show that the use of DE, apart from removing the need for setting Hough Transform parameters manually, also has the added advantage of improving the accuracy of the axle detection module, thereby improving the robustness of the overall classification system. On the other side, the process of finding the optimal Hough Transform parameters does add an extra computational cost making leading to the trade-off between speed and accuracy.

The motivation of this work is three-fold:

- As a proof of concept, we classify vehicles primarily based on the number of axles and distance between by mounting the camera sideways instead of a lamppost where the camera is conventionally mounted on the top.
- Assessing the utility of including axle count as a feature for classification aids in discrimination of trucks having similar geometric features but have distinct axle count.
- To maximize the model function representing the scene under consideration, we add an evolutionary algorithm namely Differential Evolution (DE).

To our knowledge, no other vehicle detection and classification algorithm make use of an evolutionary algorithm detecting axles for vehicle classification.

The rest of the paper is organized as follows. Section 2 describes a review of related work to tackle the vehicle detection and classification problem. In Section 3, we describe our DE based vehicle detector and classifier (DEVEC). Section 4 describes the experimental setup and shows the results obtained, and Section 5 concludes the paper with our findings.

RELATED WORK

A typical video based vehicle detection system is shown in Fig 1. First we provide a review of different techniques proposed to tackle problems associated with each stage.

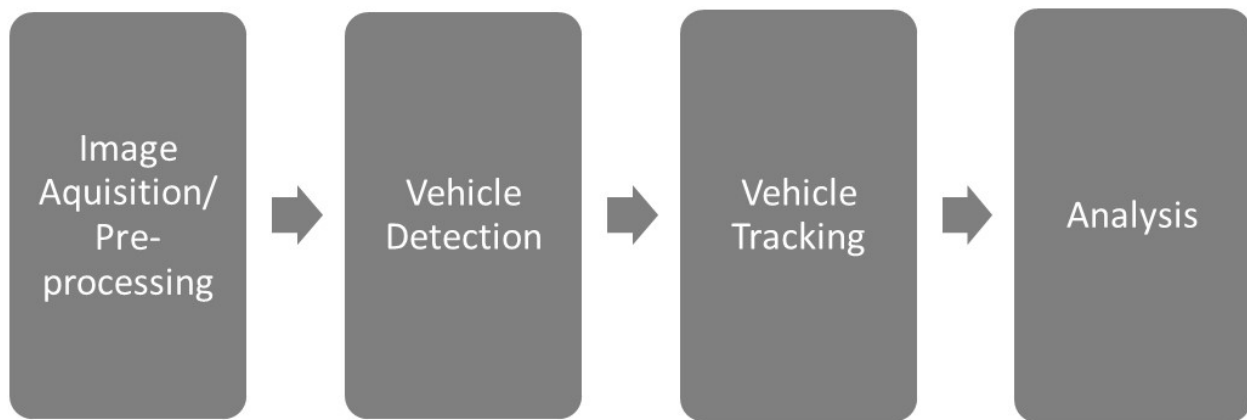


Figure 1: A typical vehicle detection system

Vehicle Detection

Detection is the primary step towards analysis of videos in an intelligent transportation system. The robustness of this step is quite critical as it feeds the higher sub systems like vehicle tracking and classification. The vehicle detection problem, in particular, has been approached through many different ways, and multiple methods have been proposed to achieve detection. These methods can be broadly classified into two classes: motion based and appearance based (Tian et al., 2015).

Motion based detection methods aim to extract the vehicle information based on a comparison of the present pixel state with an assumed stationary (background) state of the system. The easiest way to perform motion based detection is frame differencing (Li & He, 2011), (Park, Lee, & Park, 2007), (Nguyen & Le, 2008) wherein the pixel wise thresholded difference is calculated between two consecutive frames furnishing the output in terms of motion pixels, or what is generally referred to as the foreground. Although simple, this technique needs supplementation with other methods for dynamic motion, and the use of more information apart from just the difference of pixel is desirable.

Background subtraction is another common motion based segmentation technique and one of the more widely used (Gupte et al., 2002). This method tries to build a background model

of the scene based on the accumulated information. This background is then compared with the current video frame giving the motion information. There are other methods of background construction that do not assume a fixed background beforehand. For example, the background was constructed assuming a single Gaussian distribution in (Kumar, Ranganath, & Weimin, 2003), (Morris & Trivedi, 2008). Every pixel is either classified as background or foreground based on its distribution.

Other popular motion based segmentation techniques available in literature are Gaussian mixture models (GMM) (Stauffer & Grimson, 1999), (Zheng et al., 2006), (Sen-Ching & Kamath, 2004), (Haque, Mursheed, & Paul, 2008), (Zivkovic & Heijden, 2006), (Greggio et al., 2010), (Zhao & Lee, 2009), median filter (Cucchiara et al., 2003), (Mcfarlane & Schofield, 1995), kernel density estimation (Elgammal, Harwood, & Davis, 2000), kalman filtering (Kalman, 1960), (Karmann, Brandt, and Gerl, 1990), and optical flow (Ottlik & Nagel, 2008), (Indu, Gupta, & Bhattacharyya, 2011).

Appearance based detectors, in contrast with motion based detectors, use appearance features like color, texture, shape, etc., to extract the object of interest, in this case vehicles from the image or video directly. Coded descriptors based on features are utilized to model the appearance of vehicles. Local edge operators have been used in (Agarwal, Avan, & Roth, 2004), (Ma & Grimson, 2005). Recently, more sophisticated and robust feature descriptors have been proposed, detailed explanation of which, can be found in the recent survey paper (Smadi et al., 2016).

Vehicle Tracking

Vehicle tracking is essentially a state prediction and data association problem. The idea is to recognize the vehicle in subsequent frames, locate its position, and ultimately obtain its trajectory. Vehicle tracking is sometimes merged with the detection task but may also be performed separately. There are many methods available in this literature that cater to this task and these methods can be broadly classified into three categories: model based, region based, and feature based tracking.

Model based methods presume a predetermined 2-D or 3-D vehicle appearance model matching the presumed model with regions of motion in the sequence (Ottlik & Nagel, 2008). A multi view 3-D model that builds a 3-D model based on 2-D geometrical information was constructed using edge features by (Koller, Weber, & Malik). Other 3-D modeling techniques were proposed in (Lou et al., 2005) and (Buch, Velastin, & Orwell, 2011).

Region based tracing on the other hand, aims to detect a vehicle's silhouette contained within a geometric shape represented by multiple features like area, length, width, centroid, etc. The vehicle may be represented in terms of a feature vector with continuous updation. This vector is tracked through shape matching or data association in subsequent frames. To track highway vehicles, a graph based matching approach was used in (Lai, Huang, & Tseng, 2010). The information such as length and height of the convex hull was used in (Buch, Velastin, & Orwell, 2011) to track the vehicles, while in (Morris & Trivedi, 2008) centroid and velocity information was used.

Feature based tracking is based on the combined use of simple features like edges with feature descriptors like SIFT, SURF, HOG, etc. For example, region based tracking was combined with SIFT in (Wang & Hong, 2012). Main advantages of feature based tracking is its ability to perform well in crowded areas but the challenge lies in choosing effective features.

Vehicle Classification

Classification requires vehicles to be associated with a particular class. This has been achieved in the literature either by using shape features (height, aspect ratio, etc.) or appearance of the vehicle. The number of features used has a direct impact on the number of discriminant classes, and nature of the classifier needed.

Authors in (Han et al., 2005) classify vehicles into car, SUV, and minibus by making use of the curve information associated with 3-D ridges. Their classifier achieved an accuracy of 88%. A similar 3-D model based classifier was constructed in (Chen, Ellis, & Velastin, 2012) capable of discrimination between car, bus, van, and motorcycles with a classification accuracy of approximately 96%. Based on the contour information a classifier based on a combination a voting algorithm and Euclidian distance was proposed in (Negri et al., 2006) achieving a classification rate of 93%. Appearance based classifiers use information based on gradients, corners, etc. In (Zafar et al., 2007), a 2-D LDA technique was employed capable of classifying 25 vehicle types with an accuracy of approximately 91%.

PROPOSED APPROACH: DIFFERENTIAL EVOLUTION BASED VEHICLE DETECTOR AND CLASSIFIER (DEVEC)

This work presents a Differential Evolution based vehicle detector and classifier (DEVEC) primarily designed for urban two lane traffic, but may be extended for multi lanes with a few modifications. A black box view of DEVEC is presented in Fig. 2. A detailed description of the individual stages is presented below.

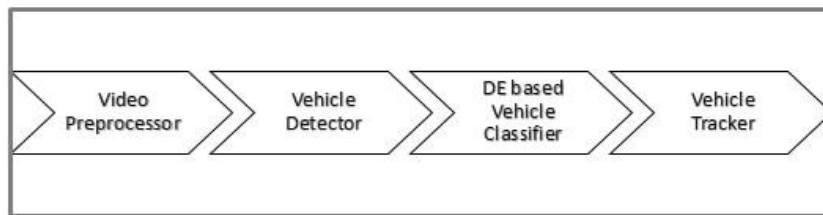


Figure 2: Differential evolution based vehicle detector and classifier (DEVEC)

Video Preprocessor

This is an optional sub-system. The main utility of this module is to reduce the video size (frame size) from the recorded/captured resolution to the one set by the user. The higher the resolution of the video, the greater is the computational cost. A low resolution video, however, will be detrimental in achieving good detection accuracy. Thus, the frame resolution should be

kept within an acceptable range. Another optional sub-feature of this module is to automatically identify the road under consideration thereby extracting the region of interest (ROI) containing the potential vehicles. The subsequent steps operate on ROI extracted from the video frames.

Vehicle Detector

This component constitutes the backbone of the whole system. The reason for this claim is because correct vehicle classification is as good as vehicle detection. Vehicle detection, in our system, primarily consists of two basic steps: background construction/modeling and detection.

In motion based segmentation and detection models, as in ours, a robust background plays a vital role in appropriate detection, as the motion pixels are extracted from the scene by subtracting it with the modeled background. There are many different background models for vehicle detection available in literature based on different techniques. (Kanhere & Birchfield, 2008), (Chen & Zhang, 2006), (Wren et al., 2006) used single Gaussian to model the background while recently authors in (Barcellos et al., 2015) used a mixture of Gaussians. A consolidated review of such research was presented in a recent survey paper (Smadi et al., 2016).

We propose a pixel-to-pixel based adaptive background construction model. Without assuming any distribution information about the background, this model compares the color RGB information of every pixel in the background with the current scene, and calculates a three dimensional Chi-Square metric. This metric is then compared with a threshold, which is determined adaptively and is continuously updated. This process is described in Algorithm 1.

Algorithm BACKGROUND

Choose initial n training frames

Initialize distance vector V , with size n

Initialize threshold T as 0

For every frame i till $n-1$ frames

1) Compute mean M , and standard deviation SD , for RGB components of frame i

2) Update vector V as

$$V[i] = M + SD$$

End For

Compute T as:

$$T = \text{mean}(V) + \text{standard_dev}(V)$$

Update frame n as current background $B_{current}$

Update frame n as previous background B_{prev}

While video has frames

For every pixel p in frame I and background $B_{current}$

1) Compute Chi-Square distance CS

2) If $CS \geq T$

Classify p as **foreground**

Else

Classify p as **background**

End For

Update foreground pixel locations in $B_{current}$ from B_{prev}

If no vehicle detected for 3 consecutive frames

Recalculate T and update current frame as $B_{current}$

EndIf

End While

End Algorithm

Algorithm 1: DEVEC adaptive background construction

To calculate the initial threshold T , n training frames are required. For every two subsequent frames I and $I-1$ in the training set, the Chi-Square distances of R, G, and B components are calculated as:

$$R_T = \sum_N \frac{(R_I - R_{I-1})^2}{R_I} \quad (1)$$

$$G_T = \sum_N \frac{(G_I - G_{I-1})^2}{G_I} \quad (2)$$

$$B_T = \sum_N \frac{(B_I - B_{I-1})^2}{B_I} \quad (3)$$

Since we presume no information about the background color, all the differences are equally weighted to calculate the gradient of pixel differences G as:

$$G = w_R \times R_T + w_G \times G_T + w_B \times B_T \quad (4)$$

with $w_R = w_G = w_B = 0.33$. Based on the assumptions of scene color, different weights may be assigned to different color components but we presume no such information.

After the initial gradient threshold G is determined, a new background is adaptively constructed. Figure 3 presents the adaptive determination of G for the first 72 frames of a particular video sequence. A gradient threshold G is then determined for every subsequent frame. Every current frame I is compared with the previous frame $I-1$ pixel by pixel, and differences between RGB components for every pixel p are calculated as:

$$RGB_{diff}(p) = w_R \times (R_I - R_{I-1}) + w_G \times (G_I - G_{I-1}) + w_B \times (B_I - B_{I-1}) \quad (5)$$

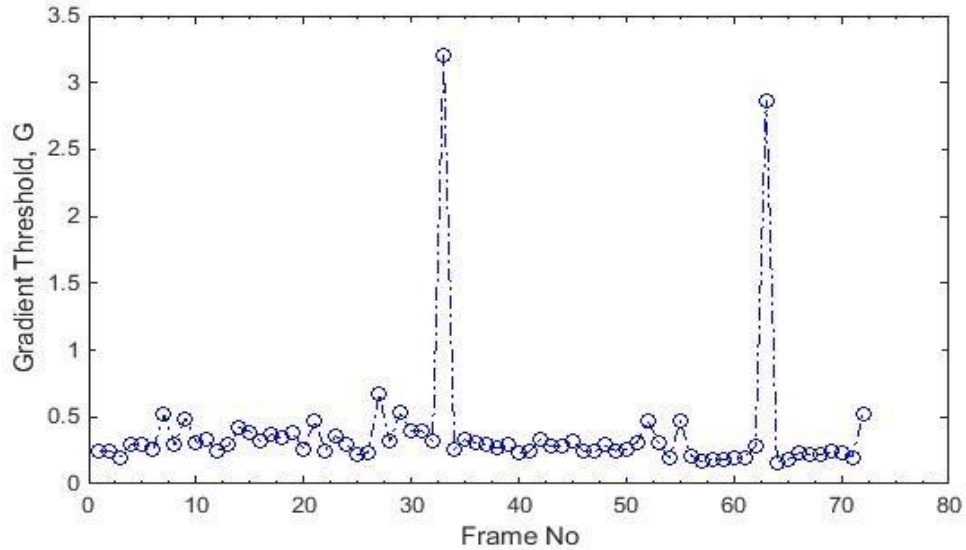


Figure 3: Change of gradient threshold during adaptive threshold calculation and update

If $RGB_{diff}(p)$ for a pixel p exceeds the pixel gradient G , p is classified as a foreground pixel else it is considered as a background pixel. In the current form every pixel difference is compared with G , and a decision is made about the pixel's class. Another version of this model can be constructed in which every pixel's own gradient can be calculated and stored but this would require extra computation and storage. For our purposes the current form will suffice.

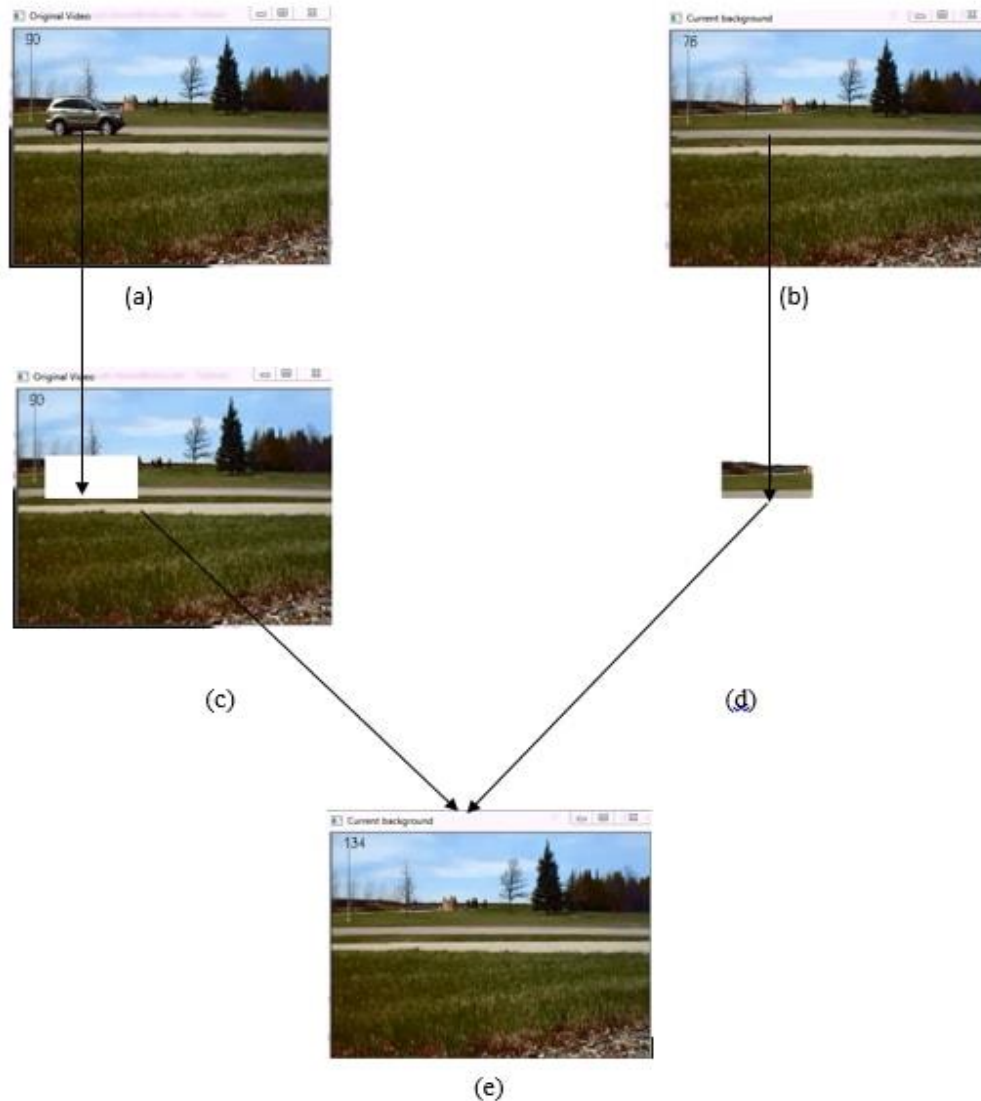


Figure 4: A step by step description of adaptive background construction

A previous background is maintained in the memory, which provides the pixels to current background under construction at locations where foreground pixels are detected. In this way the current background is updated continuously, and describes the current scene robustly. Figure 4 depicts this process of adaptive background construction while classifying the vehicles as foreground and background.

The pixel gradient threshold G is updated after x_{update} number of frames, which is a user defined number. As a word of caution, x_{update} should be under acceptable limits, as a low value

of x_{update} would mean very high frequency of threshold calculation and update, which may degrade the performance of the system.

After the background construction, the next step is to detect vehicles from the scene using motion segmentation. Background subtraction is used to identify motion pixels as contours. With motion pixels, undesired noise is most certainly always detected. We use a simple noise suppressor algorithm to remove the unwanted noise, which increases the cost marginally in terms of computation cycles but has a desirable impact on the system's fidelity. After that, a bounding box is fit around the detected contours. During the detection and tracking process, there is always a possibility that the vehicle being tracked splits up into different parts. For example, a single vehicle may split up into two unrelated and distinct parts, or a sub area capable of being identified as a new vehicle, or is formed inside the tracked vehicle itself, etc. Our algorithm is capable of resolving such anomalies and boost accurate detection.

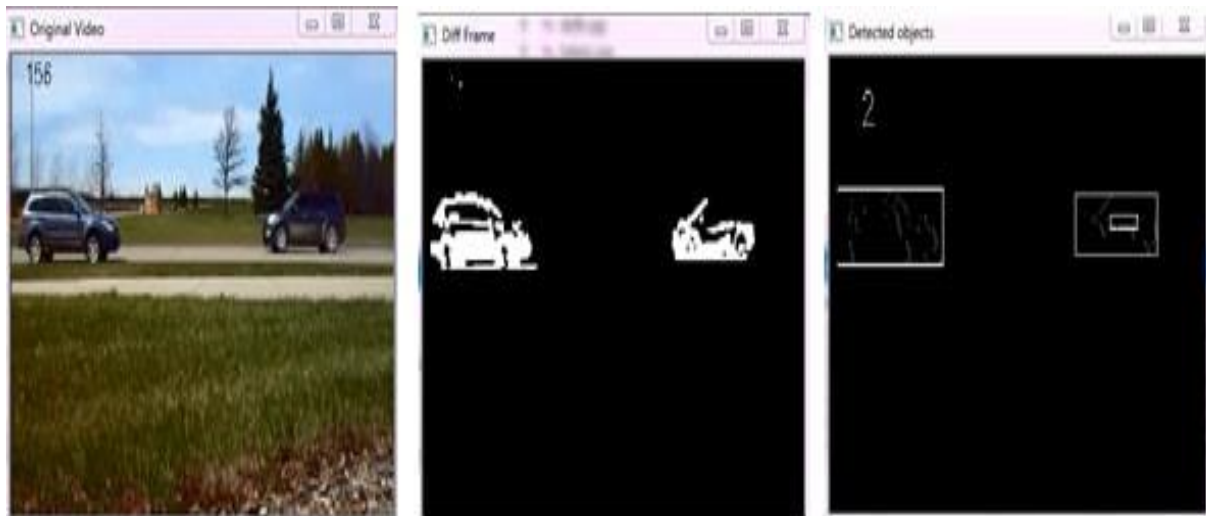


Figure 5: Vehicle detection process

In essence, the bounding box creation is a hypothesis generation step. We employ a dual hypothesis verification process with soft and hard matches. As an initial soft match, this bounding box is matched with a basic car/truck template. If this criterion is satisfied, this patch is sent to a DE enabled axle detector for a hard match. We define a soft match as positive if only the basic feature matching threshold requirements are met which in this case are length, breadth, and contour area. A hard match involves determining the number of axles in the vehicle, and minimizing a vehicle model fitness function using DE. This two-step process improves the robustness of the detection module though at the expense of extra compute cycles. Algorithm 2 depicts this detection process.

Algorithm DETECT

```
    While video has frames
        Perform background subtraction
        Extract foreground pixels
        Clean the background with a noise suppressor
        Find contours in the cleaned background
        Fit bounding boxes around contours
        Soft match the boxes with a basic vehicle template
        If soft match successful
            Send the object for a DE enabled axle detector
        End If
    End While
End Algorithm
```

Algorithm 2: Vehicle detection process in DEVEC

Classification

After the soft detection of the vehicle that only checks for a minimum contour area, length and breadth of the bounding box, the relevant region is extracted from the current color frame, and sent to the DE based vehicle classifier module. The main emphasis of using this novel module is to investigate the feasibility of using axles to classify vehicles. Identifying axles in an image is essentially a circle detection problem. Circle detection holds high significance in image analysis as is evident from its vast applications in the manufacturing goods industry, military, etc. (Costa & Cesar, 2000). This problem has been tackled with different approaches most common of which are:

- Deterministic - Hough Transform based methods (Yuen et al., 1989).
- Geometric Hashing and template matching (Iivarinen et al., 1997), (Jones et al., 1990).
- Stochastic - Simulated annealing (Bongiovanni, Crescenzi, & C. Guerra, 1995), Genetic Algorithms (GA) (Roth & Levine, 1994) etc.

After weighing the pros and cons of all these methods, we choose Hough Transform for our investigation. The main reason for this choice, apart from its good success rate and popularity, was its relative ease of use, simple setup, and open availability of relevant APIs for testing. The choice of Hough Transform as the circle detection method brings another challenge to the front. It is a parameterized method that works on thresholds. The quality and number of detected circles depend largely upon the parameter thresholds, which may vary given changing intensities, illumination of pixels, and other relevant features of the image. Manual settings of these parameters could prove difficult as these settings will have to be adjusted for different scenarios of traffic. To solve this problem, we use DE as the parameter optimizer, and attach it to the circle detection method.

Being a parameter based detection method, Hough Transform requires that the user provides some information about the circles that need to be detected. For example, the edge detector component requires a threshold to be set for the quality of edges detected. The higher this threshold, the fewer the number of circles that are detected. The important parameters for Hough Circle detection are:

- Accumulator threshold
- Edge detection threshold
- Inverse ratio of resolution
- Minimum distance between detected centers
- Minimum radius of detected circles
- Maximum radius of detected circles

DE Optimizer

All the parameters mentioned in the previous section are integers. These parameters can be tuned manually for a given scenario but the same set may show less than satisfactory performance on other test subjects. Thus, there is always a motivation to automate the process, and for that reason we employ DE to perform the parameter search. This, of course will require more compute cycles but would, at the same, improve the accuracy and robustness of the system as a whole. DE is a simple real parameter optimization algorithm. It works through a cycle of stages as represented in Figure 6.

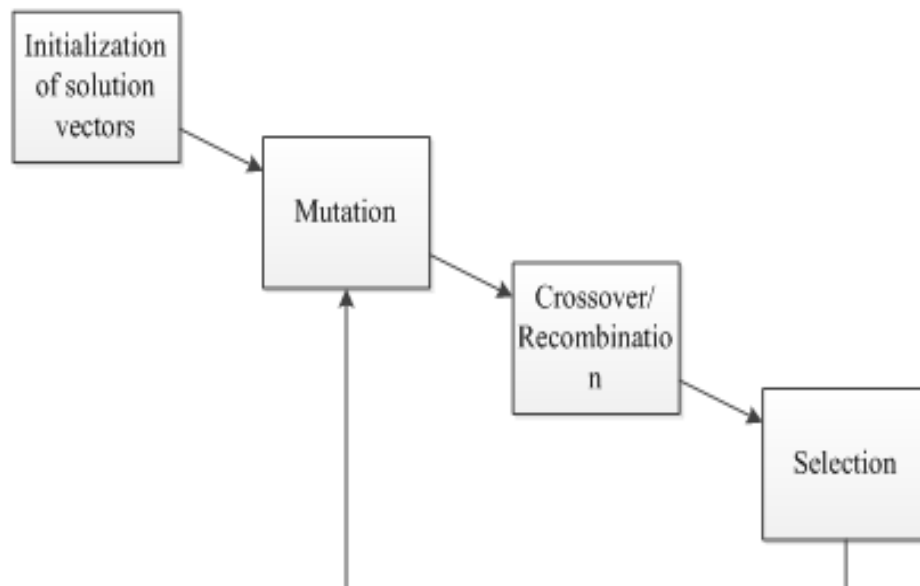


Figure 6: Steps in Differential Evolution

Presented below is the pseudo-code for DE algorithm.

Pseudo-code for classical DE algorithm

1. Read F , Cr and NP , G (no. of generations)
 2. Randomly initialize the population of NP vectors as $P = \{X_{1,G}, X_{2,G} \dots X_{NP,G}\}$ with each vector $X_{i,G} = [X_{1,G}^i, X_{2,G}^i \dots X_{D,G}^i]$, distributed uniformly between its min and max value represented as $X_{min} = [x_{1,min}, x_{2,min} \dots x_{D,min}]$ and $X_{max} = [x_{1,max}, x_{2,max} \dots x_{D,max}]$ where $i \in [1, NP]$.
 3. **While** the stopping condition for the algorithm is not met
Do
 - For** each vector from 1 to NP
 - 3.1 Perform Mutation i.e. generate a donor V_G^i for each target vector X_G^i , as follows

$$V_G^i = X_{r1,G}^i + F \times (X_{r2,G}^i - X_{r3,G}^i)$$
 - 3.2 Perform Crossover i.e. generate a trial vector U_G^i for each target vector X_G^i as follows

$$u_{j,G}^i = \begin{cases} v_{j,G}^i & \text{if } (\text{random}_j^i [0, 1] \leq Cr \text{ or } j = j_{rand}) \\ x_{j,G}^i & \text{Otherwise} \end{cases}$$
 - 3.3 Perform Selection i.e. compare the fitness of trial and target vector and choose the more deserving candidate as follows

$$X_{G+1}^i = \begin{cases} U_G^i & \text{if } f(U_G^i) \leq f(X_G^i) \\ X_G^i & \text{if } f(U_G^i) > f(X_G^i) \end{cases}$$

where $f(X)$ is the objective function.
 - 3.4 Increase the generation count, set generation $G+1$ as G .
- end While**
-

We tested 5 DE variants to gauge their ability to perform this task effectively, and suggest the one that performs best in terms of number of function evaluations used. These variants differ in the way they create new solutions by tweaking the mutation strategies, and can be mathematically distinguished as:

“DE/Rand/1/Bin”:

$$V_G^i = X_{r1,G}^i + F \times (X_{r2,G}^i - X_{r3,G}^i) \quad (6)$$

“DE/Best/1/Bin”:

$$V_G^i = X_{best,G}^i + F \times (X_{r2,G}^i - X_{r3,G}^i) \quad (7)$$

“DE/RandToBest/1/Bin”:

$$V_G^i = X_{1,G}^i + F \times (X_{best,G}^i - X_{2,G}^i) + F \times (X_{r3,G}^i - X_{r4,G}^i) \quad (8)$$

“DE/Best/2/Bin”:

$$V_G^i = X_{best,G}^i + F \times (X_{r1,G}^i - X_{r2,G}^i) + F \times (X_{r3,G}^i - X_{r4,G}^i) \quad (9)$$

“DE/Rand/2/Bin”:

$$V_G^i = X_{1,G}^i + F \times (X_{r2,G}^i - X_{r3,G}^i) + F \times (X_{r4,G}^i - X_{r5,G}^i) \quad (10)$$

The indices r_1, r_2, r_3, r_4 and r_5 are mutually exclusive, randomly chosen and are different from the *base/target* index i . $X_{best,G}^i$ is the vector with best fitness in the generation G . F is the scaling factor already described above.

DE, being a real parameter optimizer, has to be modified to work with integer values. This essentially makes the task a combinatorial optimization problem. Truncating the real values to integer values seems a straight forward solution to this problem, but it has shown to be characteristically unstable in some cases (Onwubolu & Davendra, 2009). Many novel approaches have been proposed to make DE perform the combinatorial optimization task and have yielded good results (Onwubolu & Davendra, 2006). We utilize the approach suggested in (Onwubolu & Davendra, 2006) to convert integer values to float values and vice-versa, keeping all other properties of the DE variants unchanged.

In real world applications, in general, apart from the distance between the camera and the road, other calibration parameters are usually known to the designer. This may help in determining a region of interest of the image where the vehicles are most likely to be detected. It would be computationally prudent to perform the detection and analysis on this region instead of the whole frame. As this work is primarily focused on testing the axle detection, and counting approach (examining DE’s effectiveness at the same time), we have steered clear of having to specify the calibration parameters of the camera and the captured scene. Instead, we have used video sequences where the distance between camera and the road is not fixed. This approach, though being relatively computationally expensive, tests the robustness of the system, and DE in particular by expanding its search space.

The fitness function for DE to optimize is kept simple. There is a cost associated with circles, which are detected but are not aligned horizontally within a certain threshold. This addition of cost is based on the assumption that all the axles of the vehicle are likely to be horizontally aligned. The special case of raised axles is not considered here. Another cost is added if the radii of the detected circles differ more than a certain set threshold. This again is based on the assumption that all the axles of a vehicle are more likely to be of the same radius.

There is a minimum distance between the centers that is specified and a cost is added if some circles are found to be closer than that distance. This is done to discourage DE from finding circles that are very close to each other. In mathematical form our model is represented as:

$$f(x) = (C_M)^2 \times (g(x) + h(x) + r(x)) \quad (11)$$

where

$$g(x) = \left(\frac{1}{C_A + \epsilon} + (C_T - C_A) \right) \quad (12)$$

$$h(x) = \left(\frac{1}{C_R + \epsilon} + (C_T - C_R) \right) \quad (13)$$

$$r(x) = \left(\frac{1}{C_D + \epsilon} + (C_T - C_D) \right) \quad (14)$$

and

C_M - maximum number of axles/circles to be detected in a frame; in our case it is set to 10

C_T - total number of circles detected in a frame

C_A - number of horizontally aligned circles detected

C_R - number of detected circles having almost same radius

C_D - number of circles having their centroids satisfactorily distant from each other

ϵ - a very small number to avoid divide-by-zero error

There certainly can be many more sophisticated ways to improve this model but for our purposes we have kept it simple. Another reason for keeping the fitness function quick to compute is to make the classification process time efficient.



Figure 7: Axle detection using DE

Classifying vehicles based on the number of axles, and distance between them does away with the need to compute other attributes of the vehicle like area, solidity, depth, etc. Computing these additional features may improve the classification accuracy but not without increasing the computational cost. Also, the length of a vehicle can be fairly approximated as the distance between the farthest axles.

Our approach also does away with the need for employing a specialized classification algorithm as there are only few features involved. We use a simple Decision Tree classifier. The current decision classes that we have experimented on are shown in Figure 7. As is clear from the figure, employing axle count information is crucial, in fact necessary to correctly distinguish between Truck Type I and II, Truck Type III and IV, Truck Type VI and VII, as they have similar geometric features but different axle counts. It should be noted that for this scheme to be fruitful in a production environment, the distance between the camera and the road needs to be fixed beforehand which should be considered a part of the camera calibration process. We, however, have experimented with varying distances as already mentioned and for the reasons stated in the previous section.








S. No.	Vehicle	No. of Axles	Vehicle Class
1		2	Passenger Vehicle/Truck Type 0
2		3	Truck Type I
3		4	Truck Type II
4		3	Truck Type III
5		4	Truck Type IV
6		5	Truck Type V
7		6	Truck Type I

Figure 8: Vehicle outlines and their associated classes. Axle count information is necessary to distinguish between Truck Type I and II, Truck Type III and IV, and Truck Type V and VI

Tracking

After the vehicles are detected, tracking is performed on them to ascertain their positions in the next frame. In other words, the trajectory of the vehicle during the course of its existence in the video is determined through tracking. This is an important operation if the correct count of the vehicles is to be obtained. If the vehicles are not tracked properly, the redetection of the same vehicle in the next frame may be surmised as a new vehicle, which in turn would result in over-counting.

We perform detection and tracking separately where we first detect vehicles in every frame, and then associate current vehicle data with data from the previous frame, and update the vehicle's location and features iteratively. We take advantage of the fact that any detected vehicle is not likely to be present in only one frame. This method is an example of region based tracking mechanism (Mandellos, Keramitsoglou, & Kiranoudis, 2011), (Lai, Huang, & Tseng, 2010) where the vehicle contour is detected and fitted inside a rectangular box. This box and contour, in our system, is characterized by edges, contour area, and box coordinates. Data association and template matching is then performed to track the vehicles in consecutive frames to track them. A local memory containing the features, and other relevant data is maintained for every vehicle that is detected. The features are updated during tracking as the vehicle moves from frame to frame. This way the vehicle appearance in the memory stays in sync with the current state of the vehicle. This continuous updation is vital since the vehicle appearance is prone to edge, and area deformation while in motion.

EXPERIMENTATION AND RESULTS

In this section, we discuss the performance of DEVEC tested on multiple real time, primarily urban two lane traffic scenarios. Apart from evaluating the system for its speed, correct vehicle count, and classification accuracy, the major thrust of the evaluation is on the performance of Differential Evolution (DE) as an optimizer applied to this scenario. Moreover, given the current setting and to gauge the performance of DE, the system was tested with and without the DE optimizer.

As discussed in the previous section, the main utilization of DE is to find and locate the number of correct axles of the vehicle by minimizing a fitness function. The information on the number of axles, distance between the farthest axles supplemented with height, width, and contour area of the vehicle was used to classify the vehicle in one of the listed classes. The system was initially tested with manual settings of hough transform parameters and then with five popular variants of DE, on vehicles isolated from multiple video sequences.

The videos were recorded on a two lane road, and the traffic flow was chosen to be moderate. Table 1 presents the first set of results comparing the system's classification performance with and without the DE optimizer. It lists the performance on 20 different vehicles isolated from a single video sequence with the number of axles of each vehicle. A single fine-tuned manual setting of hough parameters was used all along the tested videos. The number 1 denotes the correct identification of number of axles by the system using the manual settings, and a 0 means that the system failed to identify the correct number of axles with the given parameters leading to a misclassification.

It is important to mention that after a vehicle is detected, the rectangular region around it is extracted, and may be examined either by a pre-determined circle detector or the DE optimizer module. This extracted vehicle region from the frame may measure as less than 40×30 pixels. This presents a problem when it comes to detecting circles in such a confined image space. To maintain the practicality of the context and achieve good circle detection, hough parameter thresholds are highly relaxed. This has the effect of detecting many more circles and in these cases many false positives are generated. It is this challenge that DE has to overcome with correct circle detection, and remove unwanted circles by minimizing the fitness function.

As for DE, it was allowed to search the space for optimal hough parameters thereby minimizing the energy/fitness function. For all the five variants, the fitness function value obtained is listed in the corresponding columns. If the DE variant was able to identify the correct number of axles, it is signified with a 1 alongside the fitness value, and 0 otherwise. As for the control parameter settings of DE, we fixed the crossover rate (Cr) to 0.9 and scaling factor (F) to 0.5, and population size (NP) to 50 as suggested in (Qin & L, 2013). The maximum number of function evaluations was set to 200.

Our results show that the DE/Rand/1/Bin strategy emerges as the best DE strategy among the ones tested, reaching an accuracy of approximately 90% in all video sequences tested. All the five different variants tested in this paper have their own distinct characteristics in the way they affect the generation of new solutions and impact the overall search process. For example, DE/Best/1/Bin tries to search around the best solution achieved so far thereby moving towards the solution very quickly and in many cases converging prematurely. This strategy loses diversity of the population quite fast as compared to the other variants. DE/Rand/2/Bin on the other hand is known to create diverse solutions due to the presence of two differential vectors. This diversity leads to very slow convergence. All in all, no single strategy is perfect for all problems, and their success is partially dependent on the nature of problem. In this case, DE/Rand/1/Bin proves to be a better strategy, primarily due to its balanced and simple approach towards the search process. It is also relatively slow but manages to maintain good diversity of solutions throughout the search process.

Apart from the mutation strategy and as is true with any evolutionary algorithm, the control parameter settings play an important part in the performance of DE. Conducting a large scale control parameter analysis for this scenario is out of the scope of this work. We nevertheless experimented with multiple population sizes to see if that actually impacts the system's performance. The motivation is to investigate if a lower value of the population size NP, and for that matter fewer function evaluations, produces the same results as shown in Table 1, or would a higher NP produce better results. NP cannot be too high so as to exacerbate the performance making the system untenable. At the same time, it cannot be too low as this might seriously degrade the accuracy. In essence, this problem presents the classical accuracy versus speed dilemma and we try to find the critical and harmonious set of parameters that lead to acceptable performance on this particular problem. The results are enumerated in Table 2.

Table 1: A comparison of five variants of DE in detecting the number axles and their centers in 20 frames isolated from multiple video sequences. The values presented indicate the best/minimum value obtained by the variant along with a binary number (successful detection is represented as 1 and 0 otherwise).

Vehicle No.	No. of Axles	Manual Setting	DE/Best/1/Bin	DE/Rand/1/Bin	DE/RandToBest/1/Bin	DE/Best/2/Bin	DE/Rand/2/Bin
1	2	0	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
2	2	1	52.00(1)	52.00(1)	36.33(0)	52.00(1)	52.00(1)
3	2	1	52.00(1)	52.00(1)	52.00(1)	52.00(1)	36.33(0)
4	2	1	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
5	5	0	36.33(0)	25.00(1)	25.00(1)	36.33(0)	29.00(0)
6	2	1	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
7	2	0	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
8	4	0	36.33(0)	36.33(0)	29.00(1)	29.00(1)	29.00(1)
9	2	1	52.00(1)	52.00(1)	36.33(0)	52.00(1)	36.33(0)
10	2	1	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
11	5	0	36.33(0)	25.00(1)	25.00(1)	25.00(1)	29.00(0)
12	2	0	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
13	2	1	154.00(0)	203.00(0)	52.00(1)	154.00(0)	154.00(0)
14	2	1	52.00(1)	52.00(1)	52.00(1)	52.00(1)	52.00(1)
15	4	0	36.33(0)	29.00(1)	36.33(0)	29.00(1)	29.00(1)
16	5	1	36.33(0)	25.00(1)	25.00(1)	36.33(0)	29.00(1)
17	4	0	36.33(0)	29.00(1)	36.33(0)	29.00(1)	29.00(1)
18	2	1	52.00(1)	52.00(1)	36.33(0)	36.33(0)	52.00(1)
19	2	1	52.00(1)	52.00(1)	36.33(0)	52.00(1)	52.00(1)
20	2	0	52.00(1)	52.00(1)	36.33(0)	36.33(0)	52.00(1)
-	Wins	10	13	18	13	15	14
	Suc. Rate (%)	50	65	90	65	75	70

Table 2: Effect of increasing NP and function evaluations on success rate. Saturation point is reported at 40-60 combination

Population size (NP)	Success Rate (%)	Saturation FE's
10	70	60
20	75	50
30	85	50
40	90	60
50	90	70
60	90	70
70	90	85
80	90	90

We performed the optimized parameter search with population sizes ranging between 10 and 80 with increments of 10. It is shown that the success rate of a population size improves with an increase in function evaluations. But after a certain point, increasing the population size does not improve the success rate. On similar lines, an increase in function evaluations does not offer an added advantage after a certain limit since the success rate saturates. We found that the best set of control parameters that lead to the highest accuracy (90%), among the combinations compared, is: $F=0.5$, $Cr=0.9$, $NP=40$ with 60 FEs. Increasing NP above this value does not yield better results. If this critical point can be deduced theoretically, it may be used as an effective indicator of the extra computational budget that the DE optimizer module may consume in the system.

Figure 9, for the lack of space, visually depicts a part of the results obtained for manual settings (left aligned in the sub-figures) as compared to DE/Rand/1/bin optimized set (right aligned in the sub-figures) discovered, for 7 different vehicles sourced from multiple video sequences. Axles detected by both methods are represented by solid white circles.








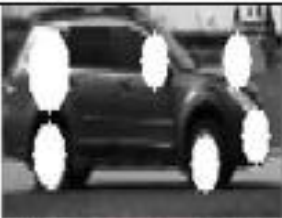





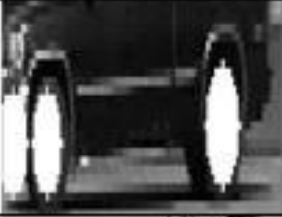
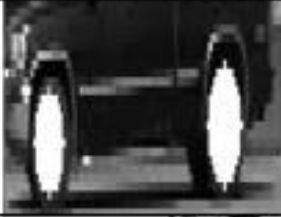






Frame No.	Original Capture	Detected Axles-Manual Setting	Detected Axles - DE Optimization
1			
2			
3			
4			
5			
6			
7			

Figure 9: Results obtained through manual settings (third column) of Hough Transform parameters vs the best settings obtained for DE/Rand/1/bin (fourth column)

Table 3 and 4 summarize the cumulative results of the DEVEC as a confusion matrix. The system was tested on three videos of approximate duration of 10 minutes each. This by no means is an exhaustive test for our system but as a proof of concept the sample size is assumed to be good enough. In total, the video sequences contained about 302 vehicles to be classified into seven classes. It must be recorded that, since the feature set is small (length, width, contour area, axles), the classification performance is less a test of the decision tree classifier, and more of a test of correct feature detection.

DEVEC shows excellent performance while detecting the passenger vehicles achieving a precision of 0.90 for this class of vehicles. The precision for truck types I, II, III, and IV is also decent with all classes having a value above 0.80. The system however shows a less than satisfactory performance on large trucks having five or more axles. One reason as to why DEVEC does not accurately distinguishes between these two classes is the immense edge discrepancies on and along the close axles of these trucks, which prohibits them from being detected. As a consequence, being of the same height, and width, truck type VI is classified as truck type V. Another reason of low precision and recall might be the limited number of truck type V, and VI samples present in the corresponding videos.

Table 3: Confusion matrix for cumulative vehicle count across three video sequences. PV stands for passenger vehicle.

		Predicted Class						
		PV / Truck Type 0	Truck Type I	Truck Type II	Truck Type III	Truck Type IV	Truck Type V	Truck Type VI
Actual Class	PV / Truck Type 0	134	5	0	0	0	0	0
	Truck Type I	14	60	3	0	0	0	0
	Truck Type II	0	8	31	2	0	0	0
	Truck Type III	0	0	4	23	2	0	0
	Truck Type IV	0	0	0	5	18	0	0
	Truck Type V	0	0	0	0	2	9	2
	Truck Type VI	0	0	0	0	0	3	6

Table 4: Confusion matrix for individual classes

	Precision	Recall
Passenger Vehicle	0.90	0.96
Truck Type I	0.82	0.77
Truck Type II	0.81	0.75
Truck Type III	0.82	0.79
Truck Type IV	0.81	0.78
Truck Type V	0.75	0.69
Truck Type VI	0.75	0.66

On a manual inspection of the test video sequences, a total of 326 vehicles were counted. DEVEC was able to detect 302 vehicles achieving a decent detection accuracy of 92%. One reason for misdetection of vehicles was the almost similar color configuration of the background and the passing misdetections, though this was a rare observation. In some circumstances, vehicles that moved very slowly, became a part of the background itself thereby generating no motion pixels through the background subtraction method. One method of improving the detection further would be to use more scene cues for the background construction than just the color information.

CONCLUSION

This work presents DEVEC, an axle count based vehicle detector and classifier capable of detecting and classifying vehicles on a two lane urban environment. We use an adaptive threshold updation technique to compare two subsequent frames, and then employ the background subtraction method to extract motion pixels from a video sequence. Axle information is utilized to perform classification of vehicles. Axle detection is performed with Hough Transform, a parameterized feature detection method. The working parameters set of Hough Transform is dependent on the image data, and type of problem being addressed. With changing weather conditions and scene outlook, manually setting these parameters is tedious, and as shown in this work often produces less than satisfactory results. To circumvent this problem, we therefore use a combinatorial version of Differential Evolution to optimize the parameter set yielding much higher accuracy as shown by the results we achieved. Five different variants of DE were tested initially, and it was observed that DE/Rand/1/bin is most suitable for this task reaching a steady success rate of 90% while excluding the false positives. To further improve the speed and accuracy of the system, we further investigated DE/Rand/1/bin. For this

we tested this variant with multiple population sizes (NP) – FEs combinations. We found that $F=0.5$, $Cr=0.9$, and $NP=40$ with 60 FEs yields an accuracy of around 90%, and increasing NP further does not yield any better results. Due to speed considerations, as of now, our current system is suited to be used as an offline vehicle classifier. To make the system perform as an online classifier, we will in the future work, consider making some changes. For example, by careful camera calibration, it is possible to specify a region of interest in the test frame where the probability of finding the axles is quite high given various assumptions about inclination of the road. This will reduce the computing load considerably. If there is enough information available about the scene, it is possible to initialize DE with good values to begin with. These and other modifications are planned as future work. Moreover, background construction that uses most of the compute cycle need to be refined further. For example, a Gaussian mixture model type pixel classification could be used instead of the per-pixel comparison to speed up the classification process.

In addition, the novel DEVEC presents some advantages as well as challenges. For example, it does away the need for a shadow removal module, as the camera is mounted sideways. This camera view also restricts the field of view. Occlusion has been handled successfully in moderate traffic. The speed of the system remains an issue as the frame processing rate currently achieved stands at 13 frames per second on average. This is not yet suitable for online classification. Future work includes developing an improved and speedy background construction algorithm to improve the frame rate. A parallel version of DE may also be investigated to identify the axles quickly.

REFERENCES

- 3M Canoga Application Note (2003). Canoga Vehicle Detection System. *Accurate Vehicle Detection at Intersections Using 3M Canoga M702 Non-Invasive Microloops TM-2003-11*.
- Al-Smadi, M., Abdulrahim, K., & Salam, R. (2016). Traffic Surveillance: A Review of Vision Based Vehicle Detection, Recognition and Tracking, *International Journal of Applied Engineering Research*, 11(1), 713-726.
- Agarwal, S., Awan, A., & Roth, D. (2004) “Learning to detect objects in images via a sparse, part-based representation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11), 1475-1490.
- Bongiovanni, G., Crescenzi, P., & Guerra, C. (1995). Parallel Simulated Annealing for Shape Detection. *Computer Vision and Image Understanding*, 61(1), 60-69.
- Buch, N., Velastin, S. A., & Orwell J. (2011). A Review of Computer Vision Techniques for the Analysis of Urban Traffic. *IEEE Transactions on Intelligent Transportation Systems* 12(3), 920-939.

- Chen, X. & Zhang, C. (1997). Vehicle classification from traffic surveillance videos at a finer granularity. *Advances in Multimedia Modeling*, 772-781. Springer Berlin Heidelberg.
- Chen, Z., Ellis, T., & Velastin, S. (2012). Vehicle detection, tracking and classification in urban traffic. In *15th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 951-956.
- Costa L. F. D., & Cesar Jr., R. M. (2000). Shape Analysis and Classification. *CRC Press, Inc.* Boca Raton FL, U.S.A,
- Cucchiara, R., Grana, C., Piccardi, M., & Prati, A. (2003). Detecting moving objects, ghosts, and shadows in video streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10), 1337-1342.
- Das, S., & Suganthan, P. N. (2011). Differential evolution - A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1), 4-31.
- Elgammal, A., Harwood, D., Davis, L. S. (2000). Non-parametric Model for Background Subtraction. In *6th European Conference on Computer Vision*. Dublin, Ireland.
- Engelbrecht, A. P. (2014). Fitness Function Evaluations: A Fair Stopping Condition? *IEEE Symposium Series on Computational Intelligence*, 1-8.
- Greggio, N., Bernardino, A., Laschi, C., Dario, P., & Santos-Victor, J. (2010). Self-adaptive Gaussian mixture models for real-time video segmentation and background subtraction. In *Proc. IEEE 10th International Conference on Intelligent Systems Design and Applications (ISDA)*, 983-989.
- Gupte, S., Masoud, O., Martin, R. F. K. & Papanikolopoulos, N. P. (2002). Detection and classification of vehicles. *IEEE Transactions on Intelligent Transportation Systems* 3(1), 37-47.
- Haque, M., Murshed, M. M., & Paul, M. (2008). Improved Gaussian mixtures for robust object detection by adaptive multi-background generation. *IEEE 19th International Conference on Pattern Recognition (ICPR)*, 1-4.
- Iivarinen, J., Peura, M., Srel, J., & Visa, A. (1997). Comparison of combined shape descriptors for irregular objects. In *8th Proc. British Machine Vision Conf., Cochester, UK*, 430-439.
- Indu, S., Gupta, M., & Bhattacharyya, A. (2011). Vehicle tracking and speed estimation using optical flow method. *Int. J. Engineering Science and Technology*, 3(1) 429-434.
- Jones, G. A., Princen, J., Illingworth, J., & Kittler, J. (1990). Robust estimation of shape parameters. In *Proc. British Machine Vision Conf.*, 43-48.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1), 35-45

Kanhere, N. K., & Birchfield, S. T. (2008). Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 148-160, 2008.

Karmann, K. P., Brandt, A.V., & Gerl, R. (1990). Moving object segmentation based on adaptive reference images. In *Proc. of 5th European Signal Processing Conference*, 2, 951-954.

Koller, D., Weber, J., & Malik, J. (1994). Towards realtime visual based tracking in cluttered traffic scenes. In *Proc. of the IEEE Intelligent Vehicles*, 201-206.

Kumar, P., Ranganath, S., & Weimin, H. (2003). Bayesian network based computer vision algorithm for traffic monitoring using video. In *Proc. Int. IEEE Conference on Intelligent Transportation Systems*, 1, 897-902.

Lai, J. C., Huang, S. S., & Tseng, C. C. (2010). Image-based vehicle tracking and classification on the highway. In *International Conference on Green Circuits and Systems (ICGCS)*, 666-670.

Lai, J. C., Huang, S.S., & Tseng, C.C. (2010). Image-based vehicle tracking and classification on the highway. In *International Conference on Green Circuits and Systems (ICGCS)*, 666-670.

Levinson, J., Askeland, J., Becker, J., Dolson, J., Held, D., Kammel, S., Kolter, J., Langer, D., Pink, O., Pratt, V., Sokolsky, M., Stanek, G., Stavens, D., Teichman, A., Werling, M., & Thrun, S. (2011). Towards fully autonomous driving: Systems and algorithms. In *Proc. IEEE Intelligent Vehicle Symposium*, 163-168.

Li, Q. L., & He, J.F. (2011). Vehicles detection based on three-frame-difference method and cross-entropy threshold method. *Computer Engineering*, 37(4), 172-174.

Linda, S., & Volling, M. T., (2003). Paradigm is shifting in vehicle detection. *IMSA journal featured article*. Retrieved May 6th, 2016, from <http://www.imsasafety.org/journal/jf03/janfeb3.htm>

Lou, J., Tan, T., Hu, W., Yang, H., & Maybank, S. J. (2005). 3-D model-based vehicle tracking. *IEEE Transactions on Image Processing*, 14(10), 1561-1569.

Mandellos, N.A., Keramitsoglou, I., & Kiranoudis, C. T. (2011). A background subtraction algorithm for detecting and tracking vehicles. *Expert Systems with Applications*, 38(3), 1619-1631.

Ma, X., & Grimson, W. E. L. (2005). Edge-based rich representation for vehicle classification," in *Proc. IEEE 10th International Conference on Computer Vision*, 2, 1185-1192.

- Mao, X., Inoue, D., Kato, S., & Kagami, M. (2012). Amplitude-modulated laser radar for range and speed measurement in car applications. *IEEE Transactions on Intelligent Transportation Systems* 13(1), 408-413.
- McFarlane, N.J., & Schofield, C.P. (1995). Segmentation and tracking of piglets in images. *Machine vision and applications*, 8(3), 187-193.
- Middleton, D., Chara, H., & Longmire, R. (2009). Alternative vehicle detection technologies for traffic signal systems. *Technical Report*. Retrieved May 5th, 2016, from <http://d2dtl5nnlpfr0r.cloudfront.net/tti.tamu.edu/documents/0-5845-1.pdf>
- Morris, B. T. & Trivedi, M. M. (2008). Learning, modeling, and classification of vehicle track patterns from live video. *IEEE Transactions on Intelligent Transportation Systems*, 9(3) 425-437.
- Negri, P., Clady, X., Milgram, M., & Poulencard, R. (2006). An oriented-contour point based voting algorithm for vehicle type classification. In *18th International Conference on Pattern Recognition*, 1, 574-577.
- Nguyen, P.V., & Le, H.B. (2008). A multi-modal particle filter based motorcycle tracking system. In *PRICAI 2008: Trends in Artificial Intelligence*, 819-828. Springer Berlin Heidelberg.
- Onwubolu, G. & Davendra, D. (2006). Scheduling flow shops using differential evolution algorithm. *European Journal of Operations Research*, 171(2), 674-692.
- Onwubolu, G. & Davendra, D. (2009). *Differential Evolution: A Handbook for Global Permutation-Based Combinatorial Optimization*. Springer-Verlag, Heidelberg.
- Ottlik, A., & Nagel, H. H. (2008). Initialization of model-based vehicle tracking in video sequences of inner-city intersections. *International Journal of Computer Vision*, 80(2) 211-225.
- Park, K., Lee, D., & Park, Y. (2007). Video-based detection of street parking violation. In *Proc. International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, 152-156.
- Qin, A. K., & Li, X. (2013). Differential Evolution on the CEC-2013 Single-Objective Continuous Optimization Testbed. *IEEE Congress on Evolutionary Computation, Cancun, Mexico*, 1099-1106.
- Roth, G., & Levine, M. D. (1994). Geometric primitive extraction using a genetic algorithm. *IEEE Trans. Pattern Anal. Machine Intell*, 16(9), 901-905.
- Sen-Ching, S. C., & Kamath, C. (2004). Robust techniques for background subtraction in urban traffic video. *International Society for Optics and Photonics in Electronic Imaging*, 881-892.

- Sharma, A., M. Harding, B. Giles, D. Bullock, J. Sturdevant, & S. Peeta (2008). Performance Requirements and Evaluation Procedures for Advance Wide Area Detectors. *Submitted to the Transportation Research Board for publication and presentation at the 87th Annual Meeting, Washington, D.C.*
- Sivaraman, S., & Trivedi, M. M. (2013). Looking at vehicles on the road: A survey of vision based detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4), 1773-1795.
- Stauffer, C. & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proceedings of IEEE Conference on Comput. Vis. Pattern Recog.*, 2, 252.
- Storn, R. M. & Price, K. V. (1995). Differential evolution - A simple and efficient adaptive scheme for global optimization over continuous spaces. *International Computer Science Institute, Berkeley, CA, USA, ICSI Technical Report 95-012.*
- Storn, R. M. & Price, K. V. (1996). Minimizing the real functions of the ICEC 1996 contest by differential evolution. In *Proc. IEEE International Conference on Evolutionary Computation*, 842-844.
- Tian, B., Morris, B. T., Tang, M., Liu, Y., Yao, Y., Gou, C., Shen D., & Tang, S. (2015). Hierarchical and Networked Vehicle Surveillance in ITS: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, 16(2), 557-580.
- Wang, X. (2013). Intelligent multi-camera video surveillance: A review. *Pattern Recognition Letters*, 34, 3-19.
- Wang, Z., & Hong, K. (2012). A new method for robust object tracking system based on scale invariant feature transform and camshaft. In *Proc. 2012 ACM Research in Applied Computation Symposium*, 132-136.
- Wren, C. R., Azarbayejani, A., Darrell, T., & Pentland, A. P. (1997). Pfunder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), 780-785.
- Yuen, H. K., Princen, J., Illingworth, J., & Kittler, J. (1989). Comparative study of Hough transform methods for circle finding. *Image Vision Comput.*, 8(1), 71-77.
- Zafar, I., Edirisinghe, E. A., Acar, S., & Bez, H. E. (2007). Two-dimensional statistical linear discriminant analysis for real-time robust vehicle-type recognition. In *International Society for Optics and Photonics Electronic Imaging*, 6496.

Zhang, J., Wang, F. Y., Wang, K., Lin, W. H., Xu, X., & Chen, C. (2011). Data-driven intelligent transportation systems: a survey. *IEEE Transactions on Intelligent Transportation Systems*, 12(4), 1624-1639.

Zhao, S.L., & Lee, H. J. (2009). A spatial-extended background model for moving blobs extraction in indoor environments. *Journal of Information Science and Engineering*, 25(6), 1819-1837.

Zheng, J., Wang, Y., Nihan, N. L. & Hallenbeck, M. E. (2006). Extracting roadway background image: Mode-based approach. *J. Transp. Res. Board*, 1944(1), 82-88.

Zivkovic, Z. & Heijden, F. (2006). Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern recognition letters*, 27 (7), 773-780.