Vehicle Identification Systems using Virtual Line Sensors and Speed Up Robust Features

Samuel Mahatmaputra Tedjojuwono*, Nathalia Devina Widjaja, and Antonius Kurniawan
Bina Nusantara University, Jakarta, Indonesia 11480

*smahatmaputra@binus.edu

Abstract. This work proposes an improved and robust algorithm to a virtual line approach object movement detection using computer vision technique. The improved performance will be emphasized on how one line could detect multiple object, determines their sizes and could at most classify the object into group similarities. The algorithm later will identify the vehicles based on their license plate using speed up robust features - SURF technique. The method incorporating Bresenham’s points plotter algorithm to determine the points coordinate in a line array, which later calculating movement within the given points and group neighboring points with pixel values indicating movement that are in close proximity as one object. This could then calculate the number of points laid within the group which then could determine the size of the object. As a result of the proposed work, a robust object classification technique that is implemented as a computer application that can be used in a live environment context.

Keywords: vehicle classification, computer vision, virtual sensors

1. Introduction

Vehicles classification has been an important issue in road management systems. Its implementation could vary from surveillance system, traffic control, crowd monitoring system, and commercial/marketing purposes. In surveillance system, traffic/ crowd monitoring system the needs to know how many people or vehicles exist in one place at a given time would assist in traffic control system and law enforcement. Current widely used implementation is by using weight in motion sensors that could be invisible and costly to be installed on every road [1]. One of the main aims of this classification is to identify users of the roads to further implement structure and regulation to increase the road lifespan. In the commercial context, the need has been extended to the area of knowing how many people visiting certain isle of products in a supermarket for example, which could improve the supply chain decision-making process. With these more extensive needs the use of computer vision or video analytics techniques such as virtual sensors turn out to be more promising and efficient in term of cost. The proposed work would aim to improve an original virtual line sensor previously implemented on a video analytics computer application [2]. The earlier work implemented could detect one movement at a time although there are more than one movement occur at the same time within the line array. It also does not inform the size of the object that passed through the line, thus in the context of heterogeneous objects the previous system could not classify the objects. The significance of the proposed research is to produce a new algorithm of robust object classification using Bresenham’s [2] integral point plotter, which could identify several objects passing through one virtual line at the same time. It is as well could...
classify the object size thus determine whether the object – in the context of live traffic system – are buses, cars, and motorbikes and later on identify them using license plate matching algorithm.

A recent work on robust vehicle classification used sparse representation and Bayesian state inference framework to trace the object [3]. However, the approach does not set specific class to where a detected vehicle should be grouped. Another approach used histogram of oriented gradients (HOG) features [4] to classify vehicles into two classes: trucks and cars. The method comprises of two steps: multi directional vehicle detection and the classification stage. The results vary from 95.1% until 98.2%, however implemented using one hidden layer neural network. Another work was achieved through the implementation of Gabor wavelet transform combined with feature dimension reduction using PCA [5], which could reach the average of 90% correct rate. Although the work was tested only on passenger cars to identify their manufacturers the method is promising both in speed and accuracy, thus could potentially solved other classification problems. Deep learning or also known as CNN (convolutional neural network) is other promising approach that could do pixel level mapping. Related work [6] implemented several types of network such as AlexNet and VGG-16 with promising results to classify car from high resolution satellite image, however low precision on van, truck, and pick up classification. Both methods could produce above 90% precision on car however averagely produce around 60% for the rest of the classes. The work as well did not test against bike or other smaller vehicles, which could introduce certain level of learning challenges. Other work would like to implement computer vision-based object classification even to the extent of classifying human age and gender based on face recognition [7]. The work implemented adaptive features and SVM using RBF kernel with a promising result of 90.8% on both male and female age-based classification. However, this approach will be computationally expensive to be implemented in live traffic road systems.

2. Experimental method

The proposed methodology was divided into several modules some of which adopted from the previous original work [2]. The modules involved are: frame grabber to get the video’s frames [2]; pre-processing (blurring / smoothing) to reduce noise [2]; generating line arrays which will detect line placement from the user, and registering all integral points within the line arrays [2]; newly proposed movement detection by generating background reference, background subtraction to calculate delta movement and to register movement within certain position; and lastly the proposed detecting valid objects through identifying beginning and end of an object movement to later classify the object based on the summary of delta movement an object has.

![Figure 1. The proposed method that comprise of pre-processing, generating line arrays, movement detection, and later on detecting valid object through that will be classified into several classes.](image)

The newly proposed work uses mean filtering technique where the background is derived from certain number n of frame F. Previous work [2] implemented frame F compose of the pixel value that laid on a two-dimensional matrix using x and y dimension at a given time t. The approach had resulting in a smaller granularity level of movement detection, thus to the level of picture element (pixel). This will be effective to count one object per line however could potentially introduce wrong vehicle classifications - by its size - in a context of fractal motion area for one object (figure 2-a).
Figur 2. (a) Comparison between different motion detection granularity level. It describes a fractal motion detection area for an object as a result of detail pixel level detection, (b) The contrary newly proposed method implementing grid motion detection resulting in a solid motion area.

In order to achieve a solid motion area for an object (see figure 2-b), current method proposed an improvement through grid motion detection [8,9] by dividing an Image F into several smaller two-dimension area called grid $\beta$. A grid $\beta$ could comprise of several pixels such as 8x8 or 16x16 pixels (see figure 3).

Figure 3. Grid area motion detection illustrated by checkered patterns. A grid would be a 8x8 or 16x16 pixels two dimensional matrix, which would serve as a more generalized motion detection point in comparison to one pixel.
Using this approach, we could have coarser granularity in detecting movement of an object, thus resulting in a more generalized and solid (less fractal) detection area of an object. The following equation represents a frame:

\[ F(x, y, t) \]  

(1)

Whereas background \( G \) at a given time \( t \) is represented through the following equation:

\[ G(x, y, t) \]  

(2)

This background matrix values \( G \) produced from the mean value of a certain number of \( n \) previous frames, where \( i \) is the index of every frame between 0 to \( n-1 \). The background \( G \) is generated through the following equation:

\[
G(x, y, t) = \frac{1}{n} \sum_{i=0}^{n-1} F(x, y, t-i)
\]

(3)

After acquiring the background then current frame \( F(x, y, t) \) could be compared with \( G(x, y, t) \) through subtraction operation and compared with a certain threshold \( T \). The process is explained through the following equation:

\[
|F(x, y, t) - \frac{1}{n} \sum_{i=0}^{n-1} F(x, y, t-i)| > T
\]

(4)

Movement detection on every registered point within a line array acquired by taking into account all pixel value with delta values for each point derived from equation (4) above that is higher than \( T \). Referring to the previous work [2], from this point onwards all detection within the a line array will be accounted as an object movement – although in fact there is possibility that a virtual line sensors could be crossed by two or more vehicles/ objects. This in return will give a wrong result of only one object instead of precisely detecting how many objects crossing the line array. The previous method as well has not classified the vehicle crossing the line through several common classes such as cars and bikes. Current method proposing object classification by detecting how many valid objects going through a line \( L \), measure them by their size and classify them to several common classes. The following sub topic will describe the process of detecting valid object crossing virtual line sensors.

To detect a valid object the algorithm will mark the beginning \( t_{start} \) and end \( t_{stop} \) of objects. This is done through implementing variable \( startobj \) that will be set to true if there exist one point's delta value within the registered member within line \( L \) that is higher than a certain given threshold \( T \). On the other hand, if there are no delta value of any points within the registered points member in the line \( L \) then the \( startobj \) will set to false. Within \( t_{start} \) and \( t_{stop} \) of object movement detected object classification will be carried out. Object classification process is done through normalizing all neighboring points that have no movement values although it lies within a moving detected object, or so-called outliers \( \Phi \). These outliers \( o \) will be set to true by using a two-dimensional operator consist of three positions \( P_{i,j}, P_{i}, \) and \( P_{i+1,j} \). The operator will traverse all points \( P \) within line \( L \) and comparing current \( P_{i} \) delta value to its neighboring points \( P' \) while normalizing all the outliers \( \Phi \). After normalizing all the outliers then the object classification process can be carried out. A class structure of Object is needed to accommodate size field, and two behaviors to access the data as a standard setter and getter methods.

Classification process was done through marking the beginning of an object \( o_{start} h=true \) and the end of an object \( o_{end} h=false \). The main difference of \( o_{start} \) - \( o_{end} \) pair and \( t_{start} - t_{stop} \) previously mentioned is the last pair are used to mark the start and the end of movement detected, which could involve more than one object segments. To the contrary, \( o_{start} \) and \( o_{end} \) are used to mark the start and
the end of every object segment detected within a lifespan of a $t_{start}$-$t_{stop}$ pair. The $o_{end}$ will be set $h=false$ if there exist condition where delta value of $P_i$ lower than the threshold however previous value of $o_{end}$ have been set to true. An object $O$ will have a construct of size $O.size$ data encapsulated with its behaviors to get and set its field's value. The field $O.size$ value will be obtained through summarizing all the delta $D$ value within a pair of $o_{start}$ and $o_{end}$.

By using this $O.size$ an object $O$ could be classified according to its class. The work implementing two classes indicated by threshold value of car class and bike / bicycle class. All objects $O$ that has $O.size$ larger than a given threshold $T_{car}$ will be put into a list structure of car; everything else will be put to the list of bike class.

$$O.size = \sum_{\Delta = o_{start}}^{o_{end}} \Delta$$

Figure 4. Sample values of one point from $t_{start}$ and $t_{stop}$ of the detection comprise of 4 $moveCount$ after twelve frames. It shows pixel values at the same integral point coordinate after 12 frames of images, with 4 detected movement highlighted.

The number of frame will be counted for all significant delta movement detected at each point within a line $L$, summarized and store as $O.size$ in an object class. This process will be started at the time of $t_{start}$, and summarizing the number of movements at $t_{stop}$. All of these summaries will be stored in a list $moveCount$. Figure 4 above shows sample values of one point from $t_{start}$ and $t_{stop}$ of the detection comprise of 4 $moveCount$ after twelve frames. Taking into account several neighboring points that have significant $moveCount$ will gives us indication of one object crossing the virtual line sensor. Figure 5 describing the neighboring movement detection of two distinct objects separated by a gap. All of the $movecount$ values stored in a $movecount$ list. The list maintains all of the aggregated movecount of the points member $P$ of the line $L$ between the beginning and end of motion detection event. The $moveCount$ list was created at the same length as the virtual line in number of integral points.

All summarized count will be stored in the list based on the line dimension and coordinate of the points within line array. Thus, a continuous sequence of movement detection point is an indicator of an object and the number of movements will indicate the size of the object crossing the virtual line. Although two or more objects crossing the line at roughly the same time, the proposed algorithm could differentiate those vehicles by recognizing the gap between the continuous sequences of delta movement within the $moveCount$ list. The result of this process will be object $O$ that will be classified to some of...
common classes `listObjectCar` and `listObjectBike`. Another universal class as well was established to cross check the classification result.

```csharp
public void classifyObjects()
{
    int[] arrNORMALIZEDCount = new int[arrayMoveCount.Length];
    // Normalizing the movement counting array:
    fill all the 0 between detected values:
    for (int i = 1; i < arrayMoveCount.Length - 1; i++) //for each +1 from begin and -1 from the end of the array count movement
    {
        if (arrayMoveCount[i-1] > TMOVEMENT ||
            arrayMoveCount[i] > TMOVEMENT ||
            arrayMoveCount[i+1] > TMOVEMENT) // fill all the 0 between detected values
        {
            arrNORMALIZEDCount[i] = 1;
        }
    }

    // calculating how many cars within the normalized counted movement (1 car = continuous 1 values for several points:
    Boolean h = false;
    int beginObj_idx = 0;
    int endObj_idx = 0;
    for (int i = 0; i < arrayMoveCount.Length; i++)
    {
        int c = arrNORMALIZEDCount[i];
        if (c == 1 && h == false)
        {
            h = true; // Beginning of new object
            beginObj_idx = i;
        }
        if (c == 0 && h == true)
        {
            h = false;
            endObj_idx = i;
            MyObject o = new MyObject(endObj_idx ...
            beginObj_idx); // create new objects
            //Adding universal objects, it means all objects
            if (o.getSize() > TSIZE)
            {
                listObject.Add(o); // end of object
            }
            if (o.getSize() > TSIZECAR)
            {
                listObjectCar.Add(o);
            }
            else
            {
                listObjectBike.Add(o);
            }
        }
    }

    // Red highlighted pixel values with a detected movement (delta pixel to background)
}
```

**Figure 5.** The neighbouring movement detection of two distinct objects separated by a gap. *Pixel* values in hexadecimal collected from 12 frames of video shows 2 object movements.

**Figure 6.** The classify objects functions that will first normalize the outliers $\phi$, detect the beginning and the end using $h$ of an object $O$, counting the delta movement of the line array `normalizedCount`, register the size of $o.size$ and finish by classification process to class list. This is to achieve valid object classification.
SURF calculates the Hessian Matrix Determinant for localizing the key points to decide whether a point is an extreme value given that $x = (x, y)$ for figure I, this is then will be used to match the vehicles license plate. The Hessian Matrix is defined as follows:

$$H(x, \sigma) = \begin{pmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{yx}(x, \sigma) & L_{yy}(x, \sigma) \end{pmatrix}$$

The method for evaluating license plate matching is described through the following steps:

1. Detect features and descriptors by detecting the location of the key points of the matrix image from the Image Object and Moving Image using the Detector Feature library which will assign key points in the form of key point matrices.
2. Descriptors of image objects and image samples are then matched using the Matching Descriptors from the FLANN library.
3. A good matching method will consist of the main points with the closest similarity distance to each other, using the nearest neighbor distance ratio. And determine whether the object is found; the determinant of whether the actual object can be found is when the list of good matches is greater than 100.

3. Result and discussion

The work was tested in two different conditions daylight and night for non-highway road to capture both cars and bikes. In contrast with previous study [2] the newly proposed implementation of virtual line sensor will only need one-line array to count multiple objects crossing the line at the same time and classifying them based on certain threshold size of car and bike. The modified version of grid motion detection implementation has resulting in a more solid (less fractal) motion detection area for one object or vehicle. This is caused by the generalization of several pixel points within one grid [8] (i.e. 8x8, or 16x16 grid) to inform motion for any movement detection in the point members.

![Figure 7](image7.png)

**Figure 7.** Implementation of one virtual line sensor that could identifies more than one vehicle crossing the line at the same time, and classifies them into two classes: car and bike.

The grid motion implemented to support measuring object by its object detection area size and to accommodate the requirement of ability to count multiple objects concurrently crossing the line. The following illustration compares the previous work of virtual line sensors [2,10] that used multiple lines to cover different parts of the road. The reason to have many lines covering multiple areas is the line could only detect one moving object each time. Thus, to cover more are then many lines to the size of one vehicle’s wide would be needed covering entire road width. By stretching the line too long will introduce possibility of information lost, as two vehicles could possibly cross the stretched line at the same time but only one will be counted.
To the contrary the newly proposed method (figure 8-b) needs only one virtual line sensors to detect multiple objects at once. This is where the grid motion detection [8] will be really needed. The technique will minimize fractal effect caused by a detail motion area [2] to the level of pixel point. Although later on during object validation every point in a line will be accounted towards one object-detection, however the noise gaps between points could already be minimized through the process of grid generalization. The proposed technique will group all close motion detected points (neighboring points) together as one object. A separated motion detection points by a gap to a certain number of pixels in length, will be indicated as different object. Therefore, minimizing the unnecessary gap or noise was implemented through grid motion area technique, before later on classifying and counting the number of objects based on their classes. The result was satisfactory which could reach between 82% to 97% of classification rate.

Figure 8. Image (a) shows a multiple virtual line sensor – four lines implemented in the previous work [2], (b) shows current proposed single virtual line to detect and count multiple vehicles crossing the line at the same time.

Figure 9. Tested against low light condition at night to calculate how objects correctly classified as car and bike. Image (a) shows difficult case caused by noise of headlight of cars. Implemented towards low light environment, image (b) shows a better sample taken from the back of vehicles.

The proposed method as well tested during low light condition at night. The position of the camera is purposely placed facing the back of the vehicles. This is to avoid head light reflection on the road surface that could be misunderstood as another object. The challenge was steeply increased at night condition. Although the generalization technique has been implemented, parts of the vehicles that are dark were wrongly misunderstood as the background (dark road surface), therefore not included as motion detected point. When this happen to a larger vehicle such as a car, the gap within the object
would create wrong classification and counting process of two distinct smaller vehicles. Some of which would be classified as two motorbikes. Similar result as well showed within bike classification at night. A bike with gaps detected –usually between the bike and its taillight- would suffer from wrong counting as two different bikes.

Table 1. Vehicles classification result on daylight.

| Class | TRUE | FALSE | Percentage | Total |
|-------|------|-------|------------|-------|
| Car   | 28   | 1     | 97%        | 29    |
| Bike  | 32   | 7     | 82%        | 39    |

Table 1 shows the result of the first case in daylight context, during the duration of a minute video sample where detected 28 cars and 32 motorbikes. The success rate was higher for class car which could reach 97%, whereas bikes reach slightly lower at 82% successful classification rate. This is a promising result in comparison with the original work of virtual lines [2] that could detect only one object per line although the success rate for this one object-one-line context was higher. The following graph depicting the data within table 1 above.

![Figure 10](image1.png)  ![Figure 11](image2.png)

Figure 10. Result in graphs depicting classification result in daylight. Accuracy level could reach 97%.

Figure 11. Result in graphs depicting classification result at minimum light. The accuracy level could reach 69% for car and 70% for bike classification.

The testing result using night video sample as well echoed at the above graph. The main cause for the low accuracy rate at night is minimum light condition that gives minimum motion detection as a result of close similarity between background value and the foreground vehicle movement.

Table 2. Vehicles classification result at night.

| Class | TRUE | FALSE | Percentage | Total |
|-------|------|-------|------------|-------|
| Car   | 18   | 8     | 69%        | 26    |
| Bike  | 35   | 15    | 70%        | 50    |

The outputs produced by the following research are prototype computer applications to identify vehicles in moving images (video) based on vehicle number plates using the Speeded Up Robust Features method; and to support the achievement of integrated and automated transportation management through the application of information technology (IT) and artificial intelligence.
The first test case is applied to vehicle license plate samples with clear images and observed video during the day (good lighting). The Speeded Up Robust Features method can detect the location of the license plate in the observed video. Figure 12 shows the process of detected cars using virtual line sensors as frame grabber function. The result of license plate identification is shown on figure 13 with accuracy of 94% during daylight.

4. Conclusion
The proposed improved method to the previous original work of virtual line sensors which is aiming to improve functionality of line sensors from only counting to classification. The proposed work has improved the efficiency of previous method of multiple lines to detect multiple area, to only one line could cover the entire road width and could detect and classify multiple objects crossing one virtual line. The accuracy of vehicle classification and counting was higher at daylight than at night. Several handicap of night scheme such as headlight reflection on the road and lack of contrast between the background and moving vehicles in a low light condition have been the main contributor to the result. The work reinforces the initial conclusions on the application of SURF in detection applications for motorized vehicle numbers. However, this method is not suitable to be applied to the context of generalization in order to localize non-specific vehicle license plates. Future development could explore more on solving the virtual line implementation during night and solving the headlight reflection problem alongside with a way to improve contrast of background detection during low light live traffic situation.

References
[1] B. Fritz, B. M. Beau, “Absolute acceleration sensor for use within moving vehicles.” U.S. Patent No 10,195,989, 2019.
[2] S.M. Tedjojuwono, “Virtual Lines Sensors for Moving Object and Vehicle Counters”, IAENG International Journal of Computer Science, vol. 44, no. 4, pp 421 -431, 2017.
[3] M. Xue, L. Haibin, “Robust Visual Tracking and Vehicle Classification via Sparse Representation”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 11, 2011.
[4] L. Kang, M. Gellert, “Fast multiclass vehicle detection on aerial images”, IEEE Geoscience and Remote Sensing Letters, no. 12.9, pp. 1938-1942, 2015.
[5] T. Yong, Z. Congzhe, G. Renshu, L. Peng, and Y. Bin, “"Vehicle Detection and Recognition for Intelligent Traffic Surveillance System”, Multimedia Tools and Applications, Springer, vol. 76, no. 4, pp 5817–5832, 2017.
[6] A. Nicolas, L.S. Bertrand, and L. Sébastien, “Segment-before-Detect: Vehicle Detection and Classification through Semantic Segmentation of Aerial Images”, Remote Sensing, vol. 9, no. 4, pp. 368, 2017.
[7] K. Vladimir, G. Alexander, G. Maxim, and S. Lev, “Audience Analysis System on the Basis of Face Detection”, Proceedings of the International Multi Conference of Engineers and Computer Scientists, vol. I, 2013.
[8] R. Vladimir, I. Haroon, S. Mubarak, “Detection and tracking of large number of targets in wide area surveillance”, European conference on computer vision. Springer, Berlin, Heidelberg, pp 186-199, 2010.
[9] A. Alireza, P. Paulo, N. Urbano, “Detection and tracking of moving objects using 2.5 d motion grids”, 18th International Conference on Intelligent Transportation Systems. IEEE, pp. 788-793, 2015.
[10] S. M. Tedjojuwono, “Fast Performance Indonesian Automated License Plate Recognition Algorithm Using Interconnected Image Segmentation”, Intelligence in the Era of Big Data, Springer Berlin Heidelberg, Vol: 516, pp. 289-300, 2015.