Real-time Detection of Illegally Parked Vehicles Using 1-D Transformation

Jong Taek Lee, M. S. Ryoo, Matthew Riley, and J. K. Aggarwal
Computer & Vision Research Center
Dept. of Electrical & Computer Engineering, The University of Texas at Austin
Austin, TX 78712, USA
{jongtaeklee, mryoo, mattriley, aggarwaljk} @mail.utexas.edu

Abstract

With decreasing costs of high quality surveillance systems, human activity detection and tracking has become increasingly practical. Accordingly, automated systems have been designed for numerous detection tasks, but the task of detecting illegally parked vehicles has been left largely to the human operators of surveillance systems. We propose a methodology for detecting this event in real-time by applying a novel image projection that reduces the dimensionality of the image data and thus reduces the computational complexity of the segmentation and tracking processes. After event detection, we invert the transformation to recover the original appearance of the vehicle and to allow for further processing that may require the two dimensional data. The proposed algorithm is able to successfully recognize illegally parked vehicles in real-time in the i-LIDS bag and vehicle detection challenge datasets.

1. Introduction

The problem of tracking cars in different scenarios has been studied widely due to its applicability to numerous practical situations. One significant application is the problem of detecting cars that have been parked in illegal locations. A system capable of accurate, real-time detection of this nature would serve to automate and greatly improve surveillance systems and would be of great value to those monitoring both public and private locales.

In this paper, we present an algorithm for automated detection of illegally parked vehicles in real-time. The contribution of this paper is the development of an image transformation that allows us to perform event detection computation quickly without compromising accuracy and the construction of the entire real-time system in transformed domain.

The proposed algorithm consists of three general steps. First, we perform an image projection that transforms the two dimensional data of each No-Parking (NP) zone into representative one dimensional data. This transformation accomplishes two tasks. First, it reduces the time complexity of the segmentation and tracking tasks. Segmentation of the NP zone is reduced from quadratic to linear time with respect to the length of the images, and tracking is simplified significantly by restricting vehicle motion to a single dimension. Second, the transformation accounts for the variation in size of the vehicle as it moves towards or away from the camera. Utilizing knowledge of the camera's orientation with respect to the NP zones, the transformation produces a representation of the vehicle whose size is consistent throughout the frames.

After transformation to 1-dimension, the segmentation process relies on both background subtraction and consecutive frame subtraction. Tracking is based on matching with a cost function. Merging and splitting operations are performed to account for occlusion of the vehicles. Together, segmentation and tracking allow the automatic detection of vehicles that are parked illegally.

After performing event detection in one dimension, we invert the transformation to restore the original appearance of the vehicles, which may be required for further processing, such as vehicle model classification. Thus, by simplifying the otherwise computationally expensive components of any vision system, our algorithm can make real-time decisions, while preserving all original information. This dramatically increases the practical applicability of our system.

Relevant work is briefly mentioned in section 2. In section 3, the proposed method is described. 1-D projection (section 3.1), segmentation (section 3.2), tracking (section 3.3), and reconstruction (section 3.4) are provided. We present tracking results with comments in section 4. The paper concludes with section 5.

2. Previous work

There is plenty of literature relating to vehicle detection, but there is a paucity of literature on specialized vehicle activities, such as detection of illegally parked vehicles or collisions. However, a patent [1] has been awarded for a

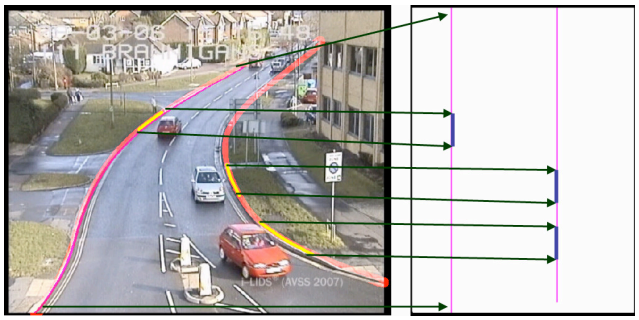

Figure 1*: transformation of curved representative lines of NP-zones to adjusted straight lines (This figure is best viewed in color.)*

system that detects and warns of an illegally parked vehicle. For simple vehicle tracking, Gupte *et al* [2] track blobs using an association graph to connect objects on continuous frames. Song and Nevatia [3] have been able to overcome difficulties arising from occlusions and track vehicles individually. Haritaoglu *et al* [4][5] track objects using concepts of region splitting and merging to handle occlusion.

3. Proposed Method

Our proposed method consists of four consecutive processes: 1-D projection, segmentation, tracking, and reconstruction. After one process is completed, the next step is allowed to proceed.

3.1 1-D Projection

In this sub-section, we present our algorithm to convert all the regions in the image that correspond to NP zones into one-dimensional vectors. Here, the NP zones are the regions where vehicles are not allowed to park, and they are known for the given video sequences. We use corresponding masks to extract these zones in each image. Usually, NP zones lie along the sides of roads, so we focus on the fact that most often roads are curved, and therefore so are the zones of interest. In order to detect and track vehicles in the NP zones more efficiently and reliably, our system transforms these zones into our 1-D representation. Using our proposed transformation, vehicles moving at a constant speed along the direction of the road are transformed into vehicles moving along a straight line at the same speed, and they are normalized to have constant size across every frame.

To this end, we first define a representative line for each NP-zone, which is composed of a set of pixels. This set of pixels on the representative line is mapped onto one straight line, according to its location, as shown in Figure 1. In order to map other pixels in the region of the NP-zone onto the straight line, we first locate for each of those

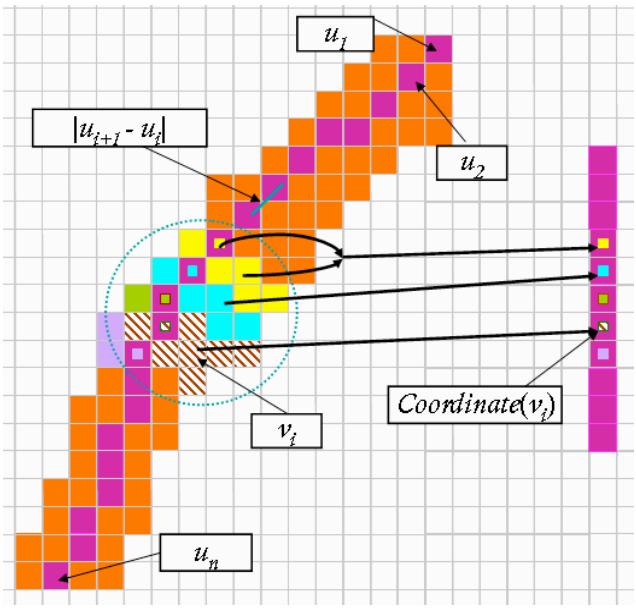

Figure 2*: an image with grid to show a process to define the ordered set of pixels u and to find coordinates of pixels on an NP-zone (This figure is best viewed in color.)*

pixels the closest pixel on the representative line. Those pixels are mapped to the same point on the straight line as the closest pixel in the representative line, we find the mean and variance of the RGB color information of all pixels mapped there. This process is explained more in depth as follows.

Let $u = \{u_i \mid 1 \le i \le n\}$ be the ordered set of pixels representing one edge of an NP zone in the two-dimensional image as shown in Figure 2.

Now, before transformation to 1-dimension, we first define how the transformed coordinate of each pixel u_i on the representative (curved) line u is decided. u is transformed into a new straight line in the form of a 1-D vector which only has vertical location information. This is computed in the following way:

$$Coordinate(u_1) = 0 \tag{1}$$

$$Coordinate(u_{i+1}) = Coordinate(u_i) + Distance(u_{i+1}, u_i) \tag{2}$$

$Coordinate(u_i)$ is the new coordinate of u_i on the straight line, and $Distance(u_{i+1}, u_i)$ represents the normalized distance between the two adjacent pixels, u_{i+1} and u_i, in the original image, such that

$$Distance(u_{i+1}, u_i) = \frac{|u_{i+1} - u_i|}{\underset{c}{mean}\left(Length_c(u_i) / \underset{1 \le i \le n}{mean}\left(Length_c(u_i)/n\right)\right)} \tag{3}$$

Due to this appropriate *Distance* function, the lengths of transformed representative lines for vehicles in Figure 1

are similar. On the right hand side of Eq. (3), the numerator $|u_{i+1} - u_i|$ is the distance between pixel u_{i+1} and u_i in the original image, which is distorted by a camera. To compensate camera distortion errors, the denominator should be proportional to the distance of the point u_i from the camera. To this end, we track several sample cars(car_c) through a sequence and find out the length of cars $Length_c(u_i)$ for all those cars and all pixels of the representative line, u_i. $Length_c(u_i)$ can be estimated either manually or automatically using the information of width and height of car blobs. With the obtained length information, we are able to estimate normalized distance as shown in Eq. (3). The size of objects represented in the transformed 1-D vector can be normalized. As shown in Figure 3 (a) and (b), the size of all similarly-sized vehicles at different locations in the original image becomes similar on the projected image.

After obtaining the new coordinates on the 1-D vector line, we transform the whole image region of a NP zone into the same vector representation as shown in Figure 2. For each pixel v in the image region of an NP zone, we find the corresponding coordinate on the representative line from u_i, which is the closest pixel to v among all pixels on the representative line as shown in Eq. (4).

$$Coordinate(v) = Coordinate\left(\arg\min_{u_i \in u}\left(Distance(v, u_i)\right)\right) \quad (4)$$

Next, the mean and variance of color (in the RGB space) is calculated along a set of pixels $\{v\}$ such that all value $\{Coordinate\ (v)\}$ is same to $Coordinate\ (u_s)$, where u_s is the closest pixel to the set of pixels $\{v\}$. As a result, the image region in 2-D space (associated with the NP zone) is converted into a 1-D vector of color values. The result is shown in Figure 3 (c).

3.2 Segmentation

Segmentation is based on two methods: background subtraction and differencing of adjacent frames in a temporal image sequence. The methods we use are the same as the regular methods used for 2-D images. After 1-D projection, the size of the data we need to handle is much smaller.

For background initialization, we assume that there are no illegally parked vehicles initially. Any background subtraction techniques can be adopted but we use a median filter on the 1-D projection of this initial image sequence in order to obtain backgrounds. To get a general background for a longer video sequence, the background can be updated by accumulating information of updating non-foreground pixels on the initial 1-D projection background and applying median filter again.

After calculating the 1-D background of NP zones, we segment the foreground objects in each frame by subtracting the background from the calculated projection of that frame. Similarly, we subtract subsequent frames to track objects as they move through the extracted zone. By combining these two results, we obtain a very good estimate of the 1-D foreground blobs.

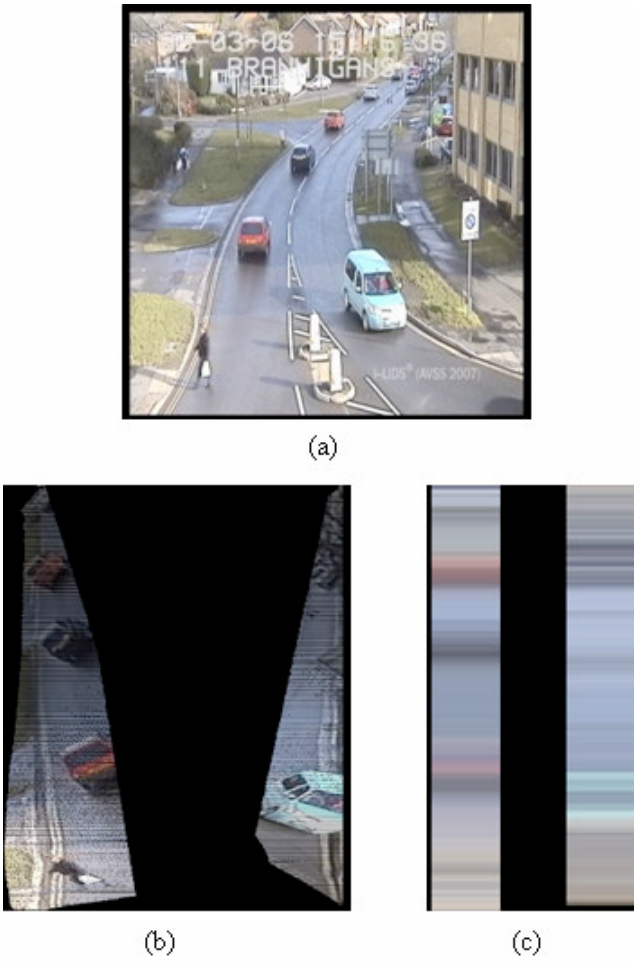

Figure 3*: Result of 1-D projection. (a) the original image (b) intermediate straightened translations of two No-Parking zones (c) two band images displaying the corresponding mean RGB color information. (This figure is best viewed in color.)*

In post-processing, we remove noise in the foreground by performing morphological operations such as dilating, erosion and filling holes [6]. Connected component analysis is used to label the 1-D foreground. Since we are labeling the 1-D line not the 2-D area, the time required to label the blobs decreases. As a result, we obtain a set of regions in the foreground that are likely to represent the true foreground objects as shown in Figure 4. Thus, the time complexity for segmentation is reduced to $O(n)$ in 1-D, whereas the time complexity for segmentation in 2-D is $O(n^2)$.

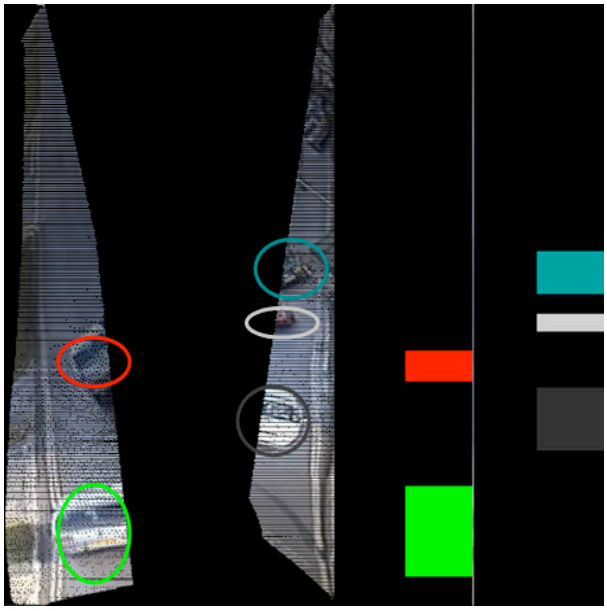

Figure 4*: Result of segmentation. Extracted blobs represent individual objects. (This figure is best viewed in color.)*

3.3 Tracking

In this system, tracking is done based on matching projected blobs in frame t to projected blobs in frame $t+1$. To match blobs, we measure feature distances between blobs on consecutive frames. The total cost function for matching is defined as a weighted sum of the feature distances. The features used are length, location, (R, G, B) color, and velocity. In addition to these features, we also use the duration for which that blob was visible. It is reasonable to expect that longer occurring blobs on the previous frame have a greater possibility of matching blobs on the current frame.

$$Cost\left(b_{k_1,t}, b_{k_2,t+1}\right) = \sum_{f_j \in Features} Difference_{f_j}\left(b_{k_1,t}, b_{k_2,t+1}\right) \times w\left(f_j\right) - Duration\left(b_{k_1,t}\right) \times w\left(Duration\right) \quad (5)$$

Here, $b_{k,t}$ is a k^{th} blob on t^{th} frame. Function $Difference_{fj}$ represents L_1 distance of feature f_j between the two blobs, $b_{k1,t}$ and $b_{k2,t+1}$. Function *Duration* is dependent on the number of frames where a tracked object is shown. In our system, *Duration* function is log function of the number of frames. $w(f)$ is the weight of each feature, with location given the greatest weight. Location is a very reliable feature even in the presence of severe noise. If two blobs are too far from each other then we can assume that the blobs cannot be matched regardless of how well other features of the blobs match. The good and fast match for all blobs is found using a greedy algorithm based on this cost function.

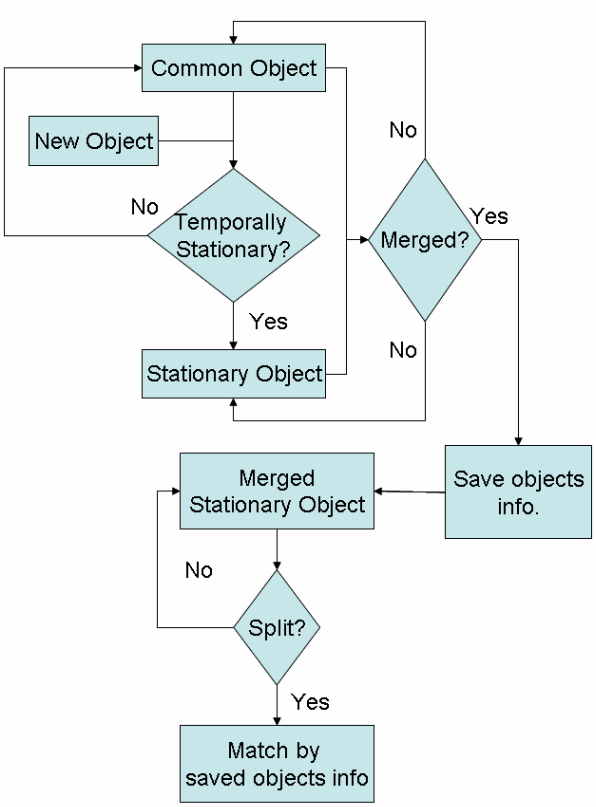

Figure 5*: Flow chart for tracked objects in order to handle occlusions on stationary objects*

When a blob passes another blob occlusion can occur. However, it is easier to handle occlusions that occur in 1-D than in a 2-D image for several reasons. Since all blobs in the 1-D projection can only move in one direction, it is easy to estimate whether the blob is occluded or not. If we are able to recognize blobs merging into each other, then we can detect blob splitting as well. In this case, we only need to handle occlusion of parked cars. A blob is suspected as a parked car - "a probably-parked car" - only if a blob stays stationary for a prescribed duration. This is detected in the following way.

i. All newly introduced blobs are initialized as "New." Existing blobs that are not "Stationary" are marked as "Moving."
ii. If the speed of the blob is less than an expected threshold speed for five continuous frames, we say that the blob is "Stationary."
iii. For a blob marked "Stationary" to retain its status in the next frame, the blob should remain close to the same location where it was first tagged as "Stationary." If the blob fails this criterion, go back to Step i.

The idea about splitting and merging in [4] and [5] motivated our system to handle occlusions. After detecting a probably-parked car, we focus on the blob as follows:

Self-Split/Merge. As a result of noise, the probably-parked blob may be split into smaller blobs. As long as these smaller blobs lie within the region of the probably-parked car, we merge them into one.

Merge. When another blob approaches the probably-parked blob and subsequently merges with it, the current information of the probably-parked blob is saved.

Split. When the two merged blobs split, we decide which blob is more likely to be the previously probably-parked blob by matching them with the blob information saved during merging process, using the cost function defined in Eq. (5).

This algorithm is shown with a flow chart in Figure 5.

The system issues an alert if the probably-parked blob remains "Stationary" for longer than 60 seconds, labeling it as an illegally parked car.

3.4 Reconstruction from 1-D projection

After tracking objects, for additional processing on tracked objects such as classification of objects, it is meaningful to reconstruct original object images from 1-D blob information. The reconstruction process is simple. Using the length and location of each blob in its 1-D projection, we can find the corresponding part of the curved NP zone in the original image. By dilating the part of the curved line, related regions can be obtained as shown in Figure 6.

Figure 6*: Reconstructed image patches of tracked vehicles. The size of the patches can be customized to display as much of the vehicles as desired.*

4. Experimental results

We use the i-LIDS vehicle detection dataset [7] for testing our system. We process every other frame (decimate by 2). We do not want to lose object information, especially the smaller objects in the image, by resampling. Therefore, we do not resample images from the video since we work on a 1-D projection line instead of 2-D images for segmentation and tracking. Tracking results are shown in Figure 7.

Results for all three daytime sequences are accurate. We not only detect the illegally parked cars correctly, but also measure the duration precisely. Table 1 and Table 2 show our results.

For the night time video, the system must be modified to accommodate the effects of headlights. The current system, by itself, was able to detect the illegally parked car correctly for 40 seconds. Due to the glare of the headlights, the timer is incorrectly reset after 40 seconds. However, the system is able to detect the parked car again for the following 20 seconds.

Processing time for each frame is less than 1 second in our sub-optimal MATLAB implementation. Therefore, processing time for each 3 minute long video is about 30 minutes. Real-time processing is certainly possible with an optimized implementation in a more sophisticated programming environment.

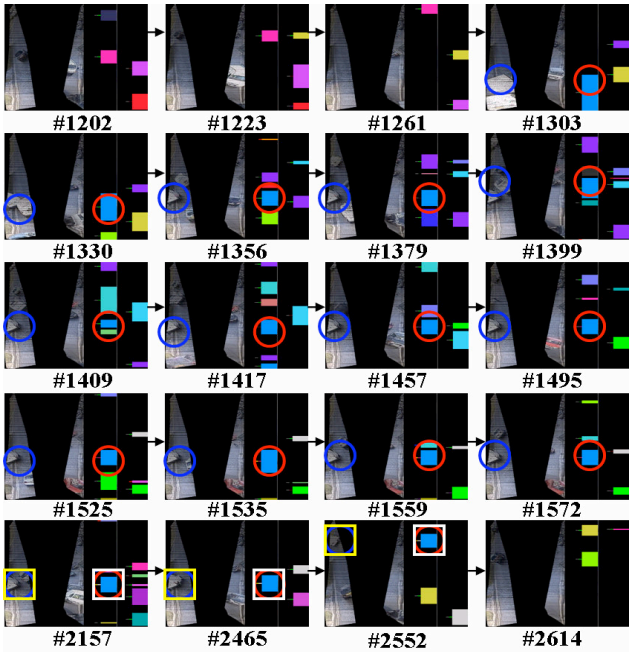

Figure 7*: Image sequence showing tracking results. Suspected cars are marked in ellipses. Cars detected as illegally parked are marked by a box. Each number under the picture is the frame number in the PV_Easy sequence. From #1202 to #1303, all cars are moving and are tracked successfully. Between #1303 and #2552, a car enters the scene, parks illegally and leaves subsequently. Merging and splitting events occur between frames #1525 and #1559. At the frame# 2157, the car is recognized as an illegally parked car and an alarm is raised until #2502. (This figure is best viewed in color.)*

Table 1: *Alarm notification results on the i-LIDS dataset.*

Sequence	Start time		Duration	
	Ground Truth	Our Results	Ground Truth	Our Results
Easy	0:02:48	0:02:52	0:00:27	0:00:27
Medium	0:01:28	0:01:41	0:00:19	0:00:14
Hard	0:02:12	0:02:08	0:00:21	0:00:29

Table 2*: Tracking Accuracy on different sequences.*

Sequence	Tracking Accuracy (%)	
	Illegally parked cars	Other cars
Easy	100	98.3
Medium	99.8	95.2
Hard	100	97.7
Night	99.7	92.5

5. Conclusion

We are able to successfully detect illegally parked vehicles by accurately tracking all the vehicles in the scene. The proposed algorithm, based on the 1-D projection, can be implemented in real-time and is effective even in poor scene conditions. The algorithm benefits greatly from the decreased complexity, allowing us to use a more time-consuming segmentation and tracking procedure. Finally, the 1-D transformation can be reversed, allowing us to reconstruct the original appearance of the objects and therefore enabling future processing steps that require a 2-D representation.

References

[1]. K Morimoto, "System for detecting and warning an illegally parked vehicle" - US Patent 5,343,237, 1994

[2]. S. Gupte, O. Masoud, R. Martin and N. Papanikolopoulos, "Detection and classification of vehicles," IEEE Transactions on Intelligent Transportation Systems, Vol. 3, No. 1, pp. 37-47, 2002

[3]. X. Song and R. Nevatia, "Detection and tracking of moving vehicles in crowded scenes," IEEE Workshop on Motion and Video Computing, Austin, TX, 2007.

[4]. I. Haritaoglu, D. Harwood, and L. Davis, "W4S: A Real Time System for Detecting and Tracking People in 2.D," European Conference on Computer Vision, Vol. I, LNCS 1406, pp. 877-892, Freiburg, Germany, 1998

[5]. I. Haritaoglu, D. Harwood, and L.S. Davis, "W4: Real-Time Surveillance of People and Their Activities," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 809-830, Aug 2000

[6]. M. L. Comer and E. J. Delp, "Morphological operations for color image processing," Journal of Electronic Imaging, Vol. 8, No. 3, pp. 279-289, 1999

[7]. i-Lids dataset for AVSS 2007, http://www.avss2007.org