Integrating 3D structure into traffic scene understanding with RGB-D data

Yingjie Xia\textsuperscript{a}, Weiwei Xu\textsuperscript{a,\textasteriskcentered}, Luming Zhang\textsuperscript{b}, Xingmin Shi\textsuperscript{a}, Kuang Mao\textsuperscript{c}

\textsuperscript{a} Hangzhou Normal University, China
\textsuperscript{b} School of Computing, National University of Singapore, Singapore
\textsuperscript{c} College of Computer Science, Zhejiang University, China

\textbf{A R T I C L E  I N F O}

Article history:
Received 15 November 2013
Received in revised form 25 March 2014
Accepted 26 May 2014
Available online 31 October 2014

Keywords:
Traffic scene understanding
Depth data
3D structure
Vehicle detection
Pedestrian detection
Overtaking warning

\textbf{A B S T R A C T}

RGB Video now is one of the major data sources of traffic surveillance applications. In order to detect the possible traffic events in the video, traffic-related objects, such as vehicles and pedestrians, should be first detected and recognized. However, due to the 2D nature of the RGB videos, there are technical difficulties in efficiently detecting and recognizing traffic-related objects from them. For instance, the traffic-related objects cannot be efficiently detected in separation while parts of them overlap, and complex background will influence the accuracy of the object detection. In this paper, we propose a robust RGB-D data based traffic scene understanding algorithm. By integrating depth information, we can calculate more discriminative object features and spatial information can be used to separate the objects in the scene efficiently. Experimental results show that integrating depth data can improve the accuracy of object detection and recognition. We also show that the analyzed object information plus depth data facilitate two important traffic event detection applications: overtaking warning and collision avoidance.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

In intelligent transportation systems (ITS), traffic flow is one of the most used indices for characterizing traffic conditions to be used in traffic control and transportation management [44]. Traditionally, the data of traffic flow are collected by inductive loop detectors [12], global positioning system (GPS) probe vehicles [31], and remote traffic microwave sensors [43]. However, all these detection devices have their inherent drawbacks [14]. The major disadvantages of inductive loop detectors are high failure ratios and high maintenance costs. The main shortcomings of GPS probe vehicles are poor statistical representation and high error rates in the map-matching, and the main disadvantages of RTMS are high installation costs and inaccurate estimation of traffic state features.

Recently, video devices have been widely deployed for traffic surveillance. The video detectors become the primary sensor to detect traffic flow from roadside or overhead mainly for the following reasons [22]: (1) People are more used to visual information than other forms of sensor data; (2) Video sequences can directly reflect the status of transportation systems by a broad time-varying range of information; (3) Video detectors can be installed, operated, and maintained easily and in low cost. Therefore, the detection, recognition, and tracking of the traffic-related objects, such as vehicles and pedestrians from the captured videos provide the critical basis for ITS applications [41,30]. Significant improvements in traffic scene understanding have been achieved in such 2D image representation based algorithms. However, there are still technical issues remaining to be solved in practice. First, the traffic-related objects cannot be efficiently detected in separation while parts of them overlap; Second, complex outdoor environments increase the difficulty to the vehicle and pedestrian detection since object detection will be influenced by the background; Moreover, it is difficult to design a system robust to detect vehicle movement and drift with 2D image representation.

With the popularization of RGB-D camera, users can now have low-cost and easy-to-use devices, such as Microsoft Kinect, to capture 3D representation of a scene in the format of depth data [11]. Therefore, recent researches in computer vision community have made great efforts on improving the robustness and accuracy of object localization and recognition by integrating 3D representation into the analysis pipeline. Local geometry features from depth data are used to analyze and segment the indoor scene images in high accuracy [35]. In [5], depth kernel descriptors was developed to improve the object recognition accuracy. A recent contribution also investigates how to accurately localize the 3D objects with the assistance of depth data [23]. Inspired by these pioneering research works, it is worth investigating how to use...
depth data in the traffic scene understanding algorithms to handle
the above technical issues.

The major contribution of this paper is a robust, RGB-D data
based traffic scene understanding algorithm. The 3D structure
information of a traffic scene is captured by Microsoft Kinect. The
algorithm starts with the computation of local 2D plus 3D features
for the captured RGB-D data. Afterwards, the random forest algo-
rithm is adopted to learn an efficient pixel-level classifier from
the features as the basis to low-level understanding of traffic scene [6].

A segmentation and labeling algorithm based on graph-cut is then
used to segment the RGB-D images into object-level, which is ready
for various high level applications in traffic surveillance.

We have tested our algorithm on a variety of traffic scene images
which contains different kinds of traffic objects, such as car, bicycle
and pedestrians. Experimental results show that depth data can
largely improve the object detection accuracy and facilitate the
subsequent high-level traffic surveillance applications.

2. Related work

2D image based traffic scene understanding: The kernel of traffic
scene understanding is traffic-related object detection, recogni-
tion, and analysis, including vehicle detection [41], pedestrian
detection [30], license plate recognition [2], and pedestrian count-
ing [39].

The detection, recognition, and analysis of vehicles and pedes-
trians have broad applications in ITS. Vehicle detection and
recognition are used for identifying cases of traffic violation, which
is the main cause of traffic accidents [29]. One of the vehicle
detection methods is designed to divide video frames into sub-
regions and extract local features from sub-regions to enable the
detection less susceptible to the variance of vehicle poses, shapes,
and angles [41]. In order to precisely separate a vehicle with its
neighborhood vehicles, Sivaraman and Trivedi integrate active-
learning and particle filter tracking to implement an on-road
vehicle detection system [37]. Cherng et al. propose a dynamic
vehicle detection model which visually analyzes the critical
motions of nearby vehicles in video [9]. However, these works
have not efficiently solved some special cases, such as vehicle
overlapping happens. To detect the vehicles in complex traffic
scenes is very useful for multiple ITS applications.

The pedestrian detection is also very important for the effective
traffic scene understanding. For example, pedestrian detection can
reduce the occurrence of pedestrian-and-vehicle-related cases,
such as collision accidents. Cao et al. use a classifier to identify
the risky regions based on vehicles from the video data, and
evaluate the risk of pedestrians by the estimated distances
between pedestrians and risky regions to avoid accidents [7].
Munder et al. utilize a Bayesian method on multiple features of
shape, texture, and 3D information to detect pedestrians in urban
[30]. In night time, Ge et al. use a monocular near-infrared camera
to detect and track pedestrians in real-time [17].

Night-vision systems are also used to model the pedestrian detection by
the probability calculated through a function with various pedestrian
features [4]. In these related work, RGB-D camera can be effec-
tively used to detect pedestrians because it uses infrared light to
capture depth information, while it still has not been employed for
pedestrian and vehicle identification in their mixed occurrence.

From video data, license plate recognition is a basic module in ITS
aiming to identify and locate the vehicle. Typical license plate
recognition consists of two steps, license plate location and characters
recognition. Morphological and chromatic processing on frames of
traffic video is widely used in license plate location. The morpho-
logical processing is based on morphological features of license plates
[21]. Some approaches utilize histograms of gray-scale images, which
are not available when incomplete characters exist or the background
is too complicated [22]. As for chromatic processing, some work uses
the specific color to locate the license plate region, while it is fragilely
interfered by the illumination changes and other similar colors in the
image [1]. In the characters recognition, the template matching
method is widely used. This method does not work well for the
images with a lot of noise, and the recognition results heavily depend
on the chosen templates [8]. Neural network is another commonly-
used approach to recognize characters of license plates [28].

As the statistical traffic data analysis on video data, pedestrian
counting is particularly useful in some special cases, such as
emergency evacuation [10]. Video is a low-cost and effective
device to implement the pedestrian counting. Zhang et al. extract
high-dimensional statistical features from the pedestrian video
data, and adopt the supervised dimension reduction technique to
select the representative features [18,45]. Tan et al. propose a
semi-supervised elastic net model based on the relationship
between each frame and its neighboring frames to achieve
pedestrian counting [39].

In addition to the aforementioned related work, the traffic signs
[3] and lanes detection [26] by traffic videos are also very
important for ITS applications. Since 2D cameras have been widely
deployed on roads, there are few applications using 3D traffic
video data. However, the depth information is very useful under
some special circumstances, such as detecting the overlapping
traffic-related objects. It is valuable to investigate how to use RGB-
D data to interpret traffic scenes in ITS applications.

RGB-D data based scene understanding: Depth data can be
exploited to the learning of discriminative object features and
the analysis of the 3D scene structure, which has proven to be
successful in scene understanding applications.

A typical application of RGB-D data is indoor scene under-
standing. Silberman et al. [35] collected a database of indoor scene
RGB-D images, and developed RGB-D SIFT descriptor to improve
the segmentation and labeling accuracy of indoor scene RGB-D images.
Koppula et al. learned a highly accurate indoor scene
object classifier through mixed integer optimization [27], and
achieved around 80% accuracy of depth data labeling. In computer
graphics, RGB-D data has been used in 3D indoor scene recon-
struction applications. [36,25,34]. Depth data is important in
correct analysis and reconstruction of the 3D indoor scene layout
in such applications. Besides geometric properties, there is also
research work on how to derive physical interactions between
objects in the scene with depth data, such as structural stability
and supporting relationship analysis [50]. These algorithms can be
combined with super-pixel or graphlet representation to acceler-
ate the RGB-D image segmentation [32,46,48,49].

RGB-D data can also be used in object recognition and retrieval
[40,16]. Research efforts have been devoted to view-invariant 3D
shape or depth data feature descriptors [42,15]. Histogram of
oriented depth is used in human detection in RGB-D images
[38]. Depth kernel descriptors developed by Bo et al. applies
match kernel to the local patch based geometric features to
generate highly discriminative and robust geometric features [5].
Integrating it into various classifier algorithms, such as support
vector machine and random forest, results in more accurate object
recognition results. With the assistance of depth data, accurate 3D
object localization in captured RGB-D images can be realized by
learning a segmentation mask in the 2D bounding box of an
extracted object in the image [23].

3. Traffic scene segmentation and labeling algorithm

The goal of segmentation and labeling is to obtain an object-
level traffic scene image understanding. That is, the captured
images are labelled into semantic regions, such as vehicles and pedestrians. In the following, we describe how to aggregate pixel-level classification results into object level representation.

3.1. Learning pixel-level classifier

For each pixel $i$, we apply a random forest algorithm to classify it into traffic object class labels $L = h(f_i; \Theta)$, where $h$ represents the random forest classifier and $\Theta$ the vector of random split parameters at nodes for the construction of decision trees. $f_i$ is the feature vector computed from the local patch around pixel $i$. The random forest algorithm is fast in training and testing, and its generalization error is upper bounded. We found it works well in handling large numbers of training data in our case (we have around 500 thousand patches sampled from the captured RGB-D images).

**Training:** We train the random forest classifier through 11 captured RGB-D video sequences. For each image in the sequences, we sample $15 \times 15$ patches around pixels, identical to the patch size in SIFT descriptor, with 5 pixels strides. For each sampled patch, we compute following 2D plus 3D features: Histogram of Gradients (HOG), Normal structure tensor, Geometry moments and Spin image, where the latter 3 features are computed from depth data. The details of feature computation are in the next section.

We adopt random split selection strategy in the training of random forest. That is, in non-leaf node splitting, simple decision stumps are tested on a randomly selected feature channel. Precisely, a large number of randomly generated thresholds $\tau$ are tested on the feature channel $F_c$. The patches satisfying $F_c > \tau$ forms the set of the patches in its right child, and the rest forms the left child.

The threshold $\tau$ maximizing the information gain $IG$ is the final parameter used to split the node into right and left children. IG is based on the concept of entropy from information theory, which is computed by the following formula:

$$IG(C) = - \sum_{c \in \{L,R\}} w_c H(c)$$  \hspace{1cm} (1)

where $H(C) = \sum_c - p(c) \ln p(c)$ is the entropy, and $p(c)$ is the probability of traffic object class label. We calculate $p(c)$ as the percentage of class $c$ in the number of patches in the node. The leaf node is created when the number of patches is below 20 or the tree reaches the maximum depth.

**Testing:** In the testing, we first compute feature vector from the $15 \times 15$ patch $P_i$ around the pixel $i$, send it to the trees in the forest, and then aggregate the result from each tree with the formula below:

$$p(c|P_i) = \frac{1}{K} \sum_{j = 1}^K p_j(c|P_i).$$  \hspace{1cm} (2)

where $K$ is the number of trees in the forest, and $p_j(c|P_i)$ is the probability of class $c$ which is also the percentage of patches with label $c$ in the arrived leaf node of tree $j$.

3.2. Feature vectors

In this section, we detail the features used in pixel-level classifier: Histogram of Gradients (HOG), Normal structure tensor, Geometry moments and Spin image, and they are of dimension 36, 36, 60, 256 respectively, resulting in a feature vector of dimension 388. Since HOG is widely used in human detection in computer vision community and its details can be found in [13], we focus on the latter 3 geometric features: normal structure tensor, geometry moments and spin image.

(a) Normal structure tensor: It is used to measure the principle directions in the normal distribution of pixels in the patch. We subdivide the patch into $2 \times 3$ sub-patches so that the local normal distribution can be efficiently measured. For each sub-patch, a tensor is computed by the following formula:

$$G_i = \frac{1}{N} \sum_{i=1}^N n_i n_i^T,$$

where $n_i$ is the normal at pixel $i$. $G_i$ is normalized by its Frobenius norm, i.e. $G = \|G\|_F$, as the final normal structure tensor feature at the sub-patches.

(b) Geometry moments: Geometry moments are also computed at 6 sub-patches similar to normal structure tensor. For each sub-patch, 10 moments for $(x, y, z)$ coordinates are computed with following equation:

$$M_{pq} = \frac{1}{N} \sum \alpha^p \beta^q z^r, p + q + r < 3.$$

In the computation of moments, the $(x, y, z)$ coordinates are normalized according to local axis-aligned bounding box of 3D points in the sub-patch.

(c) Spin image: Spin image measures the local geometry feature around a 3D point $v$ by projecting 3D points on the surface into the tangent plane associated with $v$ [19], as illustrated in Fig. 2. Specifically, give a 3D point $v$ and its normal $n$, other 3D point $x$ can be parameterized on its local tangent plane:

$$S_x = \{ (\alpha, \beta) = (\sqrt{\|v - x\|^2 - (n \cdot (v - x))^2}, n \cdot (v - x)) \}$$  \hspace{1cm} (3)

In our case, we use the central pixel of the patch and its depth data to calculate normal and then its spin image feature with $16 \times 16$ bin resolution, yielding 256 features. To reduce the influence of noise, the principal directions at patch level is used as normal in spin image feature computation.

3.3. Object-level segmentation

After pixel-level classification training, we obtain a classifier to compute the class label probability at each pixel. The next step is to aggregate such information to segment the traffic scene RGB-D image into objects. The object segmentation problem can be posed as a pixel labeling algorithm and formulated by a conditional random field (CRF):

$$E(C) = \sum_{ij} E_2(c_i, c_j) + \alpha \sum_{ij} E_3(c_i, c_j)$$  \hspace{1cm} (4)

where $C$ denotes the image labeling, $E_2(c_i, c_j)$ is the data term, computed as the $-\ln p(c_i)$, where $p(c_i)$ is the output of the trained random forest classifier. $E_3$ is the compatibility term:

$$E_3(c_i, c_j) = \delta(c_i \neq c_j) \text{sim}(v_i, v_j)$$  \hspace{1cm} (5)

where $\delta$ denotes the Kronecker delta function, and $v_i = (r_i, g_i, b_i, d_i)$ denotes the RGB-D pixel values at pixel $i$, and $\text{sim}(v_i, v_j) = \exp(-\|v_i - v_j\|^2/2\delta)$ is a function to measure the similarity between two neighboring RGB-D image pixels. While computing $\text{sim}(v_i, v_j)$, the depth channel can be ignored if the depth value in any pixel is missing, since it is difficult for the infrared light based Kinect camera to handle transparent object, such as the windows of a car, and depth data is of high probability missing there.

**Postprocessing:** We first perform simple region growing to group pixels with same class label into regions if the depth difference between two pixels is less than 20 cm. Due to occlusions in the image, a vehicle might be segmented into several regions as shown in Fig. 1. We adopt a simple rule to handle it: if two vehicle regions are separated by a pedestrian object or background and their maximum depth difference is below a
threshold, 50 centimeters in our implementation, we group these two regions into one region. This process maybe repeated in a cluttered scene. The assumption of this region merging operation is based on the characteristic of traffic scenes: it is rare that a vehicle can be occluded into two parts by another vehicle.

4. Application cases

In this section, we discuss two traffic surveillance applications, overtaking warning and collision avoidance, which necessitate the semantic objects and their spatial information analyzed by the RGB-D data based traffic scene understanding algorithm.

4.1. Overtaking warning system

Poor lighting or other complex environmental conditions might increase the chances of misjudgement in vehicle overtaking scenarios, thus increase the risk of accidents. We aim to improve the performance of automatic overtaking warning system by traffic-related objects recognition using RGB-D data.

Our implementation of the overtaking warning system consists of four components: (1) vehicles and pedestrians detection by the segmentation and labelling algorithm, (2) average speed calculation by virtual loops, (3) position estimation using 3D information, (4) overtaking judgment. Since vehicle and pedestrian detection has been discussed in Section 3, in this section, we will focus on parts (2), (3), and (4) as follows.

4.1.1. Average speed calculation

Once vehicles are detected from RGB-D data, we then identify the drive-in-loop and drive-off-loop of objects to calculate their average speed. As shown in Fig. 3, in a video clip, we choose the frames in which one vehicle enters and leaves the virtual loop (bold rectangle in Fig. 3) as the marking frames, and calculate the average speed by the following equation:

\[ \bar{v} = \frac{S}{t_m - t_n} \]


4.1.2. Position estimation

The 3D positions of traffic-related objects can be calculated through depth data, which can be utilized to answer two questions in the overtaking scene: whether one object can overtake its preceding object and on which side the overtaking will happen. As illustrated in Fig. 4, the relative 3D-position estimation is implemented by the following equation:

\[ (\Delta x, \Delta y, \Delta z) = (x_2 - x_1, y_2 - y_1, z_2 - z_1) \]

where \((\Delta x, \Delta y, \Delta z)\) denotes the differences of the right boundary center \((x_1, y_1, z_1)\) and \((x_2, y_2, z_2)\) of two objects detection rectangles.
in 3D space. If the right boundary center does not belong to the object, we replace the center point with its nearest point in the object for computation.

4.1.3. Overtaking interpretation

Based on the average speed calculation and position estimation, the overtaking model of the traffic-related objects can be evaluated as follows.

\[
\begin{align*}
L_{\text{overtaking}} &= (x_B > x_F) \& \& ((v_B - v_F) > 0) \& \& (z_B - z_F) > 3) \quad (8a) \\
R_{\text{overtaking}} &= (x_B > x_F) \& \& ((v_B - v_F) > 0) \& \& (z_B - z_F) > 3) \quad (8b)
\end{align*}
\]

where \(L_{\text{overtaking}}\) and \(R_{\text{overtaking}}\) denote the Boolean prediction of whether the overtaking happens on the left side or on the right side. \(F\) stands for the front traffic-related object and \(B\) stands for the back object. \((x_B - x_F) > 0\) means that the estimated \(x\) position of the back object is behind that of the front object. \((v_B - v_F) > 0\) means that the calculated average speed of back object is greater than that of the front object. Both conditions are necessary for traffic-related object overtaking occurring. The third condition is used to judge from which side the back object overtakes the front object. The depth information got in RGB-D data is used to calculate \((z_B - z_F) > 3\) or \((z_F - z_B) > 3\), which means that the overtaking occurs from the respective left side or right side under the assumption that the width of a lane is 3 m. This model can interpret the overtaking scenario and deliver the warning message to traffic-related objects.

By applying the overtaking rule to the cases in Figs. 3 and 4, the overtaking warning can be generated and delivered as Fig. 5. The letter in the circle in red represents the predicted side of overtaking.

4.2. Collision avoidance system

Most traffic accidents are caused by collision, including vehicle–vehicle and vehicle–pedestrian collision. Some research have demonstrated that if the objects can be warned before the accident (1.5 s in advance), 90% of such incidents can be avoided. Thus, the introduction of automatic collision avoidance systems can effectively reduce the number of traffic accidents. Such systems automatically analyze the spatial and speed relationship of objects to extrapolate the risk of accident. Since the video data from RGB-D camera contains the position and speed information of objects, it
can be utilized to forecast the behaviors of objects and avoid either vehicle–vehicle or vehicle–pedestrian collision. For proof of concept, we only take vehicle–vehicle collision as the example.

In our method, we classify the collision warning into two categories. The first category is emergency warning, which correspond to collisions that may happen in less than 1.5 s. The emergency warning method needs to be of low-complexity and should be processed in real-time. The second is moderate warning, which correspond to collisions that might happen in up to 3.5 s. The moderate warning needs to estimate the possibility of collision based on the vehicle trail recorded in video data.

Emergency warning situation is detected when distance between vehicles is smaller than the safety distance $D(v)$, which is a function of the relative velocity of the two vehicles, pavement behavior, and the reaction time of drivers in $[44]$. In each cycle, the position $p_i = (x_i, y_i)$ and speed $v$ of vehicles $h_i$ can be calculated through the video data, where $i$ is the vehicle’s index. Here, the position is described in a rectangular coordinate system, where the origin is the position of the camera, and $x$-axis and $y$-axis direct along and perpendicular to the road respectively. If we ignore the vehicle width, when the distance of two vehicles $|h_i, h_j|$ satisfies $\|p_i - p_j\|_2 < D(v)$, the system will raise the alarm. We assume that $v_{x}$ and $v_{y}$ represents the speed component of $x$-axis and $y$-axis respectively. If the size is considered, when $\|w_i - w_j\|_2 < D(\sqrt{v_{x}^2 + v_{y}^2})$ is satisfied, a head-on collision ($v_i$ and $v_j$ in the same direction) or a rear-end collision ($v_i$ and $v_j$ in the opposite direction) may happen, where $L/W$ represents the vehicle’s length/width and is the position of vehicle’s center; when $\|w_i - w_j\|_2 < D(\sqrt{v_{x}^2 + v_{y}^2})$ is satisfied, a scratch may happen.

For the moderate warning, our method also presents the predicted moving trail of vehicles according to their previous positions ($p^t_{t−1}, p^t_{t−2}, ..., p^t_{t−m}$) and the probability of collision as depicted in Fig. 6. In a short period of time, the moving trail or its differencing can be considered as a steady signal. Autoregressive Integrated Moving Average (ARIMA) models are the most general class of models for forecasting a time series with moderate computational complexity, which can be stationarized by transformations such as differencing and logging. They are fitted to time series data for better understanding of data and predicting future points. Thus, ARIMA model is adopted for curve prediction in our method, which is defined as

$$p^t_l = \phi_1 p^t_{l−1} + \phi_2 p^t_{l−2} + \cdots + \phi_m p^t_{l−m} + \theta_1 \eta^t_l + \theta_2 \eta^t_{l−1} + \cdots - \theta_n \eta^t_{l−n},$$

(9)

where $\phi_1, \phi_2, ..., \phi_m$ are regression coefficients, and $\theta_1, \theta_2, ..., \theta_n$ are average coefficients. In this model, the current position of the vehicle is a linear regression of its present and previous stochastic errors, together with its previous positions. In the process of forecasting, all the data collected when the vehicle appears in the camera covered area is used for parameters training by method of moments, and then future positions ($p_{t+k}, 1 < k < K$) of all vehicles $\Phi$ in the camera are estimated. From $t+1$ to $t+k$, if $\|p_t^l - p_j^l\|_2 < D(\sqrt{v_{x}^2 + v_{y}^2})$, the probability of collision $\gamma_l$ is calculated as $\gamma_l = 1 - k/K$. As $\gamma_l$ is a monotone decreasing function and $\gamma_l \in [0, 1)$, the less time interval means the bigger probability of collision.

5. Experimental results

We have tested the traffic scene image segmentation and labeling algorithm on a desktop PC with 2.6 Ghz, Dual core Intel i5 CPU. The RGB-D images captured from Microsoft Kinect camera

![Fig. 5. Applications of overtaking warning system. (a) Overtaking on the left side. (b) Overtaking on the right side.](image)

![Fig. 6. Sketch map of collision forecasting.](image)
if of 640x480 resolution. The backgrounds in such images are first detected thorough adaptive Gaussian-mixture model [20], a common choice in traffic surveillance applications, and we focus on how to identify the foreground objects into three types: vehicle, bicycle and pedestrians, since these three types of objects frequently occur in daily traffic scenes. We first use Labelme tool to label the captured RGB images of different vehicle and pedestrian settings [33], and sampled 61,200 labeled patches for training.

Table 1 lists the statistics of the experiments on pixel-level classification, where 2D + 3D indicated we use both 2D HOG feature and 3D geometric feature derived from depth data. Random forest classifier and 2D + 3D features achieves the highest accuracy in our experiments, which is around 10% higher than pure 2D features. Fig. 8 visualizes the comparison of the pixel level classification results, and the influence of random forest parameters to classification accuracy is illustrated in Fig. 7. We also test support vector machine (SVM) on our RGB-D patch data-set. While SVM are known to perform very well on medium scale data-set, its performance is not superior in our case of large scale high dimensional data-set. The labeling results of different traffic scene images are shown in Fig. 9 to show the robustness of our algorithm.

The importance of each geometry feature to the final recognition accuracy is reported in Table 2 for the same vehicle and pedestrian scene in Table 1. It is obvious that all kinds of geometry features can be used to improve the recognition accuracy, but none of them seems to dominate the quality of the final result. The usage of geometry moment leads to the highest accuracy gain for random forest classifier, while the spin image is the most important in Adaboost classifier according to the experiment. The table also shows that the random forest classifier gains higher accuracy than Adaboost, 83.4% vs 73.4%, in our application. Random forest usually compares favorably to Adaboost algorithm in recognition test. However, the random forest algorithm is more robust to noisy data [6], which is more suitable in our case due to the reason that

<table>
<thead>
<tr>
<th>Classifier + feature</th>
<th>Traffic scenes</th>
<th>Vehicles + pedestrians (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM 2D</td>
<td>64.25</td>
<td></td>
</tr>
<tr>
<td>SVM 2D + 3D</td>
<td>70.12</td>
<td></td>
</tr>
<tr>
<td>RF 2D</td>
<td>79.2</td>
<td></td>
</tr>
<tr>
<td>RF 2D + 3D</td>
<td>83.4</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 7. Classification accuracy with respect to random forest parameters. Left: Tree numbers in the forest. Right: Tree depth (Number of trees is fixed to 100 in this curve).

Fig. 8. Pixel-level classification result. 2D + 3D features achieve around 10% higher accuracy than pure 2D feature. The accuracy is computed by comparing the classification results to the ground-truth labeling in the testing image.
there exist sensor noises in the depth data captured by RGBD cameras.

Performance: The training time of the random forest algorithm is 20 min, and it takes around 0.6 s to test a new image. Graph cut algorithm to achieve the final labeling result is around 0.06 s. Please note that all the performance statistics are from our unoptimized serial implementation. It can be significantly accelerated through parallel implementation of random forest and super-pixel based image representation.

6. Conclusions and future work

We have developed a RGB-D data based traffic scene understanding algorithm. By integrating 3D structure, traffic-related objects, such as vehicles and pedestrians, can be robustly detected and recognized even when the objects overlap with each other. Two traffic surveillance applications are also developed to show the advantage of RGB-D data in determining the 3D spatial relationship of the traffic-related objects.

In the future, we plan to learn more discriminative depth features to further improve the detection and recognition accuracy. Although pixel-level classification is relatively expensive, it can provide fine-grained information of traffic-related objects and result in more precise object locations. It is worth further investigating how pixel-level classification can help in traffic surveillance applications. We also plan to explore how to the usage of global structural information in traffic scene understanding, inspired by recent work in [45,47]. Finally, Microsoft Kinect can only capture depth information of objects in short distance. We plan to explore the application of other long range 3D sensors and expect more interesting applications.

Acknowledgments

This paper draws on work supported in part by the following funds: National High Technology Research and Development Program of China (863 Program) under Grant number 2011AA010101, National Natural Science Foundation of China under Grant number 61002009 and 61304188, Key Science and Technology Program of Zhejiang Province of China under Grant number 2012C01035-1, and Zhejiang Provincial Natural Science Foundation of China under Grant number LZ13F020004. Weiwei Xu is partially supported by NSFC (Nos. 61272392 and 61322204).

References

Xingmin Shi received his M.S. degree from the School of Computer Science, Beijing Institute of Technology in 2004. Now he is a Ph.D. student in College of Information Engineering, Zhejiang University of Technology, Hangzhou, Zhejiang, PR China, as well as an associate researcher in the Hangzhou Normal University, Hangzhou, Zhejiang, PR China. His research interests include data mining, image and video processing, and applications in intelligent traffic systems.

Kuang Mao is a Ph.D. student at College of Computer Science, the Zhejiang University since 2010. His research area includes recommendation system, singing song recommendation, graph ranking algorithms and probabilistic modeling.