J. Vis. Commun. Image R. 25 (2014) 165–183

Contents lists available at SciVerse ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Fusion of 3D-LIDAR and camera data for scene parsing

Gangqiang Zhao [a,*], Xuhong Xiao [b], Junsong Yuan [a], Gee Wah Ng [b]

[a] School of EEE, Nanyang Technological University, Singapore
[b] DSO National Laboratories, Singapore

ARTICLE INFO

Article history:
Available online 19 June 2013

Keywords:
Scene parsing
Velodyne scanner
Camera
Fuzzy logic
Temporal fusion
MRF
Object detection
RGBD

ABSTRACT

Fusion of information gathered from multiple sources is essential to build a comprehensive situation picture for autonomous ground vehicles. In this paper, an approach which performs scene parsing and data fusion for a 3D-LIDAR scanner (Velodyne HDL-64E) and a video camera is described. First of all, a geometry segmentation algorithm is proposed for detection of obstacles and ground areas from data collected by the Velodyne scanner. Then, corresponding image collected by the video camera is classified patch by patch into more detailed categories. After that, parsing result of each frame is obtained by fusing result of Velodyne data and that of image using the fuzzy logic inference framework. Finally, parsing results of consecutive frames are smoothed by the Markov random field based temporal fusion method. The proposed approach has been evaluated with datasets collected by our autonomous ground vehicle testbed in both rural and urban areas. The fused results are more reliable than that acquired via analysis of only images or Velodyne data.

1. Introduction

Autonomous situation awareness is an important research aspect for robots and unmanned vehicles. Besides whether the terrain is traversable, they also require more specific object category information to carry out their tasks: e.g., approaching a tree, or the water area. For decades, computer vision approaches have been studied to classify scenes from images. Studies of the human visual system show us that scene perception is a highly complex process of information fusion which involves not just the human eyes, but also other human senses including hearing, tasting, etc. Even within a human vision system, there is clearly fusion of information from color, motion, depth and a whole variety of ways to infer shape, movement and physical characteristics of the things within the view [1]. In other words, efficient perceptual performance often requires integration of multiple sources of information, both within the senses and between them. As a matter of fact, other sensors like infrared laser projector in Kinect [2] and LIDAR scanners [3] have been applied to complement video cameras in recent years.

In this work, in order to help unmanned vehicles to understand their environment, two sensors are used: Velodyne HDL-64E 3D-LIDAR scanner [3] and monocular video camera. A Velodyne scanner provides 3-dimensional but sparse pointcloud of the surrounding environment. The pointcloud is trustworthy for obstacle detection but lacks color and texture information, which is valuable for more detailed categorization of objects. Besides, although Velodyne HDL-64E is a powerful LIDAR scanner in the market, its effective coverage limits within 70 m from the center of the sensor. Considering some time will be taken for information processing and task scheduling, the 70 m distance may not be sufficient for an unmanned moving vehicle to respond. Furthermore, for some tasks, we hope the vehicle can "see" as far as 200 m for advanced planning. On the contrary, images captured by video cameras can easily cover a much broader and further area and provide more discriminative information to classify objects into categories. However, due to the lack of depth information, image-based detection of obstacles of various shapes, sizes and orientations remains challenging. Due to the above-mentioned complementary features between cameras and LIDAR sensors, it is possible to acquire more reliable scene parsing by fusing information derived from these two sensors.

In addition, the sequential scene parsing also requires fusing results of consecutive frames. In fact, even after fusing results of two sensors, the obtained parsing results of consecutive frames may have abrupt changes due to stochastic errors. These abrupt changes of parsing results may confuse the vehicle navigation system. Intuitively, it is possible to obtain more cohesive sequential parsing results by including the temporal fusion.

In this research, we first propose a new way to fuse the results of two sensors by employing fuzzy logic inference [4]. Then we propose a Markov random field based approach to fuse the results of consecutive frames. Fig. 1 illustrates the fusion process. Fuzzy logic is preferable for our application due to its advantages. First, fuzzy logic is built on top of the knowledge and experience of

* Corresponding author.
E-mail address: GQZhao@ntu.edu.sg (G. Zhao).

1047-3203/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jvcir.2013.06.008

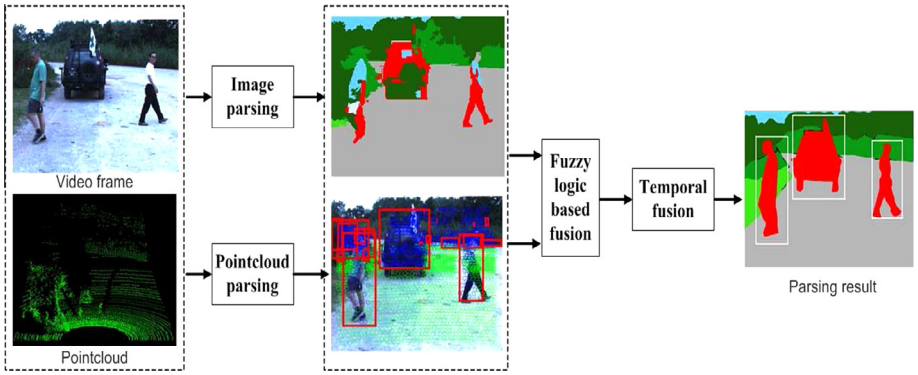


Fig. 1. Illustration of the proposed sensor fusion approach. The data of Velodyne scanner and camera are first parsed simultaneously. Then the results of two sensors are fused by the proposed fuzzy logic based method. After that, parsing results of consecutive frames are smoothed by the proposed Markov random field based temporal fusion method. By fusing results of two sensors, it localizes the obstacles correctly and therefore improves the scene parsing result.

experts. Therefore, it can employ not only results from LIDAR and video camera data but also *a priori* knowledge. Second, fuzzy logic can model nonlinear functions of arbitrary complexity. This is important as scene parsing is not a trivial problem. Third, fuzzy logic can tolerate imprecise results of two sensors. Moreover, fuzzy logic is a flexible fusion framework so that results of more sensors can be easily integrated to the system in future.

To fuse results of consecutive frames, we propose a Markov random field (MRF) based temporal fusion method [5,6]. Correspondences between consecutive frames are first estimated by using the dense optical flow method [7]. Then, a MRF model is built to integrate results of multiple consecutive frames. The result of each frame is refined by the Belief Propagation (BP) algorithm [8]. The following contributions have been made in this paper:

1. To the best of our knowledge, the proposed approach is the first systematic fuzzy logic inference based fusion work for scene understanding by fusing results of Velodyne 3D-LIDAR scanner and monocular video camera.
2. The MRF based temporal fusion method is introduced to obtain cohesive video parsing results. It can smooth whole frame simultaneously by integrating results of multiple consecutive frames.
3. We test the proposed approach on datasets collected by our autonomous ground vehicle testbed. The datasets are captured from urban and rural areas either in day or night time. The results validate the robustness and effectiveness of our fusion approach for scene parsing.

A preliminary version of this paper was described in [9]. The current version described here differs from the former in several ways, including: the introduction of MRF based temporal fusion method; comprehensive evaluation of the method with three more datasets; further analysis and discussion of the whole approach, as well as the introduction of more related works about sensor fusion and scene parsing. While the preliminary version in [9] focuses on fuzzy logic based fusion strategies, the current version will provide more details on image parsing techniques, too.

This paper is organized as follows: In Section 2, we briefly survey the sensor fusion and scene parsing literature. After giving the parsing methods for individual sensors, we describe the fuzzy logic based method to fuse the results of two sensors in Section 3.1. The MRF based temporal fusion is presented in Section 3.1. Thorough experiments are conducted in Section 3.1 for evaluation, and in-depth discussion is provided in Section 3.1. We conclude our paper in Section 3.1.

2. Related work

By combining data from multiple sensors, we can achieve improved accuracies and more specific inferences than that achieved by the use of a single sensor alone [10]. The existing methods for fusing LIDAR data and camera images can be grouped into two categories: centralized approaches, decentralized approaches. In centralized approaches, the fusion process occurs at the pixel-level or feature level, i.e., features from both LIDAR and video camera are combined in a single vector for posterior classification. Douillard et al. present a logical rule based object classifier by combining Velodyne data and monocular camera data [11]. A set of twenty-one binary features are defined based on 3D pointclouds and camera images. The logical rules are learned from training data. Häselich et al. present a novel approach for online terrain classification from fused camera and laser range data [12]. Laible et al. propose to handle the terrain classification at different lighting conditions by fusing the camera and LIDAR data [13]. Kaempchen et al. perform centralized free-form object tracking using laser scanner and camera [14]. Schneider et al. address the problems of synchronization, correction and occlusion reasoning for the fusion of camera and LIDAR [15]. Centralized methods can simplify the posterior classification process but are difficult to integrate the human knowledge and experience. Furthermore, in the centralized method, only the regions commonly observed by both sensors can be processed. This greatly limits the area they can cover due to the short range of one sensor.

Decentralized approaches separately classify the data for individual sensor first, the classification results are then combined by a fusion method. Kidono et al. propose a fusion system for reliable pedestrian recognition using Velodyne and a vision sensor to achieve high performance under various conditions [16]. Himmelsbach et al. propose to evaluate the tentacle by fusing LIDAR and camera for autonomous navigation [17]. Labayrade et al. propose a fusion strategy by matching the set of obstacles from laser scanner with the set of obstacles coming from stereo-vision based on the belief theory [18]. Premebida et al. also obtain a better performance than the single classifiers by using the decentralized scheme [19]. Generally, these methods require training data to determine the fusion model and the fusion parameters.

Besides the two fusion strategies, there are works which try to use them together [20–23]. Garcia and Olmeda propose a hybrid fusion strategy by fusing the low and high level information simultaneously [20]. Tang et al. propose to learn the contextual information from input data and then combined with given expert knowledge in classification [21]. Habtemariam et al. propose a multiple detection probabilistic data association (MD-PDA) filter

for tracking a target when more than one target originated measurement may exist within the validation gate [22]. Martin explores another type of fusion by updating the classifications of multiple objects simultaneously when given a measurement on only one of the objects [23]. Matthaei and Dyckmanns use laser and radar to classify motion for cross traffic in urban environments [24].

Scene parsing is one of the fundamental problems of computer vision. Image scene parsing aims to assign a category label to each pixel of a given image. Over the last several years, many methods have been proposed for this problem. They can be broadly categorized on the basis of their basic process units. Several methods are using the pixels as basic units [6], others using segments [25–27], group of segments [28], or intersections of multiple segmentations [29], while the whole image is considered in the extreme case [30]. Several methods are using multiple types of information to improve the parsing results. Tu et al. propose to combine segmentation, detection, and recognition for the scene parsing [31]. Ladický et al. [32] propose an image segmentation and parsing method by combing object recognition, detection and segmentation with a conditional random field defined on pixels, segments and objects. Felzenszwalb and Veksler propose a tiered scene labeling method by using the dynamic programming approach [33].

Other scene parsing methods employ the nonparametric classification method [5,34] or deep feature learning [35]. Liu et al. [5] propose a nonparametric scene parsing method via label transfer algorithm. Tighe and Lazebnik [34] pre-process the video using a spatio-temporal segmentation method that gives 3D regions that are spatially coherent within each frame as well as temporally coherent between frames. Then each 3D region is classified. Farabet et al. propose a scene parsing method by leveraging the deep learning method [35].

Besides the image parsing work, several approaches have tried many strategies to employ the cues contained in video data. Brostow et al. [36], Strurgess et al. [37] and Zhang et al. [38] recover the 3D structure information (e.g., dense depth maps or sparse point clouds) from the video sequences and then combine the 3D information and image information to parse individual frames. Xiao and Quan [39] propose a region-based parsing system on each frame and enforce temporal coherence between regions in adjacent frames by temporal fusion in a batch model.

With the development of range sensors, several recent works obtain the scene semantic labels with the 3D-LIDAR data only. Spinello et al. track the people in 3D pointcloud data using a bottom-up top-down pedestrian detector [40]. Bradley et al. employ the 3D pointcloud to detect vegetation for driving in complex environments [41]. Teichman and Thrun propose a semi-supervised approach to the problem of track classification in dense 3D range data [42]. Behley et al. evaluate several local features for the classification of 3D laser range data in urban environments [43].

3. Parsing modules for individual sensors

As a decentralized fusion method, a geometry segmentation algorithm is proposed to detect obstacles and ground from Velodyne data for this work. In the meantime, one algorithm, which combines both bottom-up and top-down analyses, is designed to classify image patches into multiple categories. In this section, we first describe the two detection algorithms separately and then summarize their advantages and disadvantages.

3.1. Obstacles and ground classification using Velodyne scanner

As mentioned earlier, due to the sparseness of pointcloud, we detect only traversability of the terrain (i.e., classifying the pointcloud into ground and candidate obstacles) from the Velodyne data. To achieve it, we first voxelize the pointcloud $\mathcal{P}$. Then we separate the ground points using a RANSAC plane fitting algorithm [44]. After that, all the above-ground points are obtained and the candidate obstacles are localized by partitioning the above-ground points using 3D adjacency. Fig. 2 illustrates the result of each step. To speed up the process, we first build a 3D cubic voxel grid using the pointcloud $\mathcal{P}$. The pointcloud data are stored in cubic voxels for efficient retrieval and the grid resolution is set to be 0.1 m. By voxelizing, the spatial neighborhood relationships of the 3D points are modeled explicitly.

The second step separates the points into two categories: ground and non-ground. Points are considered in batches, defined by their membership in a single cubic voxel in space. A voxel is considered to contain ground data if the voxel is a member of the lowest (in elevation) set of adjacent non-empty voxels in a vertical column (i.e., not part of an overhang). All 3D points stored in that set of voxels are fitted to a plane using the RANSAC algorithm and the inliers points are the ground points. All inliers points should be near the hypothesis plane (i.e., the distance to the plane is less than 0.3 m). The RANSAC algorithm terminates after testing 100 hypothesis planes. All of the voxels that contain ground points are called ground voxel set $\mathcal{G}$. Other voxels are called the above-ground voxel set $\mathcal{U}$. One above-ground voxel $V_{i,j,k} \in \mathcal{U}$ may contain a number of above-ground points or be an empty voxel, where i, j and k denotes the indexes of the 3D voxel grid.

The third step detects the possible obstacles by clustering the non-empty above-ground voxels according to 3D adjacency [45]. Each obstacle is represented by a voxel cluster. Denote all the voxel clusters as $\mathcal{C}$ and the voxels in one voxel cluster $O \in \mathcal{C}$ should meet the following 3D adjacency criterion:

$$\forall V_{i,j,k} \in O \ \left(\exists V_{i',j',k'} \in O\right) \wedge \left(|i-i'| < d \vee |j-j'| < d \vee |k-k'| < d\right), \tag{1}$$

We set $d = 2$ in current implementation. The detected results are projected to the image as shown in Fig. 2(f). Each bounding box localizes one candidate obstacle. The green circles represent the projection of ground points and the blue circles represent the projection of above-ground points.

3.2. Image parsing module

Contrary to Velodyne information processing which concerns whether the terrain is traversable [46], we intend to identify more specific categories of objects from the images. From the camera images that we have collected, we identified nine possible categories, which include ground (road), building, water, tree, grass and obstacles, etc. As a matter of fact, obstacles can be further divided into human, car, etc. But this detailed division requires sufficient training data for each specific class. Viewing that there are many types of possible known or unknown obstacles, we simply classify all of them into obstacles. For a particular task, specific models can be trained for individual interesting classes too. Table 1 summaries all employed categories. The classification of images is realized by two steps: bottom-up classification of local image patches and top-down contextual analysis to further resolve uncertainties in the bottom-up classification.

During bottom-up classification phase, an image is first over-segmented into small image patches [47]. From each patch, 131 features are extracted, including 24 features from color histograms and 107 features corresponding to different texture descriptors. 36 of them are derived from anisotropic Gauss filtered images, 12 from Gabor filtered images, and 59 Local Binary Patterns [48]. An MLP (multilayer perceptron) classifier is trained to classify the local image patches into object categories [49]. Fig. 3(b) is an

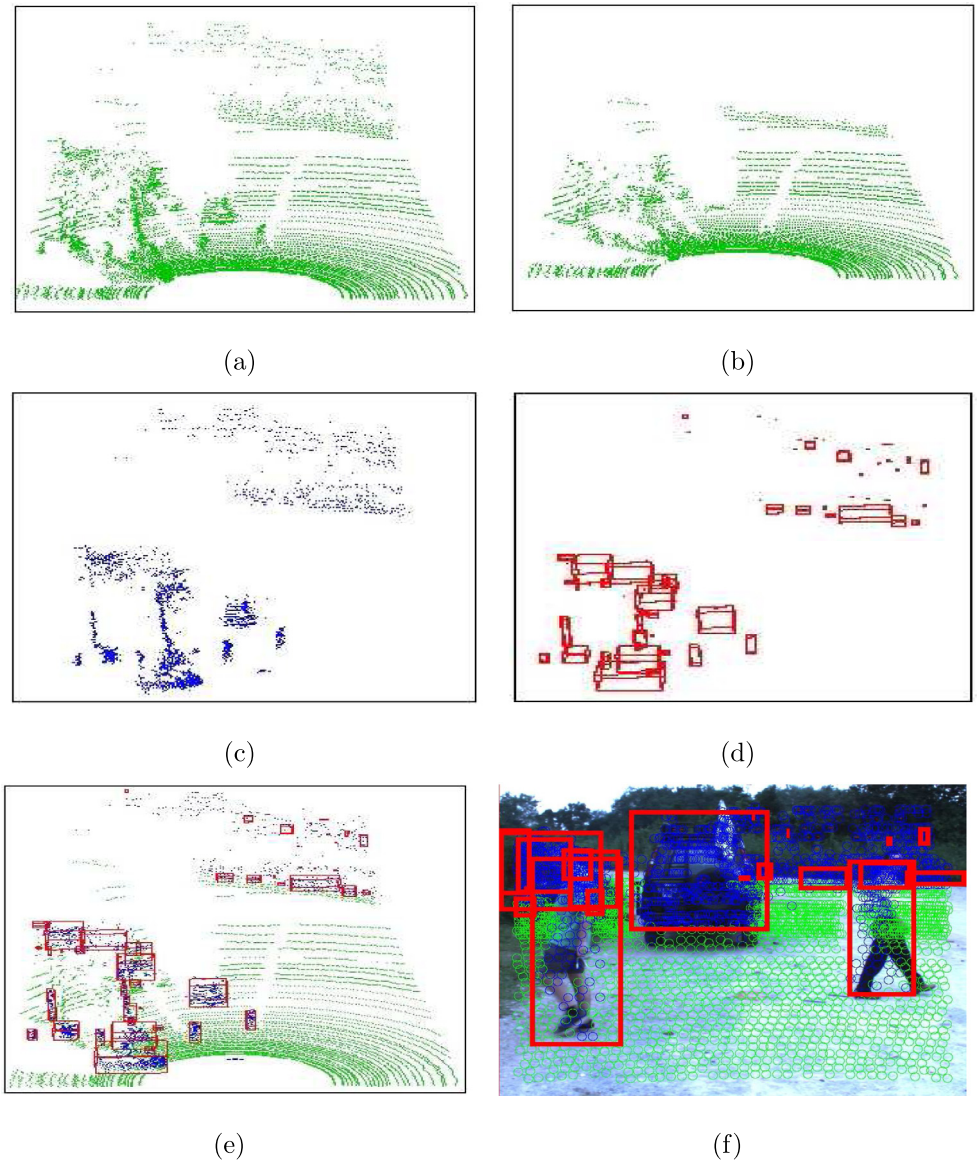


Fig. 2. Illustration of obstacle and ground classification using Velodyne scanner. (a) is the 3D pointcloud of Velodyne scanner; (b) is the ground points; (c) is the above-ground points; (d) shows the detected bounding boxes of the candidate obstacles; (e) shows the pointcloud and the detected results; (f) shows the detected results which are projected to the camera image. Each bounding box represents one candidate obstacle in (d)–(f).

Table 1
The category summary.

Category	Explanation
Ground	Traversable ground plane
Obstacle	Pedestrian, vehicle and other objects above ground
Building	Human-made structure
Grass	Vegetation with height less than 0.3 m
Bush	Vegetation with height between 0.3 m and 2.0 m
Tree	Vegetation with height more than 2.0 m
Pavement	/
Sky	/
Water	/

example of bottom-up classification result, where patches of original image in Fig. 3(a) are classified into different categories.

Sometimes, errors will occur in the bottom-up classification. For instance, in Fig. 3(b), some image patches of "sky" (area A) are wrongly classified into "ground", some part of "tree" (area B) is classified into "water", and a part of "grass" (area C) is classified as "tree". Some errors in bottom-up classification can be further corrected by a top-down contextual analysis process. This is because only local features of the image patches are considered during the bottom-up classification phase. It is possible that local patches of different object categories may look similar, leading to uncertainties in the bottom-up classification. However, when looking at an image patch from its surrounding context, e.g., the categories of its neighbors, the uncertainty can be resolved. For example, "ground" cannot be above "tree" in the image if it is taken from a moving vehicle. This property has been well recognized and employed in several computer vision systems [50]. However, most of them either treat contextual information equally with local, low-level features or mix the contextual information with low-level features in one classifier. Our work is different from them in that we model the contextual relations independent of the bottom-up classification process, allowing the contextual analysis result to feedback to the bottom-up classification module so as to update the final classification result.

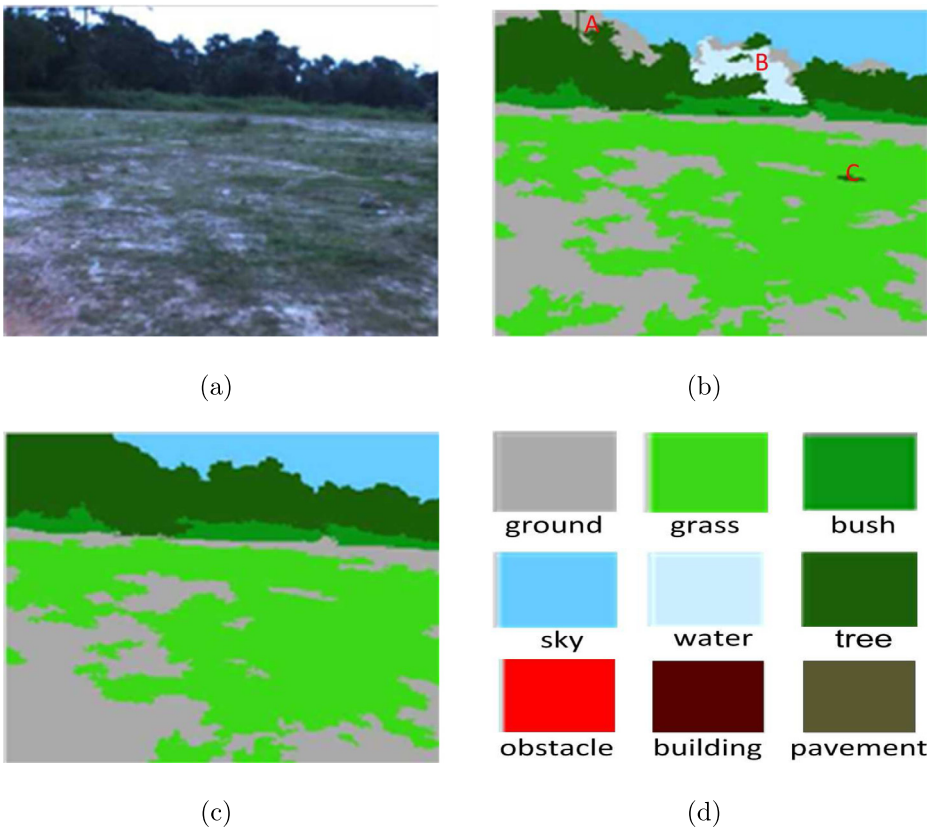


Fig. 3. Illustration of image classification. (a) is the original image; (b) is the classification result of bottom-up phase; (c) is the final classification result after top-down contextual analysis; (d) shows the color of each category. The classification errors in area A–C in (b) are corrected after top-down contextual analysis. (For interpretation of the references to colour in this figure caption, the reader is referred to the web version of this article.)

To acquire the top-down contextual relation module, the connected image patches classified into the sample category by the bottom-up classification process are first grouped into bigger components, where each component corresponds to a connected area. Then, the existence of neighboring categories of a component is derived from three directions: top, down and sides (both left and right sides). For each direction, we check the existence of each category, as well as whether the component is adjacent to the boundary of the image. They form the contextual information of the component. This contextual information is then passed to the Bayesian network as evidence. The probability of the node category will be updated through Bayesian network inference.

Denote the number of categories as L, the number of nodes in the Bayesian network is $3(L+1)+1$. The coefficient 3 means that three directions are considered: top, bottom and sides. $L+1$ means for each direction, we check the existence of each category, as well as whether the component is adjacent to the boundary of the image. Fig. 4 illustrates the corresponding Bayesian network with four categories "Tree", "Road", "Obstacle" and "Image" (i.e., pseudo category for image boundary) for drawing convenience. The root node "Class" corresponds to the component under consideration. Denote the category probability of this component as $P(\text{Comp\_Class})$ and its initial value is acquired based on bottom-up classification result as described above. The probability of this node will be updated based on the evidence from other nodes, which correspond to different contextual information respectively. For example, the node "Top-Tree" corresponds to whether the top neighbor of the component is "Tree" while the 'Side-Image" corresponds to whether the side neighbor of the component is "Image" (i.e., image boundary). The model was learned using the TAN (tree augmented Naive Bayes) training algorithm [51]. Therefore, for each node, except the root node, there will be at most two parents. The training data comes from the combination of bottom-up initial classification module and manually labeled ground truth of the training images.

With the learned model, the probability of the root node $P(\text{Comp\_Class})$ is updated through Bayesian network inference. This updated probability is fused with the bottom-up classification confidence via multiplication:

$$\begin{aligned} &P(\text{Comp\_Class}|\text{Side\_Image},\ldots,\text{Bottom\_Obstacle}) \\ &\quad = P(\text{Comp\_Class})P(\text{Side\_Image}|\text{Comp\_Class}),\ldots, \\ &\quad P(\text{Bottom\_Obstacle}|\text{Comp\_Class}). \end{aligned} \tag{2}$$

As shown in Fig. 3(c), the classification errors in area A–C are corrected after contextual analysis.

3.3. Summarization of two methods

By analysing the results, it can be seen that both methods have their own advantages and disadvantages. The laser scanner based method can separate the ground and above-ground points robustly. It can also segment the obstacles if they are not adjacent to other obstacles. However, the laser scanner can only obtain a sparse pointcloud and it has no information about water, sky and the areas out of the sensor's range, as shown in Fig. 5. Besides, the detected obstacles include many tree and bush areas, which will increase the possibility of the vehicle deviating from the road region. As for the camera based method, it can classify the image into multiple categories. However, due to the diversity of the

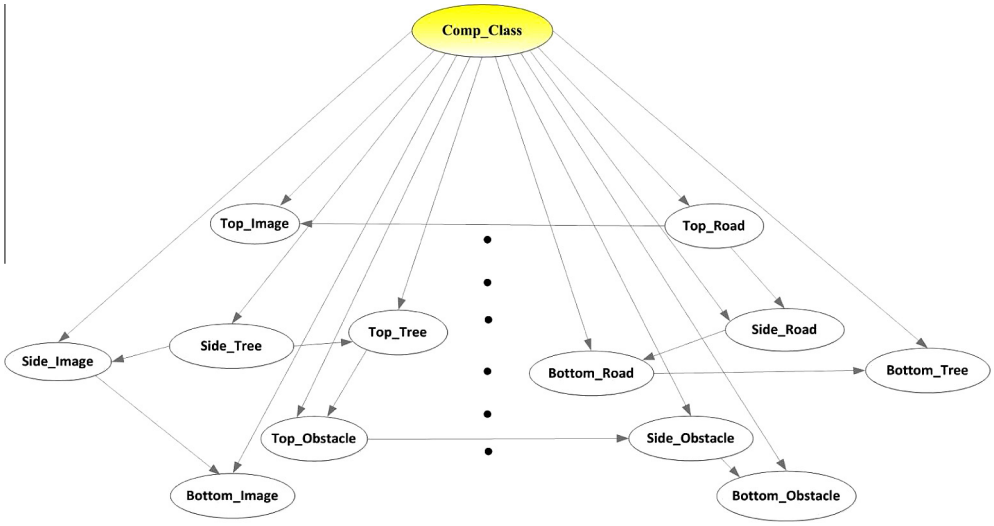
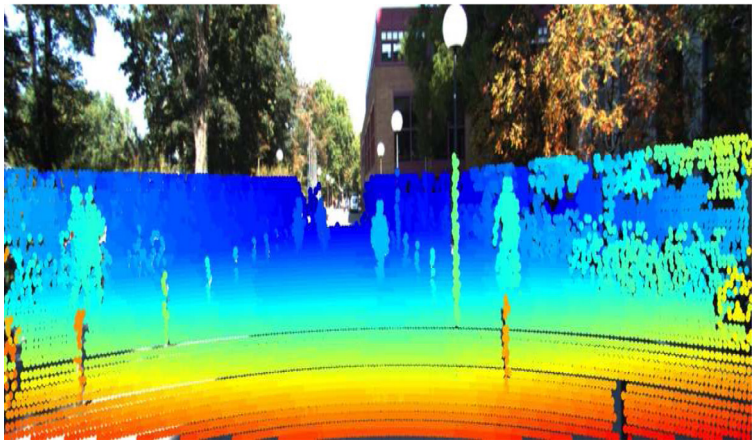


Fig. 4. Illustration of the Bayesian networks for contextual analysis. Here we show four categories "Tree", "Road", "Obstacle" and "Image" (i.e., pseudo category for image boundary) for drawing convenience. For each category, we consider three diresctions: Top, Bottom and Side (both left and right sides). The root node "Comp_Class" corresponds to the category of the component under consideration. As learned by using the TAN (tree augmented Naive Bayes) algorithm [51], for each node, except the root node, there will be at most two parents.

(a)



(b)

Fig. 5. Illustration of the range covered by Velodyne. (a) shows one camera image. (b) shows the projected Velodyne points. It can be seen that only nearby areas have the corresponding Velodyne points.

obstacles, some obstacle regions may be classified as wrong categories. The complementary performance of two methods shows the possibility to boost the scene parsing and obstacle detection by combining them.

4. Fuzzy logic based sensor fusion

Both the results of laser scanner and the results of camera image have their own advantages and disadvantages. To parse the scene correctly, the primary work of fusion is to categorize the detected candidate obstacles by Velodyne scanner. The scene parsing results are then improved based on the categorization. As a good way to utilize the *a priori* knowledge and experience of human experts [4], we propose to use the fuzzy inference method to fuse the results of two sensors.

4.1. Fuzzification of the fusion

The inputs to the fuzzy fusion module are five related attributes of each candidate obstacle: the size of candidate obstacle (*size*), the image classification result (*class*), the spatial context (*s-context*), the temporal context (*t-context*) and the absolute height (*height*) of the candidate obstacles. The output result classification (r_c) is the detection result for the candidate obstacles. Each input and output parameter is defined as a fuzzy variable.

To employ the *a priori* knowledge, all the associated fuzzy variables are first fuzzified into linguistic labels. The input variable *size* is simply expressed using four linguistic labels TIN (tiny), SMA (small), MID (middle) and LAR (large) within the universe of discourse (0,100) percents. The candidate obstacle size is defined as the percents of all image pixels which are inside the candidate obstacle bounding box. The variable *class* is expressed using three linguistic labels NOBS (non-obstacle), MID (middle) and OBS (obstacle) within the universe of discourse (0,100) percents. The classification is measured by the percent of non-obstacle pixels among all the pixels inside the candidate obstacle bounding box. All the detected grass, bush, tree and building pixels by image classification method are considered as non-obstacle pixels. When most of inside pixels belong to the non-obstacle category, the candidate obstacle is probably not the pedestrian and vehicle, and vice versa. We count the number of non-obstacle pixels to describe the bounding boxes as the tree, bush or building areas are possible to be detected as candidate obstacles while they might be far from the traversable ground area. Therefore, by removing the tree, bush or building from the candidate obstacles, the autonomous vehicle will focus on the obstacles which are above on the traversable ground.

The spatial context *s-context* is expressed using two linguistic labels NOBS (non-obstacle) and OBS (obstacle) within the range (0,8). It is obtained from the classification results of eight pixels around the candidate obstacle bounding box. Four of them are the corners of the box and the other four are the middle point of each edge of the bounding box. If one pixel is classified as ground, *s-context* is added by one. The temporal context is expressed using two linguistic labels LOW (low) and HIG (high) within the range (0,1). The temporal context describes the existence possibility of current obstacle in the previous frame. By checking the neighborhood of current position in the previous frame, if there is one obstacle with similar size and classification as current one, the temporal context is HIG. Otherwise, the temporal context is LOW. The *height* is the absolute height of the candidate obstacle obtained by the scanner directly. It is expressed using three linguistic labels LOW (low), MID (middle) and HIG (high) within the range (0,10) m. If the obstacle is very high (i.e., >4 m), it is more likely to be a tree rather than a car. It is important to note that the flat-world assumption is used here to make the absolute height work.

The output result score (r_c) is simply expressed using three linguistic labels NOBS (non-obstacle), MID (middle) and OBS (obstacle) within the universe of discourse (0,1). All the membership functions of input and output variables are illustrated in Fig. 6.

4.2. Knowledge rules of scene classification

Based on the human knowledge and experience, a vehicle is required to move on the ground and avoid all the obstacles simultaneously. To detect the categorization of each candidate obstacle, both the detection results of scanner and the camera are used. Besides, the spatial and temporal context of the obstacle is also important knowledge. When the candidate obstacle is surrounded by ground region, it is probably an obstacle. However, when the candidate obstacle is on the edge of ground region, its categorization highly depends on image classification result and other information like height of the obstacle. By analyzing the application scenario of our auto-driving vehicle, the following rules are selected.

The group of rules when the size of object box is large:

R_1: if *size* is LAR and *class* is OBS then r_c is OBS;
R_2: if *size* is LAR and *class* is MID then r_c is MID;
R_3: if *size* is LAR and *class* is NOBS then r_c is NOBS;
R_4: if *size* is LAR and *class* is NOBS and *s-context* is NOBS then r_c is NOBS;
R_5: if *size* is LAR and *class* is NOBS and *t-context* is NOBS then r_c is NOBS;
R_6: if *size* is LAR and *class* is NOBS and *s-context* is OBS then r_c is OBS;

The italic assertion in R_1–R_6 is the condition part of each rule, which is contributed by the detection result of two sensors. These rules indicate that the size of the obstacle is not the only criterion to decide categorization of the obstacle boxes. The image classification result and the context information are also very important for scene classification.

When the size of obstacle is becoming smaller and smaller, the image classification result and the context information will play a more important role for scene classification:

R_7: if *size* is MID and *class* is OBS then r_c is OBS;
R_8: if *size* is MID and *class* is MID then r_c is MID;
R_9: if *size* is MID and *class* is NOBS then r_c is MID;
R_{10}: if *size* is MID and *class* is MID and *s-context* is NOBS then r_c is NOBS;
R_{11}: if *size* is MID and *class* is NOBS and *s-context* is NOBS then r_c is NOBS;
R_{12}: if *size* is MID and *class* is NOBS and *t-context* is NOBS then r_c is NOBS;
R_{13}: if *size* is MID and *class* is NOBS and *height* is MID then r_c is NOBS;
R_{14}: if *size* is SMA and *class* is OBS then r_c is MID;
R_{15}: if *size* is SMA and *class* is OBS and *s-context* is OBS then r_c is OBS;
R_{16}: if *size* is SMA and *class* is OBS and *s-context* is NOBS then r_c is NOBS;
R_{17}: if *size* is TIN then r_c is MID;

The absolute height of one candidate obstacle is also an important criterion to decide the result. If the obstacle's height is very large (e.g., higher than 4 m), the obstacle is more likely a tree rather than a car. The height attribute is included in the following rules:

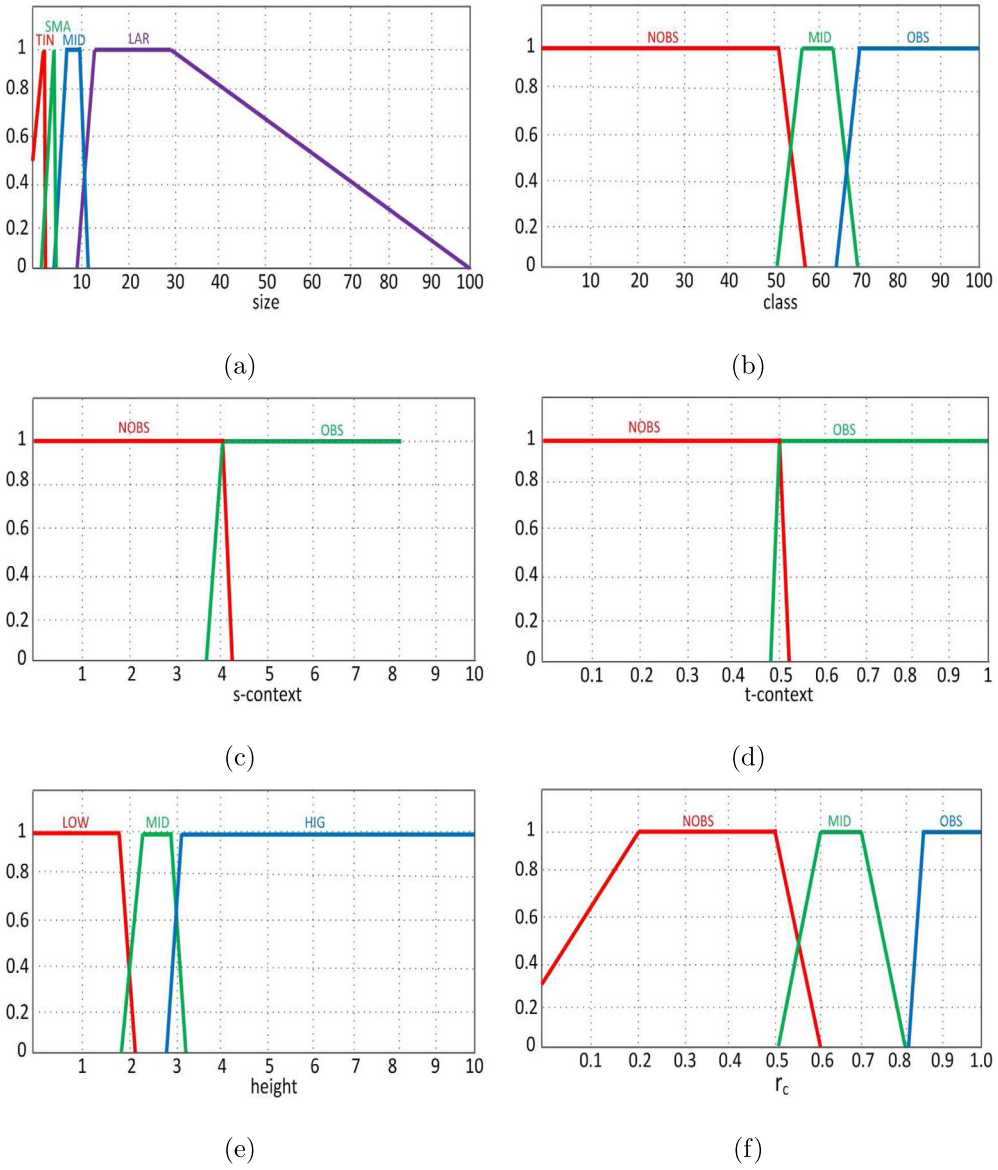


Fig. 6. Illustration of membership function for input and output fuzzy variables. (a) shows membership function of *size*; (b) shows membership function of *class*; (c) shows membership function of *s-context*; (d) shows membership function of *t-context*; (e) shows membership function of *height*; (f) shows membership function of r_c.

R_{18}: if *class* is NOBS and *height* is MID then r_c is NOBS;
R_{19}: if *class* is MID and *height* is MID then r_c is NOBS;
R_{20}: if *height* is HIG then r_c is NOBS;

Although 20 rules do not cover the complete relationships of different attributes, these rules help to integrate the results of two sensors and the human knowledge and experience.

4.3. Fuzzy reasoning

After synthesizing these 20 rules for fusion, their roles are further coordinated through Mamdani's fuzzy reasoning method in this section [52]. The process of Mamdani fuzzy inference involves steps fuzzification, inference, aggregation and defuzzification. The information flow of the fuzzy reasoning is shown in Fig. 7.

Fuzzification converts the input values into a degree via membership functions. The input is always a crisp numerical value and the output is a fuzzy degree of membership in the qualifying linguistic set. The membership functions are illustrated in Fig. 6. After the inputs are fuzzified, the inference of a rule uses the minimal operation to combine different condition assertions for logical operator *and* and generate the output grade for the conclusion assertion. Taking rule R_7 as an example, given a set of inputs size $size^*$ and $class^*$, the output grade r_s^* of the label OBS due to this rule can be inferred as:

$$U_{OBS}^{7}(r_s^*) = \min(U_{MID}(size^*), U_{OBS}(class^*)), \tag{3}$$

where $U_{MID}(size^*)$ and $U_{OBS}(class^*)$ represent the membership functions of the corresponding labels.

There are two steps involved in the aggregation process: the maximum operation of the output grades of each output label due to several rules, and the generation of the output membership function. The aggregated output grade belonging to one corresponding label (such as label OBS) is calculated as:

$$U_{OBS}(r_s^*) = \max(U_{OBS}^{1}(r_s^*), U_{OBS}^{2}(r_s^*), \ldots, U_{OBS}^{20}(r_s^*)). \tag{4}$$

The aggregated output membership function $U_O(r_s)$ is obtained by cutting the membership function $U_{OBS}(r_s), U_{MID}(r_s)$ and $U_{NOBS}(r_s)$

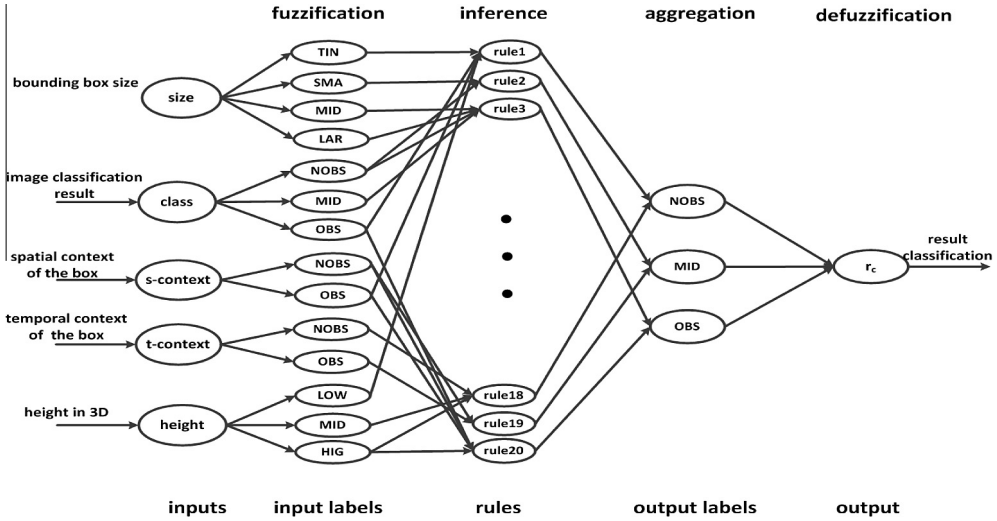


Fig. 7. Information flow of fuzzy reasoning for scene parsing.

respectively at the grades $U_{OBS}(r_s)^*$, $U_{MID}(r_s)^*$ and $U_{NOBS}(r_s)^*$, and combing them point by point:

$$U_O(r_s) = \max(\min(U_{OBS}(r_s^*), U_{OBS}(r_s)), \min(U_{MID}(r_s^*), U_{MID}(r_s))), \\ \min(U_{NOBS}(r_s^*), U_{NOBS}(r_s))). \quad (5)$$

After aggregation, the input for the defuzzification process is a fuzzy set and the output is a single number. The defuzzification process finds the center of gravity of the output membership function as the real value of the output variable:

$$r_s^* = \frac{\int U_O(r_s) r_s \, dr_s}{\int U_O(r_s) \, dr_s}, \quad (6)$$

r_s^* is the final crisp classification score for the candidate box. Based on the classification score, the categories of the candidate obstacles are decided. If the result score of one candidate obstacle is large enough (i.e., $r_c > 0.65$), it is classified as the obstacle. Otherwise, its result depends on the image classification method. After that, we update the categories of the patches inside the obstacle bounding boxes by considering the results of two sensors.

4.4. Automatic setting the fuzzy logic inference

To fuse the results of two sensors, the fuzzy logic is employed by defining the fuzzy variables and fuzzy rules. Although the fuzzy rules and fuzzy variables are decided manually by analyzing the application scenarios in our implementation, the neuro-fuzzy approach can select the rules and tune the parameters automatically [53]. Neuro-fuzzy approach combines the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks. Through the neuro-fuzzy approach, the proposed fuzzy logic based fusion method is easy to be applied in many different applications.

5. Temporal fusion of consecutive frames

By fusing the results of camera and scanner, we can have a better parsing result of each frame. The image parsing result is helpful to understand the environment for the ground vehicle. However, the results of consecutive frames may have abrupt changes due to the car moving, partial occlusion, etc. Fig. 8 shows this phenomenon and several incohesive regions are marked in one frame by the white circles. These abrupt changes of parsing results will mislead the vehicle navigation system. One major reason of cohesive problem is that the temporal information is not included in the scene parsing process.

There are several challenges to do temporal fusion for video scene parsing. First, the whole frame should be considered simultaneously to obtain the spatial coherence for all pixels. Second, to do the temporal fusion, the pixels should be matched densely between consecutive frames. Therefore, we can not use the sparse feature matching algorithm as hundreds of visual features are not enough to cover the whole frame. Moreover, one single previous frame may not have the enough information for temporal fusion and multiple previous frames are required.

As we do not assume the type of dynamical system and the probability distribution of errors and measurements, we prefer not to use the popular filter algorithm like Kalman filter. Instead, to address the above problems for temporal fusion, we model each frame as a Markov random field (MRF) [5,6]. The correspondences between two consecutive frames are first estimated by using the dense optical flow method [7]. Then each frame is represented by a MRF model to integrate results of multiple previous frames. After that result of each frame is refined by solving the MRF model using Belief Propagation (BP) algorithm [8]. Fig. 9 describes the MRF model.

For a given frame I_t, we consider its k previous frames for temporal fusion. Each previous frame I_i is described by its initial scene parsing result c_i and the optical flow field v_i. The k previous frames are denoted by the set $\{c_i, v_i\}_{t-k \leqslant i \leqslant t-1}$. For previous frame I_i, $c_i(p) \in [0,1]^L$ is the category probability vector for pixel p obtained by image parsing algorithm, i.e., $|c_i(p)| = 1$. Here we assume there are a total of L categories as defined in Table 1. $c_t(p)^l$ represents the probability of pixel p is classified to category l. v_i is the optical flow field (from I_t to I_i). We want to obtain the smoothed parsing result c_t^s for given frame I_t by fusing result of frame I_t and that of k previous frames. Therefore we build a probabilistic Markov random field model to integrate results of multiple frames and impose a spatial smoothness constraint. Inspired by Liu et al. [5] and Shotton et al. [6], the posterior probability is defined as:

$$-\log P(c_t^s | c_t, \{c_i, v_i\}) = \overbrace{\sum_p \phi_1(c_t^s(p); c_t)}^{\text{current frame cohesive}} + \overbrace{\sum_p \phi_2(c_t^s(p); c_{t-1}, v_{t-1})}^{\text{previous frame cohesive}} \\ + \overbrace{\sum_p \phi_3(c_t^s(p); \{c_i, v_i\})}^{\text{historical prior cohesive}} + \overbrace{\sum_{\{p,q\} \in \epsilon} \phi_4(c_t^s(p), c_t^s(q); c_t)}^{\text{spatial cohesive}} \\ + \overbrace{\log \mathcal{Z}}^{\text{normalization constant}}, \quad (7)$$

where $\mathcal{Z}$ is the normalization constant of the probability and ϵ is the set which represents the neighborhood relation of all pixels in

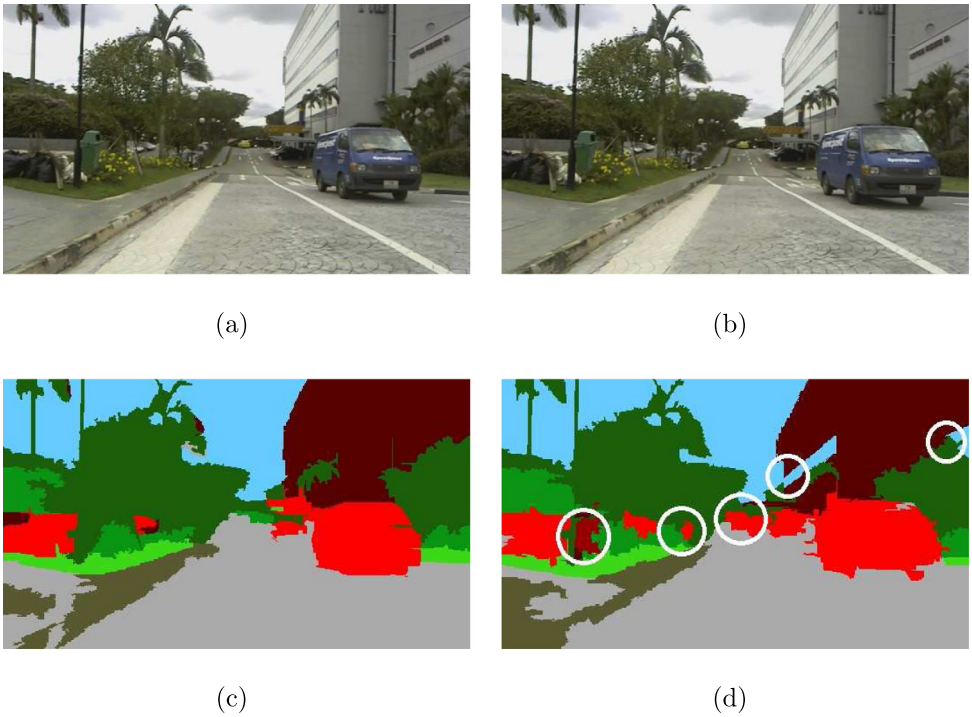


Fig. 8. The cohesive problem between consecutive frames. (a) and (b) are two consecutive frames while (c) and (d) are the corresponding image parsing results. The white circles in (d) illustrate several places which do not have cohesive parsing results between two frames.

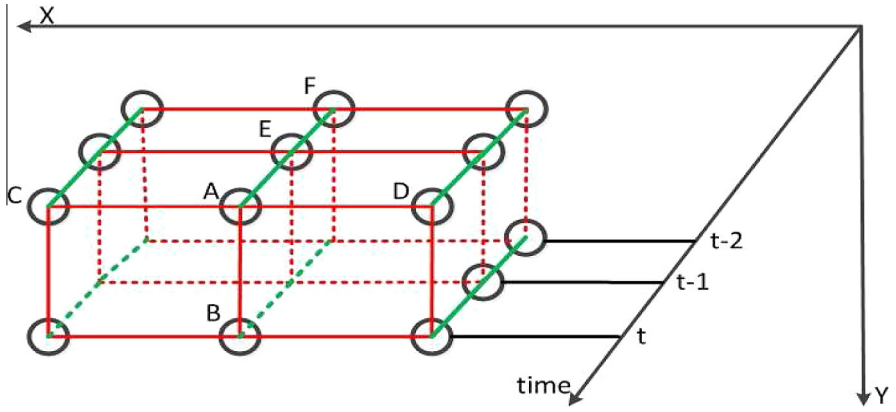


Fig. 9. Illustration of the MRF model for frame t. The frame t, $t-1$ and $t-2$ are shown and each black circle represents one pixel. The red edges show the neighborhood relation in the same frame while the green edges show the corresponding relation between consecutive frames. The smoothed parsing result of pixel A is decided by considering both its neighbor pixels in the same frame (e.g., B, C and D) and its corresponding pixels in previous frames (e.g., E in frame $t-1$ and F in frame $t-2$). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

frame I_t. p and q are pixels in frame I_t. Among four components of this posterior, ϕ_1 ensures the smoothed result similar to parsing result of current frame c_t while ϕ_2 forces the smoothed result close to parsing result of previous frame c_{t-1}. ϕ_3 depends on parsing results of corresponding pixels in multiple previous frames $\{c_i, v_i\}_{t-k\leqslant i\leqslant t-1}$ and ϕ_4 incorporates a spatial smoothness constraint which depends on smooth parsing result of current frame c_t^s. Optical flow field $\{v_i\}_{t-k\leqslant i\leqslant t-1}$ is used to find corresponding points between current frame and previous frames.

The first term ϕ_1 is defined as:

$$\phi_1(c_t^s(p)=l)=(1-c_t(p)^l), \tag{8}$$

where $c_t(p)^l$ represents the probability of pixel p is labeled as category l in image parsing result. The higher the probability $c_t(p)^l$, the more chance smoothed parsing result of pixel p is set to be l. The second term ϕ_2 is defined as:

$$\phi_2(c_t^s(p)=l)=\begin{cases}\|I_t(p)-I_{(t-1)}(p_{(t-1)})\|(1-c_{t-1}(p)^l) & \text{if } \exists\ p_{(t-1)},\\ \tau & \text{else},\end{cases} \tag{9}$$

where $p_{(t-1)}=p+v_{t-1}(p)$ is p's corresponding pixel in previous frame I_{t-1}. τ is set to be the maximum intensity difference value between corresponding pixels of two frames $\tau=\max_{p,p(t-1)}\|I_t(p)-I_{(t-1)}(p_{(t-1)})\|$.

The term ϕ_3 incorporates the probability that category l appears at pixel p's corresponding pixels in several previous frames. ϕ_3 is considered as the historical prior for category l and its value is obtained from counting the occurrence of category l at pixel p's corresponding pixels in k previous frames:

Table 2
The information of five datasets.

Dataset	Frame No.	Label level
Dataset 1	440	Pixel's category
Dataset 2	450	Pixel's category
Dataset 3	1500	Pixel's category
Dataset 4	1500	Pixel's category
Dataset 5 [54]	150	Pedestrian's bounding box

$$\phi_3(c_t^s(p) = l) = -\log(N_l + 1), \tag{10}$$

where N_l is occurrence number of category l in p's corresponding pixels in k previous frames. The smoothness term ϕ_4 compels neighboring pixels to have the same label in the event that no other information is available and its value depends on parsing result of current frame c_t:

$$\phi_4(c_t^s(p), c_t^s(q)) = \delta[c_t^s(p) \neq c_t^s(q)] \| c_t(p)^{l_*} - c_t(q)^{l_*} \|, \tag{11}$$

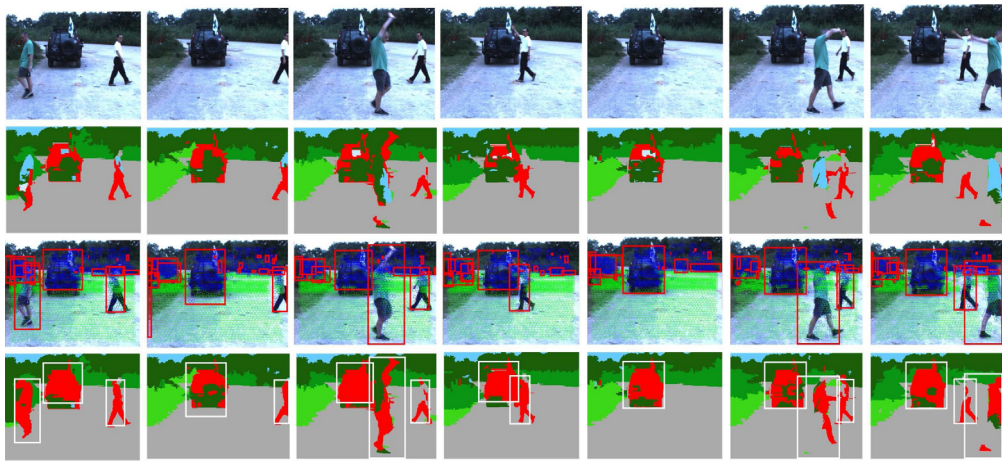
where $c_t(p)^{l_*}$ represents the maximum probability value in category probability vector of pixel p. To compel neighboring pixels have the same label, $\delta[c_t^s(p) \neq c_t^s(q)]$ is set to be 1 when $c_t^s(p) \neq c_t^s(q)$ and it is set to be 0 when $c_t^s(p) = c_t^s(q)$. ϕ_4 can add a penalty if two neighboring pixels have different smoothed labels. Once these energy functions are calculated for frame I_t, we use BP algorithm to minimize the energy [8] and the parsing result is smoothed consequently.

6. Performance evaluation

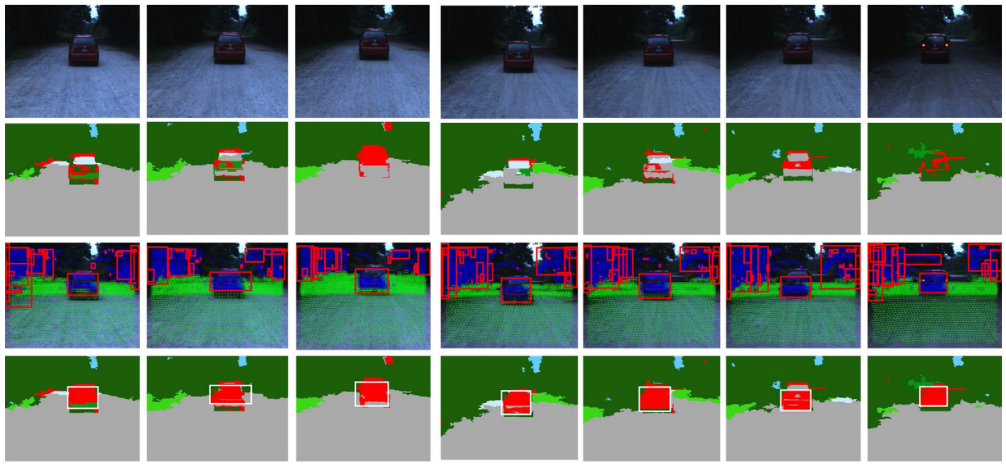
To evaluate our fusion approach, we test it on four datasets collected by our autonomous ground vehicle testbed when driving in rural and urban areas and one public pedestrian dataset [54]. In the experiments, we compare the fusion result with that of using video camera only. In addition, the MRF based temporal fusion method is further evaluated.

6.1. Dataset and sensor calibration

The datasets are collected by an autonomous ground vehicle testbed while the vehicle is outfitted with a Velodyne 3D-LIDAR scanner, a monocular camera and other sensors. The calibration



(a)



(b)

Fig. 10. Sample results of scene parsing and obstacle detection using Dataset 1 and Dataset 2. Panel (a) presents the result of Dataset 1 while panel (b) presents the result of Dataset 2. In each panel, the first row shows the original camera image and the second row shows the image classification result. The red color illustrates the region of detected obstacles and other colors have similar meaning as in Fig. 3(d). The third row shows the detected result using scanner pointcloud and the results are projected to the camera image. Each bounding box localizes one candidate obstacle. The fourth row shows the result of our fusion method. Each white bounding box localizes one detected obstacle. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

of camera and Velodyne is done in a coarse-to-fine manner. We first estimate extrinsic parameters of camera using Caltech calibration toolbox [55]. Then we initially estimate the tilt, roll and yaw of camera with regard to the world coordinate system based on the estimated vanishing line on selected images. In this step, we assume the Velodyne coordinate system is the world coordinate system as the calibration is done when the vehicle is static. After obtaining the initial result, we fine-tune the parameters based on the mapping result between Velodyne points and image pixels.

Table 2 summaries all five datasets. The first dataset corresponds to an open ground in rural area while the second one corresponds to the road in rural area. The first dataset consists about 440 frames and the second one consists about 450 frames. The third and fourth datasets correspond to the road in urban area while they both have about 1500 frames. The first three datasets are collected in the day time while the fourth one is collected in the night time. Besides these four self-collected datasets, we select the pedestrian data from the recent public dataset [54] as our fifth dataset. Two challenging video sequences are selecetd: "2011-09-28-drive-0038" and

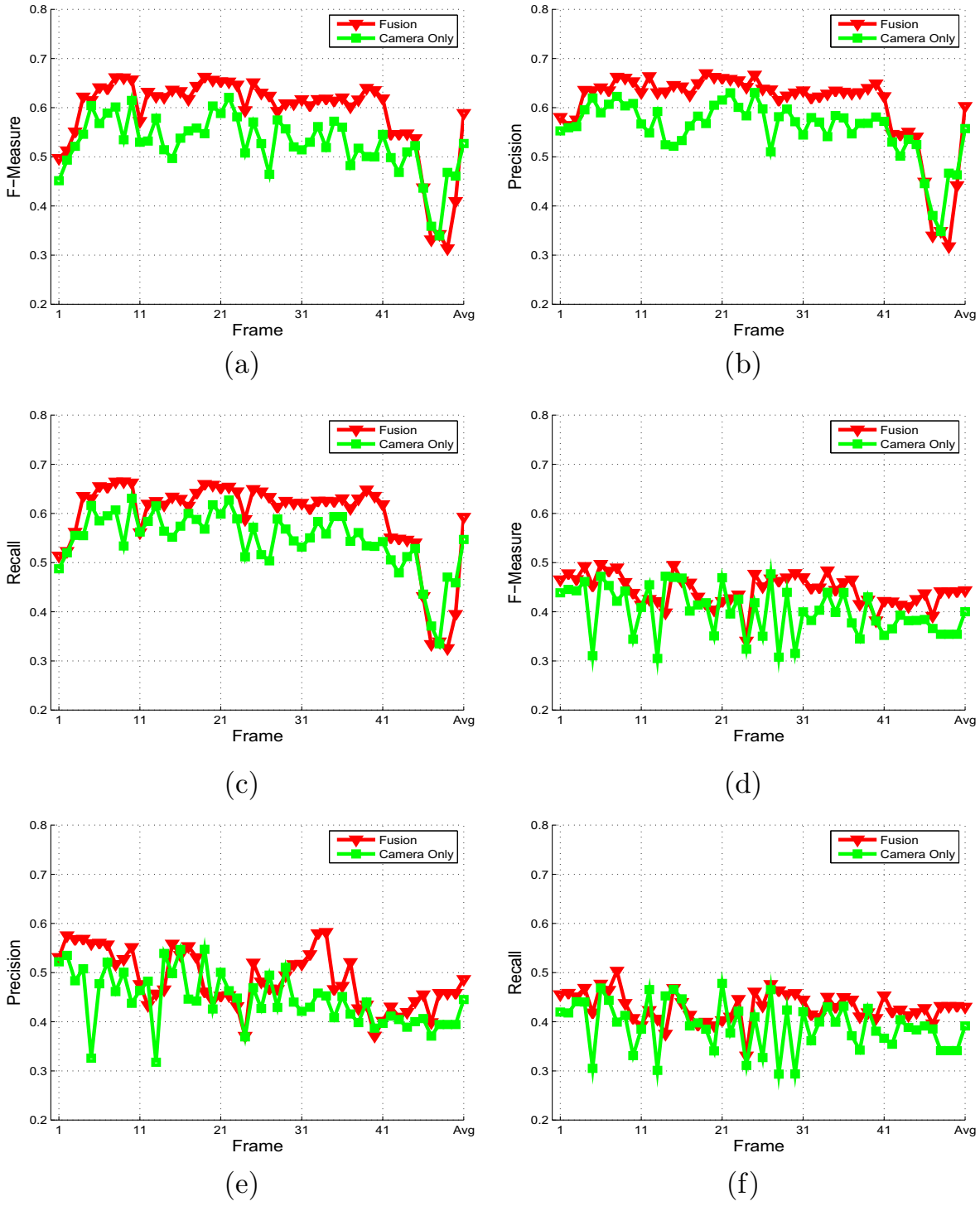


Fig. 11. The scene parsing performance of the proposed fusion based scene parsing method (fusion) and the image parsing method (camera only) using Dataset 1 and Dataset 2. The results of all categorizes are averaged to obtain the *F-measure*, *precision* and *recall* values for each frame. (a) shows the *F-measure* value of Dataset 1; (b) shows *precision* value of Dataset 1; (c) shows *recall* value of Dataset 1; (d) shows the *F-measure* value of Dataset 2; (e) shows *precision* value of Dataset 2; (f) shows *recall* value of Dataset 2.

"2011-09-28-drive-0045". To quantify the performance of the proposed approach, we manually labeled about 20% of all frames in the first and second datasets, and 5% of all frames in the third and fourth datasets. There are total 9 labeled categorizes which include road, obstacle, building, tree, sky, water, etc. For the fifth dataset, we use the groundtruth of pedestrians provided by Geiger et al. [54]. Dataset 1, Dataset 2 and Dataset 5 contain data of both camera and Velodyne while Dataset 3 and Dataset 4 contain data of camera.

The proposed scene parsing approach aims to provide the environment situation awareness ability for autonomous ground vehicles. We evaluate our fusion method on these collected datasets. The $k = 4$ previous frames are considered in the temporal fusion step and ϵ contains eight neighbors of each pixel in order to obtain the spatial smooth. For each category, the set of pixels which are classified to this category by our method is denoted as *DR* (i.e., detect region). The set of pixels which are manually labeled to this category is denoted as *GT* (i.e., ground truth). The performance is measured by

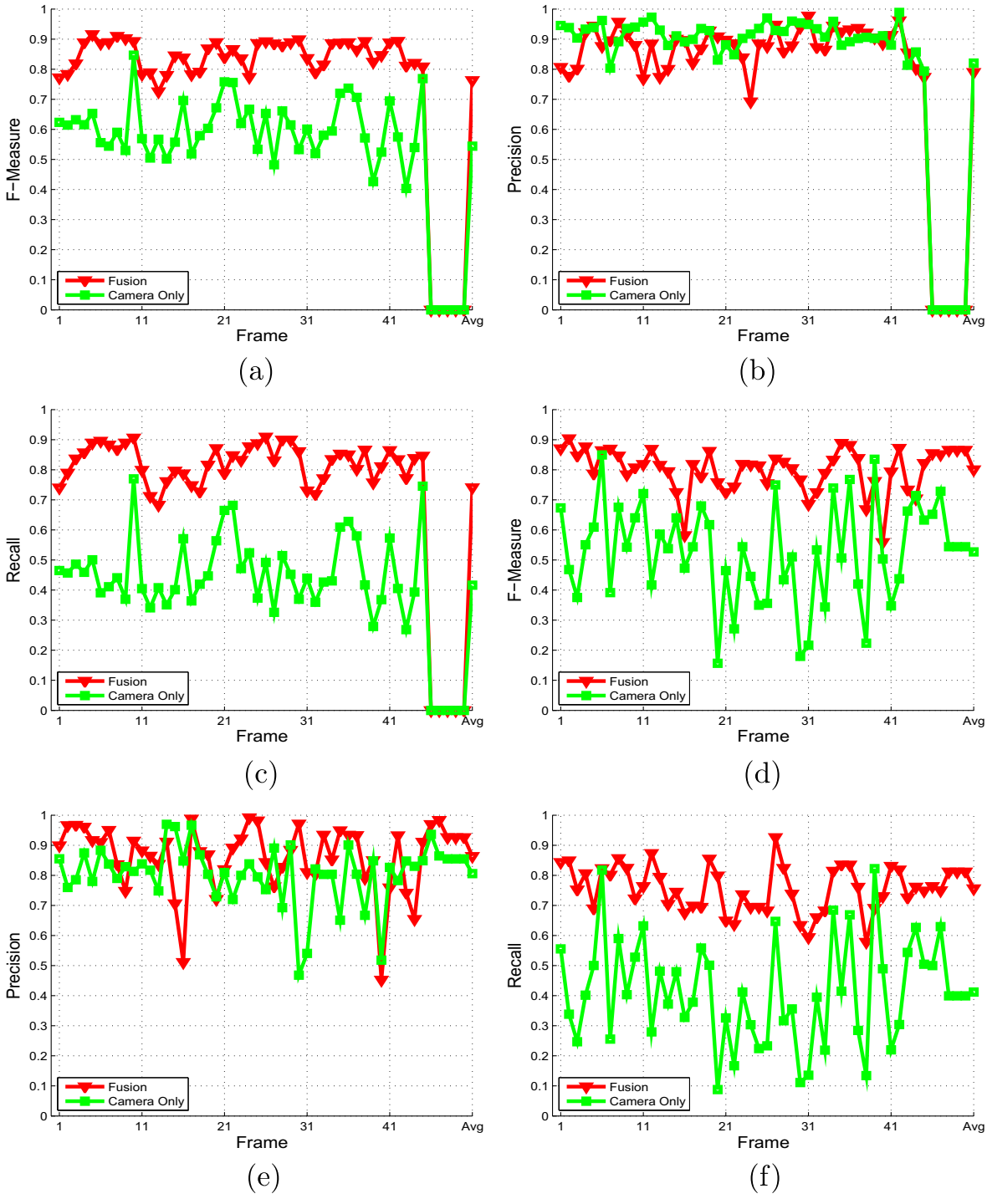


Fig. 12. The obstacle detection performance of the proposed fusion based scene parsing method (fusion) and the image parsing method (camera only) using Dataset 1 and Dataset 2. (a) shows the *F-measure* value of Dataset 1; (b) shows *precision* value of Dataset 1; (c) shows *recall* value of Dataset 1; (d) shows the *F-measure* value of Dataset 2; (e) shows *precision* value of Dataset 2; (f) shows *recall* value of Dataset 2.

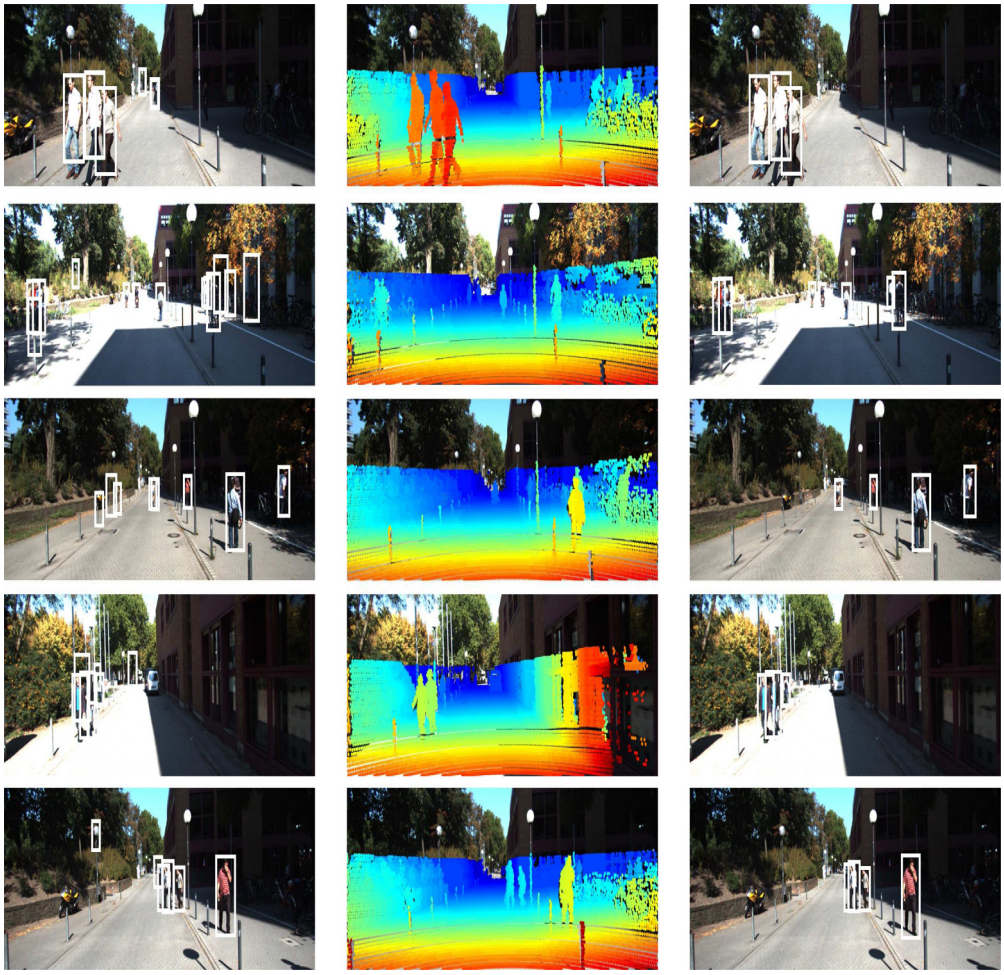


Fig. 13. Sample results of pedestrian detection using Dataset 5 [54]. Panel (a) presents the result of pedestrian detection using the image only; panel (b) shows the projected Velodyne points; panel (c) shows the pedestrian detection result using the proposed sensor fusion method. Each bounding box represents one detected pedestrian. By fusing of camera and Velodyne, we can remove many false detections. Meanwhile there is clearly room for improvement. For example, detect pedestrians in shadow.

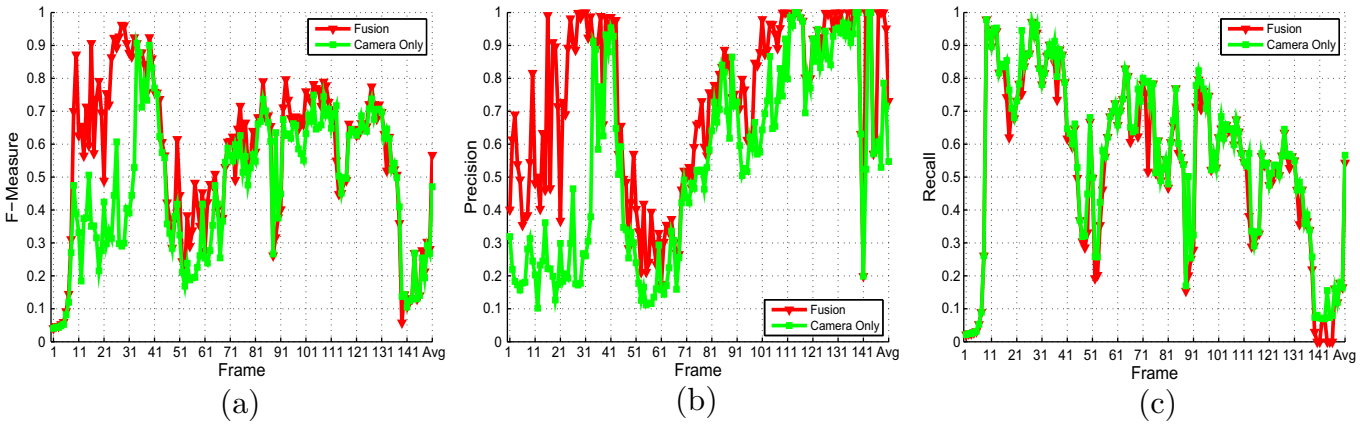


Fig. 14. The pedestrian detection performance of the proposed fusion based method (fusion) and the image based method [56] (camera only) using Dataset 5. (a) shows the *F-measure* value of Dataset 5; (b) shows *precision* value of Dataset 5; (c) shows *recall* value of Dataset 5.

two criteria: $precision = \frac{|GT \cap DR|}{|DR|}$ and $recall = \frac{|GT \cap DR|}{|GT|}$. By combining *precision* and *recall*, we use a single *F-measure* as the metric for performance evaluation. $F\text{-}measure = \frac{2 \times recall \times precision}{recall + precision}$ is the weighted harmonic mean of *precision* and *recall*. In each frame, these criteria values are first calculated for each category, respectively. Then the average value of all categories is used to evaluate one frame.

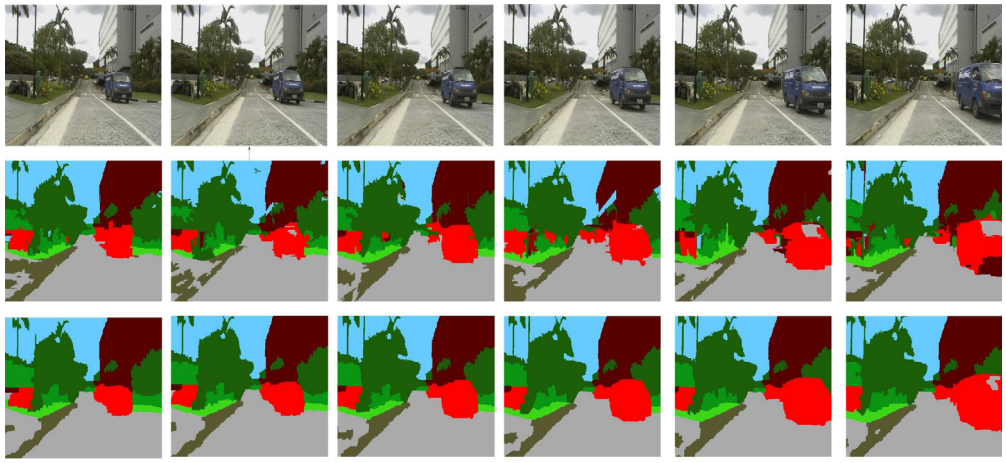
6.2. Scene parsing by fusing results of two sensors

Fig. 10 shows the result of several frames. The top panel shows the result of Dataset 1 and the bottom panel shows the result of Dataset 2. In each panel, the first row shows the original camera image and the second row shows the image classification result. The red color illustrates the region of detected obstacles and other colors have similar meaning as in Fig. 3(d). The third row shows the detected result using scanner pointcloud and the results are projected to the camera image. Each bounding box localizes one candidate obstacle. The green circles represent the projection of ground points and the blue circles represent the projection of above-ground points. There are ground points inside several bounding boxes due to the 3D–2D projection. The fourth row shows the result of our fusion method. Each white bounding box localizes one detected obstacle. In the sequences, the obstacles are subject to variations introduced by moving vehicles and pedestrians, static obstacles, road curvature changes, etc. It is possible that some frames contain only one obstacle and some frames do not contain any obstacles. These results show that the proposed approach performs well for scene parsing and obstacle detection from real-world driving environment.

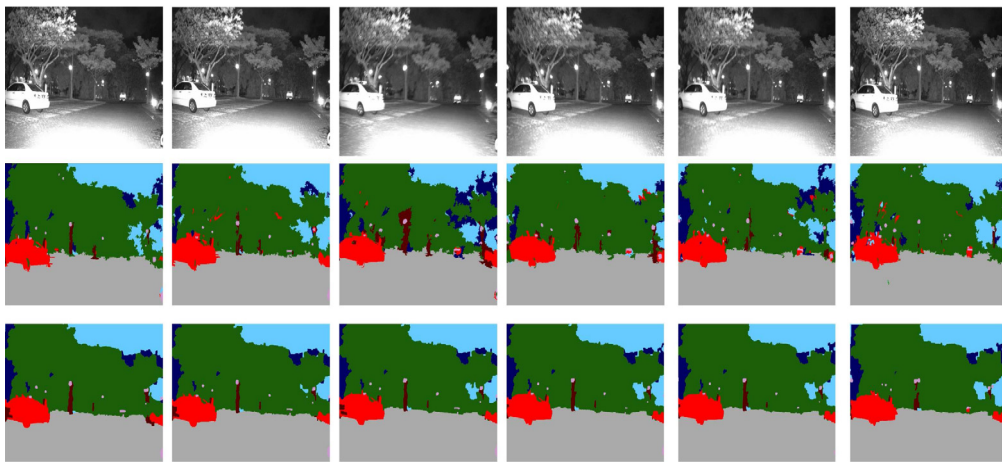
To further evaluate the proposed method, we compare its result with that of using image only. As shown in Fig. 11, our proposed fusion approach improves the scene parsing result in terms of *F-measure* value. The position of obstacles is very important information in the scenario of autonomous ground vehicles. The obstacle parsing evaluation is shown in Fig. 12 and our proposed fusion approach improves the obstacle parsing significantly in terms of *F-measure* value. This is because the detected results of our method include major parts of or the complete obstacle regions. On the contrary, the image parsing method only detects small parts of the obstacle regions due to the diversity of the obstacles. Therefore, it obtains a high *precision* value but with a low *recall* value. These comparisons clearly demonstrate the advantages of the proposed fusion method.

6.3. Pedestrian detection by using the fusion method

Recently, Geiger et al. published one dataset which has both camera image and Velodyne scanner pointcloud [54]. As this



(a)



(b)

Fig. 15. Sample results of scene passing of consecutive frames using Dataset 3 and Dataset 4. Panel (a) presents the result of day-time Dataset 3 while panel (b) presents the result of night-time Dataset 4. In each panel, the first row shows the original camera image and the second row shows the image parsing result. The third row shows the result smoothed by the temporal fusion method. The red color illustrates the region of detected obstacles and other colors have similar meaning as in Fig. 3(d). After the temporal fusion, the results between consecutive frames are more cohesive. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

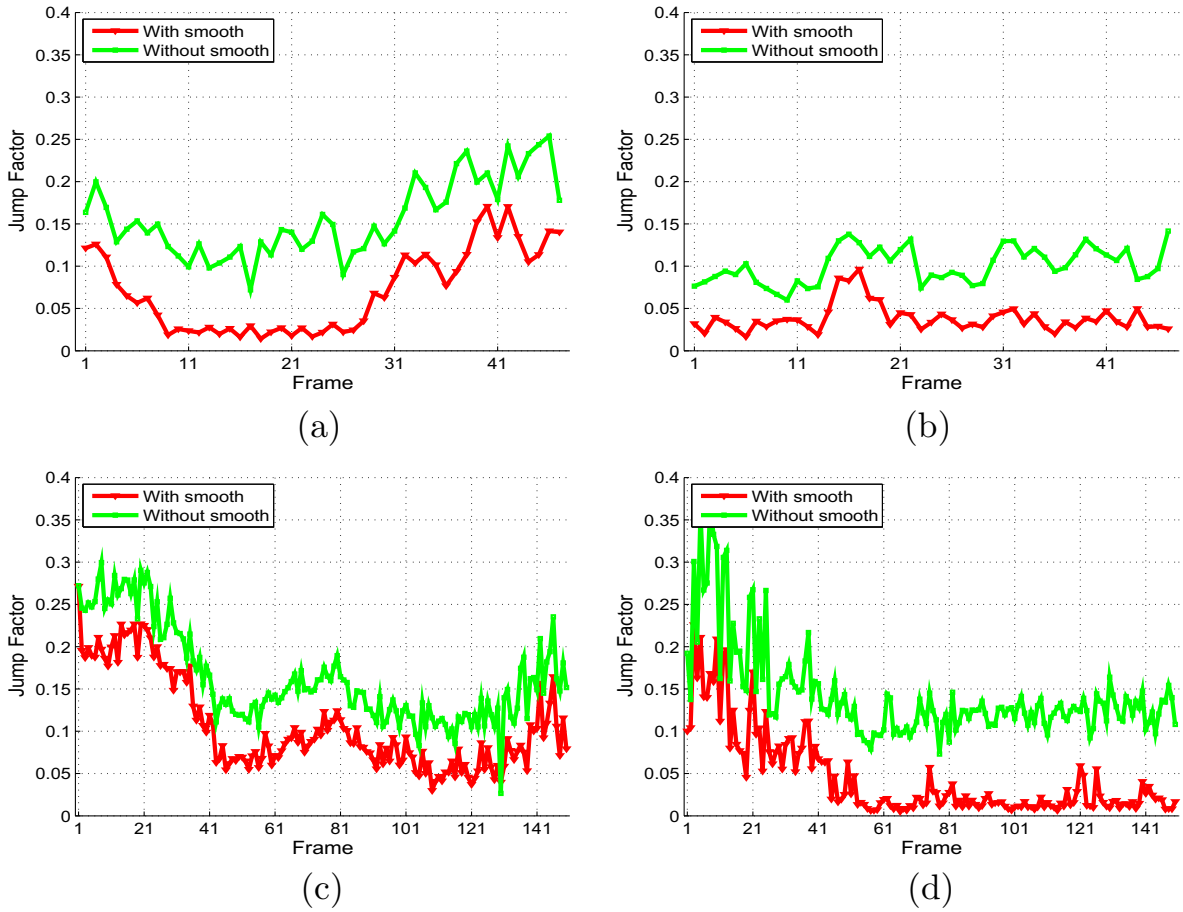


Fig. 16. The comparison of the *Jump factor* with or without using the temporal fusion. (a)–(d) show the results of Dataset 1–Dataset 4, respectively. The *Jump factor* values are significantly reduced by the proposed MRF based temporal fusion method. The *Jump factor* is only used for qualitative evaluation of the proposed temporal fusion method.

Table 3
The *F-measure* of two temporal smoothing methods in Dataset 1.

Method	Obstacle	Road	Bush	Tree	Sky	Average
MRF	**0.76**	**0.97**	**0.70**	**0.84**	**0.81**	**0.59**
MFV	0.48	0.96	0.68	0.77	0.77	0.53

Table 4
The *F-measure* of two temporal smoothing methods in Dataset 2.

Method	Obstacle	Road	Bush	Tree	Sky	Average
MRF	**0.79**	**0.94**	0.21	**0.91**	**0.68**	**0.44**
MFV	0.51	0.93	**0.24**	0.90	0.67	0.41

Table 5
The *F-measure* of two temporal smoothing methods in Dataset 3.

Method	Obstacle	Road	Building	Tree	Sky	Average
MRF	**0.35**	**0.82**	**0.67**	**0.47**	**0.57**	**0.30**
MFV	0.31	0.81	0.66	0.44	0.53	0.29

Table 6
The *F-measure* of two temporal smoothing methods in Dataset 4.

Method	Obstacle	Road	Building	Tree	Sky	Average
MRF	0.35	**0.87**	**0.24**	0.65	0.61	**0.25**
MFV	**0.37**	0.85	0.18	0.65	0.61	0.24

dataset provides only the pedestrian bounding box information, we evaluate our fusion method for pedestrian detection only. We first detect pedestrians from camera images using the method provided by Dollár et al. [56]. Each detected pedestrian is located by a bounding box and the corresponding image classification result *class* is set to be the pedestrian detection response [56]. Other fuzzy variables are set according to the description in Section 3.1. Then we apply the proposed sensor fusion method to remove the false pedestrian detections. Fig. 13 shows the sample results of pedestrian detection. It can be seen that by fusing data of camera and Velodyne, we can remove many false detections. The quantitative evaluation is shown in Fig. 14 and our proposed fusion approach improves the pedestrian detection significantly in terms of *F-measure* value. However, there is clearly room for improve the detection performance. For example, detect pedestrians who are in shadow using the Velodyne pointcloud directly as [40].

6.4. Evaluate the temporal fusion method

The temporal fusion method is proposed to smooth the results of consecutive frames as the abrupt changes of parsing results will mislead the vehicle navigation. Fig. 15 shows the sample scene parsing results of consecutive frames using two datasets in urban area. In each panel, the first row shows the original camera image. The second row shows the image parsing result before the temporal fusion and it can be seen that each frame has several places which are not cohesive with its consecutive frames. After the temporal fusion, the results are more cohesive between consecutive frames, as shown in the third row. To measure the cohesive

performance of the proposed approach, we define a *Jump factor* criterion for each frame:

$$\textit{Jump factor} = \frac{\text{No. of pixels changing label}}{\text{No. of all pixels}}.$$

The *Jump factor* represents the ratio of pixels which have different labels with their corresponding pixels in the previous frame and it is used for qualitative evaluation of the proposed temporal fusion method. The pixel to pixel correspondence between two consecutive frames are obtained by dense optical flow method [7]. The larger the *Jump factor*, the more pixels in one frame have changed their parsing labels comparing with the previous frame. Fig. 16 shows the comparison of the *Jump factor* values with or without using the temporal fusion in four datasets. It can be seen that the *Jump factor* values are significantly reduced by the proposed temporal fusion method.

To further evaluate the proposed Markov random field based temporal fusion method, we compare its result with that of multiple frames voting (MFV) method. The multiple frames voting method decides the label of pixel p in current frame by the voting of pixel p and its corresponding pixels in the $k = 4$ previous frames. The label with maximum votes is assigned to pixel p. Tables 3–6 show the comparison of *F-measure* values for four datasets. As Dataset 5 only has the bounding box label, we do not evaluate the temporal fusion performance on it. It can be seen that the proposed MRF based temporal fusion method can obtain a better performance with more than 3% improvement in terms of the average *F-measure* values for these datasets. The MRF based temporal fusion method has a better performance in Dataset 1 and Dataset 2 as the obstacles are moving fast in these two datasets. These comparisons clearly demonstrate the advantages of the proposed MRF based temporal fusion method.

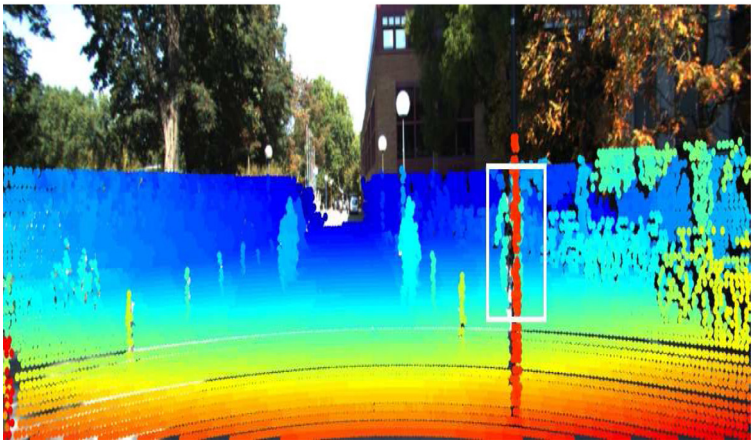
It is important to note that temporal fusion will introduce a latency in scene parsing result. This can be seen in Fig. 15(a). Although pixels behind the vehicle are classified to be road by the image classifier, the fusion method adopts these new measurements after several frames.

7. Discussions

7.1. Selection of image classifier

An MLP (multilayer perceptron) classifier has been finally chosen to parse the image superpixels due to its lower computational cost than other classifiers like kernel support vector machine (SVM) or structured learning approaches like conditional random field (CRF) [32]. According to our experiments, the linear SVM does not work in our case. The non-linear SVM with RBF kernel could achieve comparable F-measure with MLP. However, the non-linear SVM runs much slower than the MLP as the number of learned

(a)



(b)

Fig. 17. Illustration of occlusion problem. (a) shows one camera image. (b) shows the projected Velodyne points. The white bounding box localizes one cyclist. Due to the occlusion problem, wrong color information will be assigned to a number of 3D points inside the bounding box.

support vectors amounts to 8000, while the MLP contains only about 200 nodes.

Our image parser only requires about 0.5 s to process one frame (400×300 pixels) in the common PC computer. The whole system can have a real time performance after the appropriate optimization. We suggest to speed up the feature extraction using GPU parallel computing technique as Benenson et al. [57]. Besides, both the optical flow and MRF can obtain a real time implementation by using GPU technique [58]59. Moreover, both the proposed fuzzy logic based sensor fusion method and the MRF based temporal fusion method are able to integrate the results of any classifiers.

7.2. Fusing two sensors at feature level

In this paper, we have demonstrated promising results of the fuzzy logic fusion method by showing how it outperforms the results of individual sensors. Due to the sparseness of the pointcloud data of Velodyne scanner, we propose the geometry segmentation method to detect the obstacles and ground area from the scanner data. However, we do not think that our system alone is the ultimate answer to fuse Velodyne scanner and camera data. It is possible to extract discriminative features from the pointcloud sequence [43] and train a scene classifier by using both the image features and pointcloud features. Therefore, a natural future step is to combine the centralized and decentralized fusion methods for scene parsing.

7.3. Occlusion reasoning for fusion of camera and LIDAR

The fusion of data is correct if both sensors capture data from same view point. However, due to different viewpoints of both sensors, the occlusion occurs sometimes in the process of sensor fusion [15]. This lets LIDAR obtain 3D points of objects which are occluded in camera view. One occlusion example is shown in Fig. 17. Occlusion problem results in wrong fused categorization of 3D points that are not visible to camera. Although the occlusion problem is not handled in this paper, it can be solved by ordering the occluded 3D points or by using the pointcloud segmentation algorithm. Further details can be found in [15].

7.4. Integration of dynamic object tracking results

In the temporal fusion process, we represent each frame by a MRF model and integrate results of multiple previous frames. Although this can smooth the scene parsing results, the object motion information is not incorporated. By considering dynamic objects, we can leverage object detection techniques [60] and object tracking techniques [61] to obtain the category of corresponding pixels directly. Furthermore, object track information is also helpful for occlusion reasoning and collision warning.

8. Conclusions

In this paper, we present a sensor fusion method for scene parsing using laser scanner and video camera. By employing fuzzy logic inference, our method can incorporate not only results of two sensors, but also the human experience and knowledge. To smooth parsing results of consecutive frames, we further propose a Markov random field based temporal fusion method. The proposed approach has been evaluated with five datasets. Four of them are collected by our autonomous ground vehicle testbed in rural and urban areas while the fifth one is a new public vision and laser scanner dataset [54].

Our experiments underline the observation that fused results are more reliable than those provided by individual sensors. For future work it would be interesting to explore the fusion with complementary sensors such as RADAR or stereo camera, which should allow for further improvements. The feature level fusion of laser scanner and video camera also deserved to be explored. Moreover, occlusion handling and dynamic object tracking are also important for robust environment perception.

Acknowledgments

This work is supported in part by Nanyang Assistant Professorship SUG M4080134, JSPS-NTU joint project M4080882, NTU CoE seed grant M4081039, and NTU-DSO joint project M4060969.

References

[1] D. Burr, P. Binda, M. Gori, Combining vision with audition and touch, in adults and in children, Sens. Cue Integr. (2011) 167–181.
[2] Kinect, MicroSoft, Inc., Dec, 2012, <http://en.wikipedia.org/wiki/Kinect>.
[3] Velodyne, Lidar, Inc., Hdl-64e, Dec, 2012, <http://velodynelidar.com/lidar/hdlproducts/hdl64e.aspx>.
[4] L. Zadeh, Fuzzy sets, Inform. Control 8 (3) (1965) 338–353.
[5] C. Liu, J. Yuen, A. Torralba, Nonparametric scene parsing via label transfer, IEEE Trans. Pattern Anal. Mach. Intell. 33 (12) (2011) 2368–2382.
[6] J. Shotton, J. Winn, C. Rother, A. Criminisi, Textonboost for image understanding: multi-class object recognition and segmentation by jointly modeling texture, layout, and context, Int. J. Comput. Vision 81 (1) (2009) 2–23.
[7] T. Brox, A. Bruhn, N. Papenberg, J. Weickert, High accuracy optical flow estimation based on a theory for warping, Comput. Vision-ECCV 2004 (2004) 25–36.
[8] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient belief propagation for early vision, Int. J. Comput. Vision 70 (1) (2006) 41–54.
[9] G. Zhao, X. Xiao, J. Yuan, Fusion of velodyne and camera data for scene parsing, in: 15th International Conference on Information Fusion (FUSION), 2012.
[10] D. Hall, J. Llinas, An introduction to multisensor data fusion, Proc. IEEE 85 (1997) 6–23.
[11] B. Douillard, A. Brooks, F. Ramos, A 3d laser and vision based classifier, in: Proceedings of the Fifth International Conference on Intelligent Sensors, Sensor Networks and Information Processing ISSNIP, 2009.
[12] M. Häselich, M. Arends, D. Lang, D. Paulus, Terrain classification with markov random fields on fused camera and 3d laser range data, in: Proceedings of the Fifth European Conference on Mobile Robotics (ECMR), 2011, pp. 153–158.
[13] S. Laible, Y.N. Khan, K. Bohlmann, A. Zell, 3d lidar- and camera-based terrain classification under different lighting conditions, in: AMS, 2012, pp. 21–29.
[14] N. Kaempchen, M. Buehler, K. Dietmayer, Feature-level fusion for free-form object tracking using laserscanner and video, in: Intelligent Vehicles Symposium (IV), IEEE, 2005.
[15] S. Schneider, M. Himmelsbach, T. Luettel, H.-J. Wnsche, Fusing vision and lidar – synchronization, correction and occlusion reasoning, in: Intelligent Vehicles, Symposium, 2010, pp. 388–393.
[16] K. Kidono, T. Naito, J. Miura, Reliable pedestrian recognition combining high-definition lidar and vision data, in: 15th International IEEE Conference on Intelligent Transportation Systems, 2012.
[17] M. Himmelsbach, T. Luettel, F. Hecker, F. von Hundelshausen, H.-J. Wuensche, Autonomous off-road navigation for mucar-3 – improving the tentacles approach: integral structures for sensing and motion, KI (2011) 145–149.
[18] R. Labayrade, C. Royere, D. Gruyer, D. Aubert, Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner, Auton. Rob. 19 (2005) 117–140.
[19] C. Premebida, O. Ludwig, U. Nunes, Lidar and vision-based pedestrian detection system, J. Field Rob. 26 (2009) 696–711.
[20] F. Garcia, D. Olmeda, Hybrid fusion scheme for pedestrian detection based on laser scanner and far infrared camera, in: Proceedings of the Intelligent Transportation Systems Conference, 2010.
[21] W. Tang, K.Z. Mao, L.O. Mak, G.W. Ng, Z. Sun, J.H. Ang, G. Lim, Target classification using knowledge-based probabilistic model, in: Proceedings of the 14th International Conference on fusion (FUSION), 2011.
[22] B.K. Habtemariam, R. Tharmarasa, T. Kirubarajan, D. Grimmett, C. Wakayama, Multiple detection probabilistic data association filter for multistatic target tracking, in: Proceedings of the 14th International Conference on fusion (FUSION), 2011.
[23] S. Martin, Sequential bayesian inference models for multiple object classification, in: Proceedings of the 14th International Conference on fusion (FUSION), 2011.
[24] R. Matthaei, H. Dyckmanns, Motion classification for cross traffic in urban environments using laser and radar, in: Proceedings of the 14th International Conference on fusion (FUSION), 2011.
[25] D. Batra, R. Sukthankar, T. Chen, Learning class-specific affinities for image labelling, in: CVPR, IEEE Computer Society, 2008.
[26] C. Galleguillos, A. Rabinovich, S. Belongie, Object categorization using co-occurrence, location and appearance, in: CVPR, IEEE Computer Society, 2008.

[27] L. Yang, P. Meer, D.J. Foran, Multiple class segmentation using a unified framework over mean-shift patches, in: IEEE Conference on Computer Vision and, Pattern Recognition, 2007, pp. 1–8.
[28] A. Rabinovich, A. Vedaldi, C. Galleguillos, E. Wiewiora, S. Belongie, Objects in context, in: ICCV, 2007, pp. 1–8.
[29] C. Pantofaru, C. Schmid, M. Hebert, Object recognition by integrating multiple image segmentations, in: ECCV (3), 2008, pp. 481–494.
[30] D. Larlus, F. Jurie, Combining appearance models and markov random fields for category level object segmentation, in: Conference on Computer Vision and, Pattern Recognition, 2008.
[31] Z. Tu, X. Chen, A.L. Yuille, S.-C. Zhu, Image parsing: unifying segmentation, detection, and recognition, in: Proceedings of the Ninth IEEE International Conference on Computer Vision, ICCV '03, vol. 2, IEEE Computer Society, Washington, DC, USA, 2003, pp. 18–26.
[32] L. Ladický, P. Sturgess, K. Alahari, C. Russell, P.H.S. Torr, What, where and how many? Combining object detectors and CRFs, in: Proceedings of the 11th European Conference on Computer Vision: Part IV, ECCV'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 424–437.
[33] P.F. Felzenszwalb, O. Veksler, Tiered scene labeling with dynamic programming, in: CVPR, IEEE, 2010, pp. 3097–3104.
[34] J. Tighe, S. Lazebnik, Superparsing: scalable nonparametric image parsing with superpixels, in: Proceedings of the 11th European Conference on Computer Vision: Part V, ECCV'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 352–365.
[35] C. Farabet, C. Couprie, L. Najman, Y. LeCun, Scene parsing with multiscale feature learning, purity trees, and optimal covers, in: Proc. International Conference on, Machine learning (ICML'12), 2012.
[36] G.J. Brostow, J. Shotton, J. Fauqueur, R. Cipolla, Segmentation and recognition using structure from motion point clouds, in: Proceedings of the 10th European Conference on Computer Vision: Part I, ECCV '08, Springer-Verlag, Berlin, Heidelberg, 2008, pp. 44–57.
[37] P. Sturgess, K. Alahari, L. Ladicky, P.H.S. Torr, Combining appearance and structure from motion features for road scene understanding, in: BMVC, 2009.
[38] C. Zhang, L. Wang, R. Yang, Semantic segmentation of urban scenes using dense depth maps, in: Proceedings of the 11th European Conference on Computer Vision: Part IV, ECCV'10, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 708–721.
[39] J. Xiao, L. Quan, Multiple view semantic segmentation for street view images, in: ICCV, 2009, pp. 686–693.
[40] L. Spinello, M. Luber, K.O. Arras, Tracking people in 3d using a bottom-up top-down detector, in: ICRA, 2011, pp. 1304–1310.
[41] D.M. Bradley, R. Unnikrishnan, J. Bagnell, Vegetation detection for driving in complex environments, in: IEEE Int. Conf. on Robotics and Automation, 2007.
[42] A. Teichman, S. Thrun, Tracking-based semi-supervised learning, in: Robotics: Science and Systems, Los Angeles, CA, USA, 2011.
[43] J. Behley, V. Steinhage, A.B. Cremers, Performance of histogram descriptors for the classification of 3d laser range data in urban environments, in: ICRA, 2012, pp. 4391–4398.
[44] M.A. Fischler, R.C. Bolles, Readings in computer vision: issues, problems, principles, and paradigms, San Francisco, CA, USA, 1987 (Ch. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated, cartography, pp. 726–740).
[45] B. Douillard, J. Underwood, V. Vlaskine, A. Quadros, S. Singh, A pipeline for the segmentation and classification of 3d point clouds, in: International Symposium on Experimental, Robotics 2010, 2010.
[46] L. Jacoby, T. Mohan, B. Michael, Off-road terrain traversability analysis and hazard avoidance for UGVs, Technical Report, 2011.
[47] P.F. Felzenszwalb, D.P. Huttenlocher, Efficient graph-based image segmentation, Int. J. Comput. Vision 59 (2) (2004) 167–181.
[48] J. Zhang, M. Marszalek, S. Lazebnik, C. Schmid, Local features and kernels for classification of texture and object categories: a comprehensive study, Int. J. Comput. Vision 73 (2) (2007) 213–238.
[49] L. Du, L. Ren, D.B. Dunson, L. Carin, A bayesian model for simultaneous image clustering, annotation and object segmentation, in: Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, A. Culotta (Eds.), NIPS, Curran Associates, Inc., 2009, pp. 486–494.
[50] A. Oliva, A. Torralba, The role of context in object recognition, Trends Cognit. Sci. 11 (12) (2007) 52–527.
[51] D. Koller, N. Friedman, Probabilistic Graphical Models: Principles and Techniques, MIT Press, 2009.
[52] E.H. Mamdani, S. Assilian, An experiment in linguistic synthesis with a fuzzy logic controller, Int. J. Hum. Comput. Stud. 51 (2) (1999) 135–147.
[53] C.-T. Lin, C.S.G. Lee, Neural Fuzzy Systems: A Neuro-Fuzzy Synergism to Intelligent Systems, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1996.
[54] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? The kitti vision benchmark suite, in: Computer Vision and Pattern Recognition (CVPR), Providence, USA, 2012.
[55] J.-Y. Bouguet, Camera calibration toolbox for matlab, Dec, 2012, <http://www.vision.caltech.edu/bouguetj>.
[56] P. Dollár, S. Belongie, P. Perona, The fastest pedestrian detector in the west, in: BMVC, 2010.
[57] R. Benenson, M. Mathias, R. Timofte, L. Van Gool, Pedestrian detection at 100 frames per second, in: CVPR, 2012.
[58] G. Pascal, Z. Henning, G. Sven, B. Andres, W. Joachim, A highly efficient GPU implementation for variational optic flow based on the Euler–Lagrange framework, in: ECCV Workshop on Computer Vision with GPUs (CVGPU), 2010.
[59] Y. Xu, H. Chen, R. Klette, J. Liu, T. Vaudrey, Belief propagation implementation using CUDA on an NVIDIA GTX 280, in: Proceedings of the 22nd Australasian Joint Conference on Advances in Artificial Intelligence, AI '09, 2009.
[60] C. Wojek, S. Walk, S. Roth, K. Schindler, B. Schiele, Monocular visual scene understanding: understanding multi-object traffic scenes, Trans. Pattern Anal. Mach. Intell. 35 (4) (2013) 882–897.
[61] M. Teutsch, W. Kruger, J. Beyerer, Fusion of region and point-feature detections for measurement reconstruction in multi-target Kalman tracking, in: Proceedings of the 14th International Conference on Fusion (FUSION), 2011.