Frontiers of Information Technology & Electronic Engineering
www.zju.edu.cn/jzus; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

A vision-centered multi-sensor fusing approach to self-localization and obstacle perception for robotic cars*

Jian-ru XUE†, Di WANG, Shao-yi DU, Di-xiao CUI, Yong HUANG, Nan-ning ZHENG
(*Lab of Visual Cognitive Computing and Intelligent Vehicle, Xi'an Jiaotong University, Xi'an 710049, China*)
†E-mail: jrxue@xjtu.edu.cn

Received Dec. 29, 2016; Revision accepted Jan. 8, 2017; Crosschecked Jan. 10, 2017

Abstract: Most state-of-the-art robotic cars' perception systems are quite different from the way a human driver understands traffic environments. First, humans assimilate information from the traffic scene mainly through visual perception, while the machine perception of traffic environments needs to fuse information from several different kinds of sensors to meet safety-critical requirements. Second, a robotic car requires nearly 100% correct perception results for its autonomous driving, while an experienced human driver works well with dynamic traffic environments, in which machine perception could easily produce noisy perception results. In this paper, we propose a vision-centered multi-sensor fusing framework for a traffic environment perception approach to autonomous driving, which fuses camera, LIDAR, and GIS information consistently via both geometrical and semantic constraints for efficient self-localization and obstacle perception. We also discuss robust machine vision algorithms that have been successfully integrated with the framework and address multiple levels of machine vision techniques, from collecting training data, efficiently processing sensor data, and extracting low-level features, to higher-level object and environment mapping. The proposed framework has been tested extensively in actual urban scenes with our self-developed robotic cars for eight years. The empirical results validate its robustness and efficiency.

Key words: Visual perception; Self-localization; Mapping; Motion planning; Robotic car
http://dx.doi.org/10.1631/FITEE.1601873 **CLC number:** TP181

1 Introduction

Rapid integration of artificial intelligence with applications has provided some notable breakthroughs in recent years (Pan, 2016). The robotic car is one such disruptive technology that may enter real life in the near future, and this is also a good example of a hybrid artificial intelligence system (Zheng *et al.*, 2017). A robotic car needs to answer three questions during all the time of its driving: where it is, where it is going, and how to go. To answer these questions, the robotic car needs to integrate three coupled consequential tasks: self-localization, decision making and motion planning, and motion control. Among these tasks, the ability to understand the robot's surroundings lies at the core, and the robotic car's performance heavily depends on the accuracy and reliability of its environment perception technologies including self-localization and perception of obstacles.

Almost all relevant information required for autonomous driving can be acquired through vision sensors. This includes but goes well beyond lane geometry, drivable road segments, traffic signs, traffic lights, obstacle positions and velocity, and obstacle class. However, exploiting this potential of vision sensors imposes more difficulties than LIDAR, radar, or ultrasonic sensors. The sensing data of LIDAR, radar, or ultrasonic sensors involve distance and/or

* Project supported by the National Key Program Project of China (No. 2016YFB1001004) and the National Natural Science Foundation of China (Nos. 91320301 and 61273252)
ORCID: Jian-ru XUE, http://orcid.org/0000-0002-4994-9343

velocity, i.e., information necessary for vehicle control. Nevertheless, camera-based driver assistance systems have entered the automotive markets (Ulrich, 2016). However, the computer vision based approach for autonomous driving in urban environment is still an open research issue, since these state-of-the-art vision technologies are still incapable of providing the high rate of success demanded by autonomous driving. Fortunately, recent approaches to scene understanding using deep learning technologies suggest a promising future of vision-centered approach for robotic cars (Hoiem *et al.*, 2015).

In this paper, we propose a vision-centered multi-sensor fusing framework for the robotic cars' perception problem, which fuses camera, LIDAR, and GIS information consistently via geometrical constraints and driving knowledge. The framework consists of self-localization and processing of obstacles surrounding the robotic car. At first glance these two problems seem to have been well studied, and early works in this field were quickly rewarded with promising results. However, the large variety of scenarios and the high rates of success demanded by autonomous driving have kept this research alive. Specifically, integrating computer vision algorithms within a compact and consistent machine perception system is still a challenging problem in the field of robotic cars.

Self-localization is the first critical problem of the aforementioned challenges. The capability of a robot to accurately and efficiently determine its position at all times is one of the fundamental tasks essential for a robotic car to interact with the environment. Different accuracies and update frequencies of self-localization are required by various applications of a robotic car. Taking parking as an example, the accuracy needed is at the centimeter level, and the update frequency is about 100 Hz. In contrast, for routing and guidance, the required accuracy is reduced to 10–100 m level and the update frequency is about 0.01 Hz. To address GPS measurements' critical problems of low accuracy and being easily affected, the map-based method becomes one of the most popular methods for robotic cars, in which one map is used to improve upon GPS measurements and to fill in when signals are unavailable or degraded. In the line of the map-based localization method, an ideal map should provide not only a geometrical representation of the traffic environment, but also some kinds of sensor-based descriptions of the environment to alleviate the difficulty of self-localization as well as of motion planning (Fuentes-Pacheco *et al.*, 2015). However, traditional road maps for a human driver cannot be used directly for a robotic car, since it is composed of evenly sampled spatial points connected via polylines, with low accuracy, especially in urban areas, over about 5–20 m. Inevitably, building a high definition map becomes one of the core competencies of robotic cars.

Mapping approaches build geometric representations of environments. They adopt sensor-based environment description models, which integrate visual and geometric features and have been designed in conjunction with Bayesian filters so that the sensor-based description can be updated over time (Douillard *et al.*, 2009). Mapping for robotic cars through local perception information is a challenging problem for a number of reasons. Firstly, maps are defined over a continuous space; the solution space of map estimation has infinitely many dimensions. Secondly, learning a map is a 'chicken-and-egg' problem, for which reason it is often referred to as the simultaneous localization and mapping (SLAM) or concurrent mapping and localization problem (Thrun and Leonard, 2008). More specifically, the difficulty of the mapping problem can be increased by a collection of factors including map size, noise in perception and actuation, perceptual ambiguity, and alignment of spatial-temporal sensing data acquisition from different types of sensors running asynchronously. With a given map of the traffic environment, self-localization becomes the problem of determining its pose in the map.

Another critical problem is the need for high reliability in processing obstacles surrounding the robotic car. This guarantees the robotic car's safety in driving through real traffic. The robotic car needs to know positions, sizes, and velocities of the surrounding obstacles to make high-level driving decisions. However, real-time detection and tracking algorithms relying on a single sensor often suffer from low accuracy and poor robustness when confronted with difficult, real-world data (Xue *et al.*, 2008). For example, most state-of-the-art object trackers present noisy estimates of velocities of obstacles, and are difficult to track due to heavy occlusion and viewpoint changes in the real traffic environment (Ess *et al.*, 2010; Mertz *et al.*, 2013). Additionally,

without robust estimates of velocities of nearby obstacles, merging onto or off highways or changing lanes becomes a formidable task. Similar issues will be encountered by any robot that must act autonomously in crowded, dynamic environments.

Fusing multiple LIDARs and radars is an essential module of a robotic car and of advanced driver assistance systems. With the improvement of vision-based object detection and tracking technologies, integrating vision technologies with LIDAR and radars makes it possible to make a higher level of driving decision than previous methods which fuse only LIDARs with radars.

In this paper, we summarize our 8-year effort on a vision-centered multi-sensor fusing approach to the aforementioned problems, as well as lessons we have learned through the long-term and extensive test of the proposed approach with our robotic car autonomously driving in real urban traffic (Xue *et al.*, 2008; Du *et al.*, 2010; Cui *et al.*, 2014; 2016). Fig. 1 illustrates the timeline of the robotic cars we developed for the test of the vision-centered multi-sensor fusing approach.

Fig. 1 The timeline of the robotic cars for the long-term test of the vision-centered multi-sensor fusing approach

2 Related works

In this section, we present a brief survey of recent works on self-localization, and obstacle detection and tracking.

2.1 Self-localization

The core problem of self-localization is mapping, and mapping and localization were initially studied independently. More specifically, mapping for robotic cars is realized as a procedure of integrating local, partial, and sequential measurements of the car's surroundings into a consistent representation, which forms the basis for further navigation. The key to the integration lies in the joint alignment of spatial-temporal sensing data from multiple heterogenous sensors equipping the robotic car, which is usually performed off-line.

With a given map, one needs to establish correspondence between the map and its local perception, and then determines the transformation between the map coordinate system and the local perception coordinate system based on these correspondences. Knowing this transformation enables the robotic car to locate the surrounding obstacles of interest within its own coordinate frame—a necessary prerequisite for the robotic car to navigate through the obstacles. This means that the localization is actually a registration problem (Du *et al.*, 2010), and can be solved via map-matching technologies (Cui *et al.*, 2014). With its localization in the global map, the robot can obtain navigation information from the map. Additionally, the navigation information can be further used as a prior in verifying the local perception results, for the purpose of increasing the accuracy and reliability of the local perception (Cui *et al.*, 2014; 2016).

Mapping and localization were eventually known as SLAM (Dissanayake *et al.*, 2001). SLAM methods are able to reduce the accumulative drift relative to the initial position of the robotic car by using landmarks and jointly optimizing over all or a selection of poses and landmarks. Efficient optimization strategies using incremental sparse matrix factorization (Montemerlo *et al.*, 2002) or relative structure representation (Grisetti *et al.*, 2010) have been proposed to make these algorithms tractable and scalable. Thus, at a theoretical and conceptual level, SLAM is now considered a solved problem in the case that LIDARs are used to build 2D maps of small static indoor environments (Thrun and Leonard, 2008). Comprehensive surveys and tutorial papers on SLAM can be found in the literature (Durrant-Whyte and Bailey, 2006).

For large-scale localization and mapping, metric-topological mapping constructs maps that navigate between places which can be recognized perceptually (Blanco *et al.*, 2007). A popular representation is to use sub-maps that are metrically consistent, and connect them with topological constraints. Generating such a metric-topological map is based on the reconstruction of the robot path in a

hybrid discrete-continuous state space, which naturally combines metric and topological maps (Blanco *et al.*, 2008). The topological graph is usually built on top of a graph SLAM system, which can be efficiently optimized even for very large environments (Konolige *et al.*, 2011). For a robotic car with a metric-topological map, it navigates locally using local metric maps, while the overall planning is formed on the topological graph. As a consequence, the metric-topological map can be created and updated incrementally, and this avoids the computation and storage burdens of building the whole global metric map. However, several substantial issues remain in realizing SLAM solutions practically for large scale, dynamic, and complex traffic environments and notably in building and using perceptually rich maps for robotic cars.

To address these aforementioned issues, incorporating visual information into the mapping systems has been attempted. Many such research efforts are described in a recent survey (Fuentes-Pacheco *et al.*, 2015). One milestone work, involving real time SLAM using only monocular vision, has been achieved, but only for small indoor environments with fewer than 100 landmarks (Davison *et al.*, 2007). Even maps provided by these works are often in a relative sense, and they make a bidirectional map matching procedure possible, which not only produces a position and trajectory consistent with the road network but also feeds back information from the map matching to camera sensor fusing (Cui *et al.*, 2014). There are many good examples in this line of thinking. Although significant improvements have been achieved (Brubaker *et al.*, 2016), there are still some problems. For example, many visual SLAM systems still suffer from large accumulated errors when a large scale environment is being explored. This leads to inconsistent estimates of maps as well as map-based localization. Furthermore, the most common assumption of visual SLAM systems, i.e., the environments to explore are static, may become invalid since traffic environments usually contain people and vehicles in motion. Last but not least, a traffic scene is visually repetitive. There are many similar textures, such as repeated architectural elements, green belts, and walls. This makes it difficult to recognize a previously explored area.

2.2 Processing obstacles surrounding

Safe autonomous driving needs to know accurate obstacle velocities as well as positions. Remarkable progress has been achieved in highway traffic situations and other largely pedestrian-free traffic scenarios (Aeberhard *et al.*, 2015). However, an urban traffic scene with a large number of moving obstacles, in particular with many pedestrians, bicyclists, and vehicles, still poses significant challenges for reliable obstacle detection and tracking. Processing of surrounding obstacles in such scenarios is a largely unsolved problem, and the problem becomes even worse in cases where obstacles' future states required for high-level decision making and path planning need to be predicted.

Multi-sensor fusing approaches have become widely adopted in robotic cars, since no single sensor exists that fulfills the requirements for reliable obstacle detection and tracking in urban environments. In most current robotic cars, cameras, LIDAR, GPS/INS, and conventional odometry have been used to integrate as much as possible location and/or geometric information for the purpose of improving the performance of obstacle detection and tracking (Cho *et al.*, 2014). Unfortunately, using multiple heterogeneous sensors increases the complexity of the sensor fusion task, since each sensor has different characteristics that need to be considered to combine their results effectively. Since vision sensors at the moment do not reach the geometric accuracy of LIDAR, several successful LIDAR based systems for detection and tracking of moving obstacles have been built for robotic cars, and these systems work robustly and reliably with several different kinds and configurations of two- and three-dimensional LIDARs, and demonstrate impressive performance (Mertz *et al.*, 2013; Darms *et al.*, 2008; Buehler *et al.*, 2009).

Mertz *et al.* (2013) presented an elegant formulation of the multi-sensor fusion for obstacle detection and tracking, and they classified multiple sensor fusion into three levels, i.e., point level, segment-to-object level, and object level. The object level treats each sensor as a standalone system and different sensor systems are combined into one object list. Darms *et al.* (2009) proposed an architecture which encapsulates all sensor-specific algorithms into a sensor layer and a fusion layer. For each dynamic object

hypothesis in the fusion layer, two tracking models, i.e., the point model and the box model, were completed according to votes from multiple sensors detecting the objects. The false alarms and missing detections are thus dramatically reduced. Unfortunately, almost all robotic cars reported in the literature still frequently encounter a noisy estimate of the velocity of obstacles and inadequately anticipate the motion of the dynamic obstacles. This makes decision-making and motion planning challenging, and leads to a large gap between the obstacle avoidance capabilities of human drivers and autonomous driving.

Vision sensors provide the advantage that in addition to the scene geometry they deliver rich appearance information as well as obstacle semantics. Benefiting from recent significant improvement in visual tracking technologies, isolated obstacles or a small number of obstacles with transient occlusions can be tracked with acceptable reliability (Ess *et al.*, 2010). Fusing vision sensors with LIDAR for detection and tracking of obstacles is proven efficient in improving the accuracy of estimating the obstacle's motion paths and future locations (Held *et al.*, 2016), since the vision module provides classification and shape information for the fusion layer, and thus the quality for model selection, data association, and movement classification is improved. Cho *et al.* (2014) extended and improved the system in Darms *et al.* (2009) by incorporating a vision module into the sensor layer. To address problems of false alarms and wrong associations, Schueler *et al.* (2012) represented traffic environments as model-based objects and an occupancy map. The occupancy map was used to classify raw data from multiple sensors into a static or dynamic state, and the model-based objects were used to compensate for the position of dynamic objects on the occupancy map. Even so, several challenging issues remain, including temporal inconsistencies in appearance and occlusions of obstacles, reinitialization of tracking, and using minimum prior knowledge about the tracked obstacle.

3 Self-localization

In this section, we present the building of a hybrid metric-topological map in lane-level via fusion of camera, LIDAR, and GPS trajectory, and then use the map constructed for self-localization. The work presented in this section is an extension of our previous work on self-localization (Cui *et al.*, 2016). The architecture of the proposed self-localization system is illustrated in Fig. 2, which is similar to that of our previous work. The inputs of the system include a forward-looking camera, an inertial measurement unit (IMU), a standard GPS receiver, and a digital map. The system outputs lane detection and a global localization of the vehicle at the centimeter level. We first present the map generation method, and then the implementation of self-localization at the centimeter level. We have made several modifications, especially in the generation of the road boundary map, and achieved a more accurate and robust map compared with that of our previous work (Cui *et al.*, 2016).

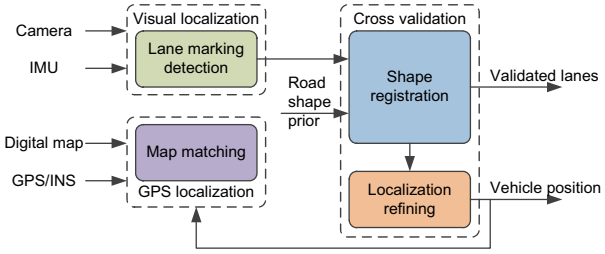

Fig. 2 Architecture of the proposed self-localization system

3.1 Generation of road boundary map

Our road boundary map is in a simple, compact, and point-by-point format, which stores the GPS positions of both the leftmost and rightmost boundaries of the road. Map data is stored in the east-north-upper (ENU) coordinate system. The road shape prior can then be presented by a sequence of GPS points after map matching. Additionally, we embedded localization of road intersections, as well as traffic regulation information of lanemarkings including dashed, solid, yellow, and double yellow, into the road boundary map for high-level driving decision making and path planning. Considering the simple format of our road boundary map, we use a three-step map generation method.

3.1.1 Step 1: data acquisition and preprocessing

The data used to build a road boundary map comes mainly from the GPS based vehicle trajectory and the vision-based detection of lanemarkings collected while the vehicle is driving. The vehicle

trajectory data was sampled using differential GPS (DGPS). In practice, as the GPS measurements are easily affected by trees, buildings, etc., the vehicle trajectory sampled by DGPS is thus not smooth enough, and one needs to remove the outliers by using a road shape model. However, since road shapes are usually complicated, a high-order complex road shape model may not be a good choice. We instead use a piecewise linear model as the road shape model, and solve it via the classic least-squares algorithm.

Suppose that the GPS based vehicle trajectory point set is denoted as $\boldsymbol{S}=\{\boldsymbol{s}_i\}_{i=1}^{N}$, where $\boldsymbol{s}_i=(x_i,y_i)$. To fit a set of new trajectory data points $\boldsymbol{S}$ to a line which connects with the previous fitted line with parameters $\{a_0,b_0\}$, we should make sure the fitted line crosses the last point (denoted as (x_0,y_0)) of the prior fitted line when applying the classic least-squares algorithm. Therefore, the objective function of the least-squares algorithm for the new line with parameters $\{a,b\}$ can be defined as follows:

$$\begin{aligned}&\min_{a,b}\sum_{i=1}^{N}[y_i-(ax_i+b)]^2\\&\text{s.t.}\quad y_0=ax_0+b.\end{aligned}\tag{1}$$

The optimal parameters of the fitted line model can be obtained by minimizing the objective function in Eq. (1), and we have

$$\begin{cases}\hat{a}=\dfrac{\sum y_ix_i-x_0\sum y_i-y_0\sum x_i+y_0x_0N}{\sum x_i^2-2x_0\sum x_i+x_0^2N},\\ \hat{b}=y_0-ax_0.\end{cases}\tag{2}$$

With this method, we can remove some outliers of the GPS point measurements, and obtain a smooth trajectory.

The lane marking is obtained via a vision based lane marking detection algorithm we proposed in Cui *et al.* (2014). Lanemarking detection has been greatly covered in the literature of autonomous driving and in the automotive industry (commercial lane detection systems). Our lane marking detection system integrates the vision based detection of lane markings and camera pose estimation with a self-built IMU. The lane markings are modeled as white bars of a particular width against a darker background in an input image. Image regions which satisfy this intensity profile can be identified through a template matching procedure over the inverse perspective mapping (IPM) of the input image. A trilinear camera calibration method (Li *et al.*, 2004) is used to guarantee the accurate IPM transformation. An IMU composed of a fiber optic gyroscope (FOG) and a speed sensor is then used to associate the lane marking detections in consecutive time steps.

3.1.2 Step 2: extracting road boundaries

As the width of the road along the vehicle trajectory varies, the estimated rightmost boundary and leftmost boundary of the road based on the fitted GPS trajectory may be noisy. To obtain the precise road boundary, we directly perform a registration between the fitted vehicle trajectory with the detected lane markings. First, we divide the vehicle trajectory into different segments according to the points in the actual intersection, and then use the iterative closest point (ICP) algorithm (Du *et al.*, 2010) to register these segments to the corresponding vision based lane marking detection. Finally, we link the transformed segments to form a more accurate road boundary.

More specifically, the vehicle trajectory is transformed to match the lane marking detection with a rigid transformation. The ICP algorithm is applied to obtain the rigid transformation, i.e., the rotation matrix and translation vector that make the vehicle trajectory and the detected lane markings best matched in terms of the Euclidean distance.

We take the following: two point sets in $\mathbb{R}^d$ (d is the dimension of points and often equals 2 or 3), a data set of the vehicle trajectory $X\equiv\{\boldsymbol{x}_i\}_{i=1}^{N_x}$, and a model set of the detected lane marking $Y\equiv\{\boldsymbol{y}_i\}_{i=1}^{N_y}$. The goal of the rigid registration is to find a rotation matrix $\boldsymbol{R}$ and a translation vector $\boldsymbol{t}$, with which the data set X best aligns with the model set Y. This can be well formulated as a least-squares (LS) problem based on the Euclidean distance described as follows:

$$\begin{aligned}&\min_{\boldsymbol{R},\boldsymbol{t},c(i)\in\{1,2,\dots,N_y\}}\sum_{i=1}^{N_x}||(\boldsymbol{R}\boldsymbol{x}_i+\boldsymbol{t})-\boldsymbol{y}_{c(i)}||_2^2\\&\text{s.t.}\quad \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R}=\boldsymbol{I}_d,\ \det(\boldsymbol{R})=1.\end{aligned}\tag{3}$$

The ICP algorithm iteratively calculates $\boldsymbol{R}$ and $\boldsymbol{t}$, and each iteration consists mainly of two steps. In the first step, correspondence needs to be found between the data point set and the model set:

$$\begin{aligned}&\min_{c(i)\in\{1,2,\dots,N_y\}}||(\boldsymbol{R}_{k-1}\boldsymbol{x}_i+\boldsymbol{t}_{k-1})-\boldsymbol{y}_{c(i)}||_2^2,\\&i=1,2,\dots,N_x,\end{aligned}\tag{4}$$

where $\boldsymbol{R}_{k-1}$ and $\boldsymbol{t}_{k-1}$ are the solutions of $\boldsymbol{R}$ and $\boldsymbol{t}$ at the $(k-1)$th iteration step, respectively.

In the second step, the rotation matrix and translation vector are updated by minimizing the following function:

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i=1}^{N_x} ||\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} - \boldsymbol{y}_{c_k(i)}||_2^2 \quad \text{s.t.} \quad \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R} = \boldsymbol{I}_d, \det(\boldsymbol{R}) = 1. \tag{5}$$

The algorithm runs iteratively until the registration error is small enough or the maximum number of iterations is reached.

3.1.3 Step 3: generating a multi-lane map

After obtaining the accurate road boundary, we need to interpolate the road data to make the boundary points point pairs. According to the detected lane markings, we can estimate the lane width. According to each point pair of the leftmost and rightmost boundaries, we can obtain the number of the lane by dividing the road width to the lane width. Finally, we can obtain the global multi-lane map via the GraphSlam algorithm (Grisetti *et al.*, 2010). The semantic information of each lane marking is also integrated into the multi-lane map. With the aforementioned steps, we finally obtain a multi-lane map illustrated in Fig. 3.

3.2 Self-localization at the centimeter level

The self-localization procedure consists of three modules: (1) visual localization, (2) GPS localization, and (3) cross validation. Both accuracy and robustness of the final localization are improved by cross validation of visual localization and GPS localization.

Visual localization is implemented by integrating lane marking detection with the camera pose measurements from an IMU. Based on lane marking localization in images output by the lane marking detection algorithm, along with camera pose measurements from the IMU, the visual localization module recovers the position and orientation of the vehicle within the lane. The road in front of the vehicle is modeled as a flat surface, which implies that there is a simple projective relationship between the image plane and the ground plane. The lane detection is then reduced to the localization of lane markings painted on the road surface. Varying illumination, shadow, and occlusions caused by other vehicles on the lane are three problems addressed in the lane marking detection algorithm.

The GPS localization outputs filtered GPS position fixes, which alone cannot recover the position of a vehicle with sufficient accuracy to perform autonomous driving. In this module, we use the GPS position fixes as an initialization to start a map matching algorithm (Hillel *et al.*, 2014), which finds the localization of the vehicle in the digital map. After the map matching procedure, we obtain a sequence of GPS position fixes to represent the shape of the road on which the vehicle is traveling, which is denoted as the road shape prior in this study. For convenience, the road shape prior is simply represented as two sets of spatially sampled GPS points corresponding to the two boundaries of the road, while clothoids have been used widely in the literature (Buehler *et al.*, 2009). Alternatively, the road shape prior can be obtained from mobile mapping data.

The cross-validation module is responsible for fusing road shape prior, visual localization, and GPS localization by a shape registration algorithm. The final localization provides a vehicle position estimate at the centimeter level. This module consists of a shape registration algorithm and a position

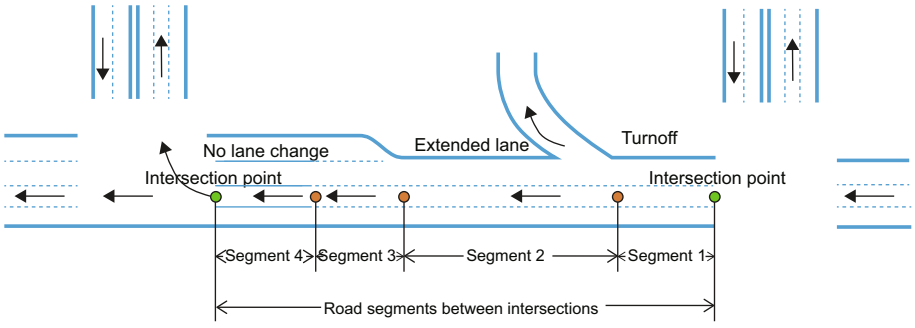

Fig. 3 The multi-lane map generated via the proposed method

refining procedure. The shape registration algorithm validates the detected lanes since the lane markings detected may contain noisy results. We introduce the road shape prior to address this problem. More specifically, shape registration is achieved by an efficient iterative closest point algorithm (ICP), which measures the similarity between the lane shape formed by the detected lane markings and the road shape prior. The position refining procedure can then correct the GPS position fixes according to translation output by ICP based shape registration.

The basis of the proposed self-localization method lies in the fact that the errors in GPS localization and visual localization are complementary in nature. On the one hand, GPS position fixes are inaccurate and at times may be unavailable altogether. However, the errors in GPS positioning fixes are bounded. On the other hand, visual localization technologies generally cannot be used to localize a vehicle accurately for indefinitely long periods of time because they do not measure absolute position. Without an occasional measurement of the absolute position, the error in localization estimate using vision technologies alone grows without bound. Using visual localization in conjunction with GPS localization can then enhance the overall performance of the proposed positioning technology.

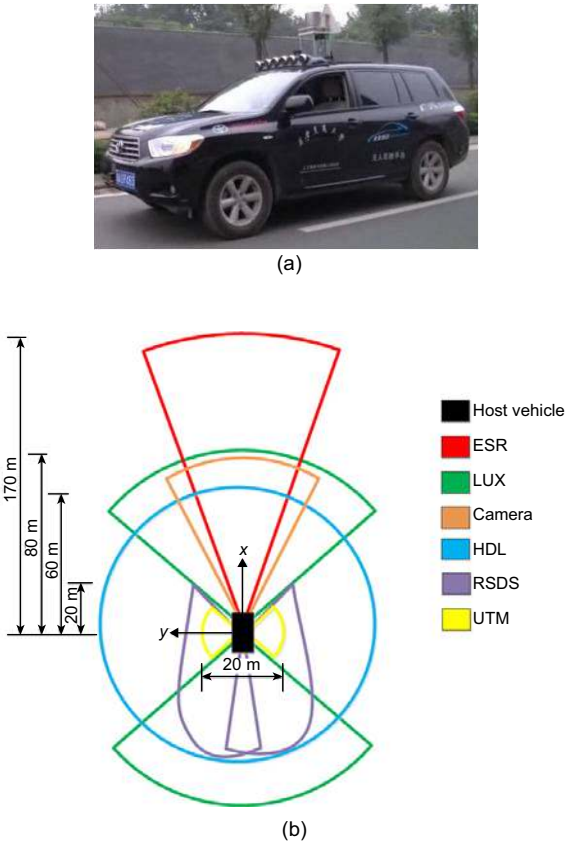

Fig. 4 The robotic car KUAFU-1 (a) and the coverage of each sensor equipped with KUAFU-1 (b) (References to color refer to the online version of this figure)

4 Multi-sensor fusing for obstacle detection and tracking

In this section, we present our vision-centered multi-sensor fusing approach to the processing of surrounding obstacles of a robotic car. We first introduce the sensor configuration of our robotic car, KUAFU-1 (Fig. 4). We then present our pedestrian detection via fusion of camera and LIDAR, and our further considerations of the multi-sensor fusing approach.

4.1 Sensor configuration and obstacle representation model

KUAFU-1 is the robotic car we used for testing our vision-centered fusing approach. Fig. 4a illustrates the robotic car and its sensors. It is equipped with five LIDARs: two 8-laser IBEO LUX-8Ls in the front and rear bumpers, two Hokuyo UTMs in the left and right sides of the rear, and one 64-laser Velodyne HDL-64E on the roof. KUAFU-1 is also equipped with three millimeter-wave radars: one is installed in the front bumper, and the other two in the left and right sides of the rear. The coverage of each sensor is as illustrated in Fig. 4b. With accurate calibration and temporal alignment, the multi-sensor system can provide a 360-degree panoramic view covering the surroundings of the robotic car.

We define three coordinate systems including camera coordinate system C_c, LIDAR coordinate system C_l, and vehicle coordinate system C_v (Fig. 5). X_c and Y_c of the camera coordinate system determine the 2D image coordinate system. During installation, the transformation between the LIDAR coordinate system and vehicle coordinate system can be adjusted to be a pure translation without rotation, which is easily measured and is convenient for calibration with other sensors. Thus, the 3D LIDAR data can be readily translated into the vehicle coordinate system. For simplicity, we denote a 3D point in the vehicle coordinate system as $\boldsymbol{X}$ in the following.

The obstacle representation models are various

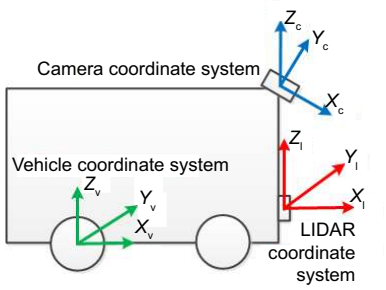

Fig. 5 Three coordinate systems defined for sensors and the vehicle of KUAFU-1

across sensors. For LIDARs, a detected obstacle is represented as a bounding box denoted as $\boldsymbol{O}_{\text{box}} = (\boldsymbol{R}, v, a, c)$, where $\boldsymbol{R} \equiv (x_{\text{c}}, y_{\text{c}}, \theta, W, L)$ represents an oriented rectangular box, $(x_{\text{c}}, y_{\text{c}})$ is the center point of the box, θ is the obstacle's heading, W and H are the width and length of the box, respectively, v is velocity, and a is acceleration. In radars, a detected obstacle is represented as a single point denoted as $\boldsymbol{O}_{\text{point}} = (\boldsymbol{P}, v, a, c)$, where $\boldsymbol{P} \equiv (x_{\text{c}}, y_{\text{c}}, \theta)$ is an orientation vector, and c is the obstacle's class label, taking its value as traffic cone, pedestrian, truck, motorcyclist, bicyclist, or unknown.

4.2 Calibrating LIDAR with camera

The successful fusion of LIDAR and camera depends heavily on the accuracy of their calibration. However, accurate camera calibration itself is a difficult problem in real mobile applications. We come up with a simple but effective semi-automatic calibration method. Different from the common method that projects all sensing data into the vehicle coordinate system for further fusion, our vision-centered approach projects all sensing data into an image to provide more accurate and robust detection and tracking, as well as augment the detected obstacles with semantic labels. The proposed calibration method can estimate both intrinsic and extrinsic parameters, and this forms a solid foundation for further fusion of LIDAR data and image data.

In the vision-centered setting, one hopes to find a mapping $\boldsymbol{P}$ which can find a given 3D point $\boldsymbol{X}$ with its corresponding pixel's location $\tilde{\boldsymbol{x}}$ in the image. The mapping $\boldsymbol{P}$ can then be represented as

$$\tilde{\boldsymbol{x}} = \alpha \boldsymbol{P} \boldsymbol{X}, \tag{6}$$

where $\tilde{\boldsymbol{x}}$ and $\boldsymbol{X}$ are homogeneous representations of $\tilde{\boldsymbol{x}}(u, v)$ and $\boldsymbol{X}(x, y, z)$, respectively, $\boldsymbol{P}$ denotes a 3×4 matrix, and α is a normalization coefficient which makes the third element of $\tilde{\boldsymbol{x}}$ be 1.

For each pair of $\boldsymbol{X}_i$ and $\tilde{\boldsymbol{x}}_i$, $\boldsymbol{P}\boldsymbol{X}$ and $\tilde{\boldsymbol{x}}$ are colinear and their cross product equals zero. Thus, Eq. (6) is written as

$$\begin{pmatrix} \boldsymbol{0}^{\text{T}} & -\boldsymbol{X}_i^{\text{T}} & v_i \boldsymbol{X}_i^{\text{T}} \\ \boldsymbol{X}_i^{\text{T}} & \boldsymbol{0}^{\text{T}} & -u_i \boldsymbol{X}_i^{\text{T}} \\ -v_i \boldsymbol{X}_i^{\text{T}} & u_i \boldsymbol{X}_i^{\text{T}} & \boldsymbol{0}^{\text{T}} \end{pmatrix} \begin{pmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{pmatrix} = \boldsymbol{0}, \tag{7}$$

where $\boldsymbol{P} = [\boldsymbol{p}_1 \quad \boldsymbol{p}_2 \quad \boldsymbol{p}_3]^{\text{T}}$ and $\boldsymbol{p}_i$ is a 4×1 vector. Since the three rows of the coefficient matrix in Eq. (7) are in linear correlation, Eq. (7) can be further simplified as

$$\begin{pmatrix} \boldsymbol{0}^{\text{T}} & -\boldsymbol{X}_i^{\text{T}} & v_i \boldsymbol{X}_i^{\text{T}} \\ \boldsymbol{X}_i^{\text{T}} & \boldsymbol{0}^{\text{T}} & -u_i \boldsymbol{X}_i^{\text{T}} \end{pmatrix} \begin{pmatrix} \boldsymbol{p}_1 \\ \boldsymbol{p}_2 \\ \boldsymbol{p}_3 \end{pmatrix} = \boldsymbol{A}_i \boldsymbol{p} = \boldsymbol{0}. \tag{8}$$

Using several pairs of $\boldsymbol{X}_i$ and $\tilde{\boldsymbol{x}}_i$, we can compose a $2n \times 12$ matrix $\boldsymbol{A}$. By solving the following equation with a singular value decomposition (SVD) algorithm, we can obtain $\boldsymbol{P}$:

$$\boldsymbol{A}\boldsymbol{p} = \boldsymbol{0} \quad \text{s.t.} \quad ||\boldsymbol{p}|| = 1. \tag{9}$$

The problem now becomes how to find the corresponding pairs $\boldsymbol{X}_i$ and $\tilde{\boldsymbol{x}}_i$. It is quite difficult to find the corresponding $\tilde{\boldsymbol{x}}(u, v)$ with a given $\boldsymbol{X}(x, y, z)$. We simplify it by assuming $z = 0$; the mapping now becomes a homography mapping. We use several traffic cones as our calibration tool. The calibration procedure is shown in Fig. 6. First, we put several traffic cones on a flat ground plane ($z = 0$) uniformly. We then label each cone's central position on the ground and its peak in the image, as the yellow and green '+' illustrated in Fig. 6a, and record these coordinates $\tilde{\boldsymbol{x}}_i$ and $\tilde{\boldsymbol{x}}_j$, respectively. The symbol '*' in Fig. 6b indicates the corresponding LIDAR points, and the symbol '○' is for the clustering results. With this setting, each cone has one '○' label and its coordinates (x, y) in the vehicle coordinate system. Since the height of a standard traffic cone is about 0.65 m, we can determine two corresponding sets of $\boldsymbol{X}$ and $\tilde{\boldsymbol{x}}$ manually. One set is $\tilde{\boldsymbol{x}}_i$ and $\boldsymbol{X}_i(x, y, 0)$, and the other is $\tilde{\boldsymbol{x}}_j$ and $\boldsymbol{X}_j(x, y, 0.65)$. According to Eq. (8), we denote them as $\boldsymbol{A}_1$ and $\boldsymbol{A}_2$ to compose $\boldsymbol{A}$ as $[\boldsymbol{A}_1^{\text{T}} \ \boldsymbol{A}_2^{\text{T}}]^{\text{T}}$, and we then obtain Eq. (9). The green symbol '+' in Fig. 6a is the mapping result of LIDAR data $\boldsymbol{X}_j$.

In general, we cannot recover the 3D geometrical structure with a single image alone. However,

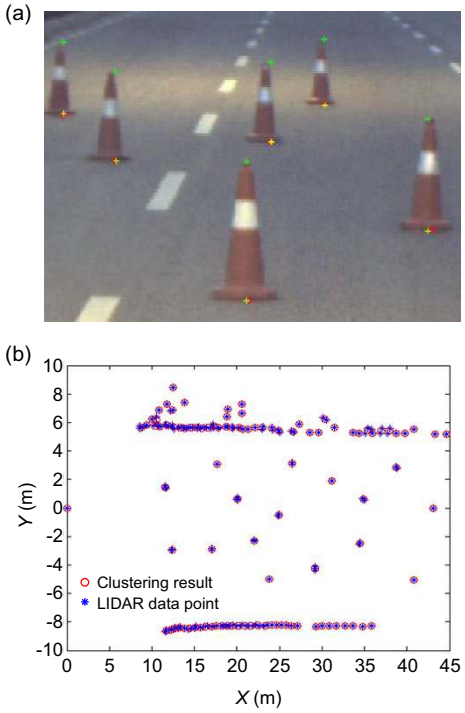

Fig. 6 The calibration setting for fusing camera and LIDAR with traffic cones: (a) manually labeled points '+' in the image; (b) corresponding LIDAR data points and clustering results (References to color refer to the online version of this figure)

a special case occurs with all points on a ground plane. For a point on the ground plane $\boldsymbol{X}(x, y, 0, 1)$, the third column of $\boldsymbol{P}$ is useless, and the mapping in Eq. (6) can be simplified as

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \alpha \boldsymbol{H} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}, \tag{10}$$

where $\boldsymbol{H}$ is a full rank matrix, which is composed by the first, second, and fourth columns of $\boldsymbol{P}$.

4.3 Pedestrian detection via fusing LIDARs and cameras

In this section, we present the method which fuses LIDARs and cameras to improve the accuracy of pedestrian detection. The pedestrian detector we chose in this study is based on the object detection method proposed in Dollár *et al.* (2014), since it provides an efficient image feature computation mechanism, and the pedestrian detector is reported as being successful in several public pedestrian data sets. However, we found false detections frequently when using it in a real urban traffic environment.

4.3.1 Geometrical verification

We reduce the false alarm rate of the pedestrian detector via fusion due to two observations: (1) The ground is flat, and pedestrians are on the ground. (2) The location, especially the bottom edge of the bounding box of the pedestrian, is usually accurate. Thus, we can use the prior knowledge of a pedestrian's height to remove some false detection from visual detection. We denote the center of the bottom edge of the bounding box as $(u_{\mathrm{g}}, v_{\mathrm{g}})$, and the distance of the detection can be easily calculated according to $\boldsymbol{H}^{-1}$ and $(u_{\mathrm{g}}, v_{\mathrm{g}})$. Since the pitch angle of the camera is $\theta_z = 0.58^\circ \approx 0^\circ$, we can estimate the height H of the pedestrian according to pinhole imaging theory:

$$H = \frac{hx}{f/m_x}, \tag{11}$$

where h is the height estimated in the image, x the x-axis component for the pedestrian's position in the camera coordinate system, f the focal length of the camera, and m_x the pixel size. If the estimated H does not fall into the height range predefined, the detection can be discarded as a false detection. However, sometimes H may be wrongly calculated due to inaccurate detection of x.

To improve the robustness, we use LIDAR measurements instead to estimate x, i.e., $x = X - \Delta d$ (the rotation matrix between camera coordinate and LIDAR coordinate is almost an identity matrix), and $\Delta d = 1.7$ m. As illustrated in Fig. 7, the symbol '+' indicates the LIDAR data mapped onto the image, the red and yellow boxes are the original detections, and the red box is the false detection found by this method. The pedestrian's LIDAR data is searched as follows: for a box (x, y, w, h), the search region of the image is also a rectangle $(x, y + h - \delta, w, 2\delta)$, where δ is the height threshold for a search rectangle. We use $(u_{\mathrm{g}}, v_{\mathrm{g}})$ to search the nearest LIDAR data, and take the data as the pedestrian's data.

Fig. 7b illustrates the non-maximum suppression (NMS) of the original detection results. Fig. 7c shows the results of false alarm filtering via height constraints. Fig. 7d shows the NMS of the detection results after removal of false detections.

4.3.2 False alarm removal by overlapped area

The most frequently used NMS approach chooses the bounding box with the highest score as

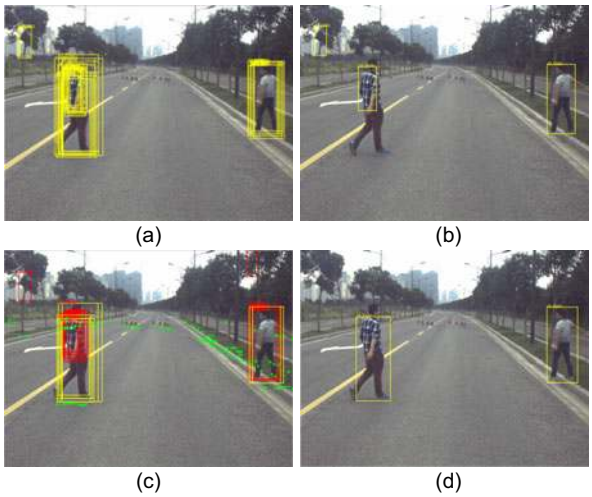

Fig. 7 Pedestrian detection results: (a) original detections; (b) NMS (nonmaximal suppression) without false alarm removal; (c) false alarm filtering via height constraints; (d) NMS with false detection removal. The boxes in yellow indicate the detections, and boxes in red indicate the false detection which can be removed via geometric cues (References to color refer to the online version of this figure)

the detection result if multiple boxes are overlapped. However, we find that a false alarm occurs due to small regions of a pedestrian, as the yellow boxes shown in Fig. 7b. The geometrical verification rule for removing false alarms is under an assumption that the small boxes are false alarms if they overlap with a large box. The rule can be expressed as

$$\begin{cases} \alpha = \dfrac{\text{area}(R_\text{s} \cap R_\text{l})}{\text{area}(R_\text{s})}, \\ \beta = \dfrac{\text{area}(R_\text{s} \cap R_\text{l})}{\text{area}(R_\text{l})}, \end{cases} \tag{12}$$

where R_s and R_l are the small box and the large box, respectively, and area$(\cdot)$ denotes the area. We use $\alpha > 0.9$ && $\beta < 0.6$ as a criterion for taking a small box as a false detection.

4.3.3 Reducing the search region

Most of the pedestrian detection methods spend too much time on feature extraction, which needs to calculate a multi-resolution representation of the image to meet scale invariance. The pedestrian detector proposed in Dollár *et al.* (2014) needs 322 ms to detect a 1292×964 image, in which 285 ms is spent on feature extraction. To meet the real-time requirement, we need to reduce the search region so that only pixels belonging to obstacles are left by finding the pixels corresponding to LIDAR data.

We can make a general assumption that a LIDAR data $\boldsymbol{X} = (x, y, 0)$ belongs to an obstacle in the front of the robotic car, and in this case $y = 0$. For a given x, the true value of y falls into an interval defined as $(y - x\tan(\theta/2), y + x\tan(\theta/2))$, where θ denotes the LIDAR's angle resolution (0.8° for the LIDAR used in the experiments). By taking this uncertainty into account, the pixels corresponding to LIDAR data (Fig. 8a) are determined via Eq. (10). They can be extended to those as shown in Fig. 8c. The image resolution on the x-axis is nonlinear with respect to a given y (Fig. 8b). We can approximately learn this nonlinear mapping off-line empirically, and implement it with a lookup table. The image resolution on the y-axis is very small, and thus we do not need to consider it.

In practice, by assuming that the highest pedestrian is 2.1 m tall, we can obtain corresponding h pixels in the image according to Eq. (11). As shown in Fig. 8a, the green bars are points with a height of 2.1 m in the image. Considering the inaccuracy of calibration and slight up and downs of the ground, we add margins to these bars according to the learned lookup table. A search region generated with this method is illustrated in Fig. 8c. To further speed up feature calculation, connected component analysis is performed, and the resulting regions with their bounding box are illustrated in Fig. 8d. With such a simple process, the detection can be sped up by several times.

4.4 Further considerations for multi-sensor fusion

In addition to pedestrian detection, KUAFU-1 is capable of perceiving other traffic participants which include, but are not limited to, traffic cones, vehicles, and bicyclists. We summarize some common considerations in the following.

4.4.1 Fused obstacle representation

A unified representation model regardless of sensors forms the basis for multi-sensor fusion, and it should consist of essential information such as shape, kinematic, and semantic information, and it is conveniently represented as $\boldsymbol{F} = (\boldsymbol{O}_\text{fusion}, \boldsymbol{s}, c)$. $\boldsymbol{O}_\text{fusion}$ can be the following types: (a) box obstacle $\boldsymbol{O}_\text{box}$, (b) box-pair obstacle $\boldsymbol{O}_\text{boxpair}$ as explained in Section 4.4.3, and (c) point obstacle $\boldsymbol{O}_\text{point}$. $\boldsymbol{s}$ is the

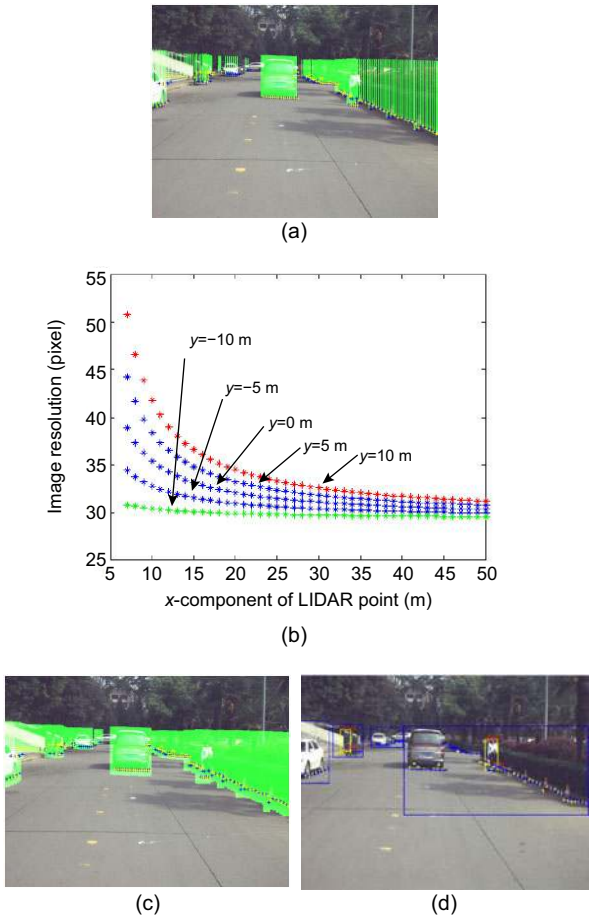

Fig. 8 The result of detection area reduction: (a) LIDAR points are projected onto the image with a height of a pedestrian (blue and yellow points are from the second and the third layers of LIDAR, respectively); (b) image resolution for different range LIDAR points (from green to red, the curves correspond to fixed y-component of LIDAR points with $y = \{-10, -5, 0, 5, 10\}$); (c) the results after adding the margins; (d) results after connected component analysis (References to color refer to the online version of this figure)

state vector which indicates the state of detection for each sensor. For front obstacles, the state vector is defined as $\boldsymbol{s} \equiv [s_{\text{lux}} \ s_{\text{hdl}} \ s_{\text{radar}} \ s_{\text{cam}}]^{\text{T}}$. For example, $s_{\text{lux}} = 1$ means the obstacle is detectable by LUX, $s_{\text{lux}} = 0$ means the obstacle cannot be detected by LUX. The state vector is important for determining the existence of an obstacle. c is the obstacle's class label, taking its value as traffic cone, pedestrian, truck, motorcyclist, bicyclist, or unknown.

4.4.2 Spatial and temporal alignment

To detect an obstacle via multi-sensor fusion, the spatial-temporal alignment of sensing data should be considered first. With calibration parameters tuned at raw sensing data (LIDAR points or image pixels), the obstacle measurements from different sensors are transformed into the vehicle coordinate system. For each individual sensor, an obstacle detection and tracking algorithm is implemented subsequently to formulate a track list. The temporal alignment is used to compensate for the motion of obstacles due to sensors running asynchronously.

4.4.3 Data association

There exist two kinds of data association, i.e., box-to-box association for obstacles from different LIDARs, and point-to-box association for obstacles from LIDAR and radar respectively. For simplification, box-to-box association is performed first, and the output is a box/box-pair obstacle. The box/box-pair obstacle is subsequently associated with point obstacle from radar.

For two box obstacles $\boldsymbol{O}_0 = (\boldsymbol{R}_0, v_0, a_0)$ and $\boldsymbol{O}_1 = (\boldsymbol{R}_1, v_1, a_1)$ from different LIDARs ($\boldsymbol{R}_0$ and $\boldsymbol{R}_1$ are oriented rectangular boxes from two LIDARs, respectively), the intersection-over-union (IOU) measure is defined as

$$\text{IOU}(\boldsymbol{R}_0, \boldsymbol{R}_1) = \frac{\text{area}(\boldsymbol{R}_0 \cap \boldsymbol{R}_1)}{\text{area}(\boldsymbol{R}_0 \cup \boldsymbol{R}_1)}. \tag{13}$$

The IOU measures the similarity of the two boxes: value 1 means perfect match, value 0 means that two boxes are either not overlapped or matched. The association rule here is simple: $\text{IOU}(\boldsymbol{R}_0, \boldsymbol{R}_1) \geq \delta$, where δ is the threshold empirically determined (in our experiments, it is set as 1.0). If two boxes are associated successfully, they comprise a box-pair obstacle denoted as $\boldsymbol{O}_{\text{boxpair}} = \{\boldsymbol{O}_0, \boldsymbol{O}_1, \boldsymbol{R}\}$, where $\boldsymbol{R} \equiv \boldsymbol{R}_0 \cup \boldsymbol{R}_1$ is the bounding box containing $\boldsymbol{R}_0$ and $\boldsymbol{R}_1$. $\boldsymbol{R}$ is useful for ROI generation for obstacle classification.

For box obstacle $\boldsymbol{O}_0 = (\boldsymbol{R}_0, v_0, a_0)$ and point obstacle $\boldsymbol{O}_1 = (\boldsymbol{P}_1, v_1, a_1)$, the distance between box obstacle and point obstacle is calculated as

$$d = \begin{cases} 0, & \boldsymbol{P}_1 \in \boldsymbol{R}_0, \\ \min\limits_{i=1,2,3,4}(\|\boldsymbol{P}_1, \boldsymbol{l}_i\|), & \boldsymbol{P}_1 \notin \boldsymbol{R}_0, \end{cases} \tag{14}$$

where $\boldsymbol{P}_1 \in \boldsymbol{R}_0$ denotes that the radar point is within the box, and vice versa. $\boldsymbol{l}_i$ is one of the four edges of the box $\boldsymbol{R}_0$. For a box-pair obstacle $\boldsymbol{O}_p = \{\boldsymbol{O}_0, \boldsymbol{O}_1, \boldsymbol{R}\}$, the distance d is calculated in

the same way except that $\boldsymbol{R}_0$ is replaced by $\boldsymbol{R}$. The association rule is $d \geq \tau$, where τ is a threshold and determined empirically (it is set as 1.0 in our experiments).

The radar can measure the velocity of a dynamic obstacle accurately. Thus, the velocity and acceleration of a box/box-pair obstacle should be replaced with that of the point obstacle in the case that the point obstacle is associated with the box/box-pair obstacle.

4.4.4 State estimation

Finally, we use a multi-hypothesis-tracker (Xue *et al.*, 2008) to estimate the state of the fused obstacle.

In our setting, obstacles from different sensors are treated as obstacle measurements. Obstacles in the real traffic road environment are sparse, and thus the multi-object tracking problem can be well divided into single-object tracking problems. With the data association rules aforementioned, the fused obstacle state can be easily estimated via a Bayesian filter.

5 Experiments and discussions

In the following, we demonstrate the effectiveness of the vision-centered mapping and obstacle detection method.

5.1 Mapping and localization

To demonstrate the performance of our method, we test our algorithms on the robotic car KUAFU-1 driving in a real road environment. First, we evenly sample a sequence of GPS positions from a vehicle trajectory (Fig. 9a). The sampled data is then fitted into many continuous line segments via the piecewise linear model presented in Section 3.1.1.

Second, we use the ICP algorithm to register the vehicle trajectory to the detected lane markings, and the results are shown in Fig. 9b. It shows that the traveling track points are perfectly aligned with the detected lane data.

Finally, by mapping the road boundary and the lane markings, we obtain a global map of all the lane as well as lane markings. Figs. 10a and 10b present a map of the real road environment using our method. To validate the performance of our mapping method,

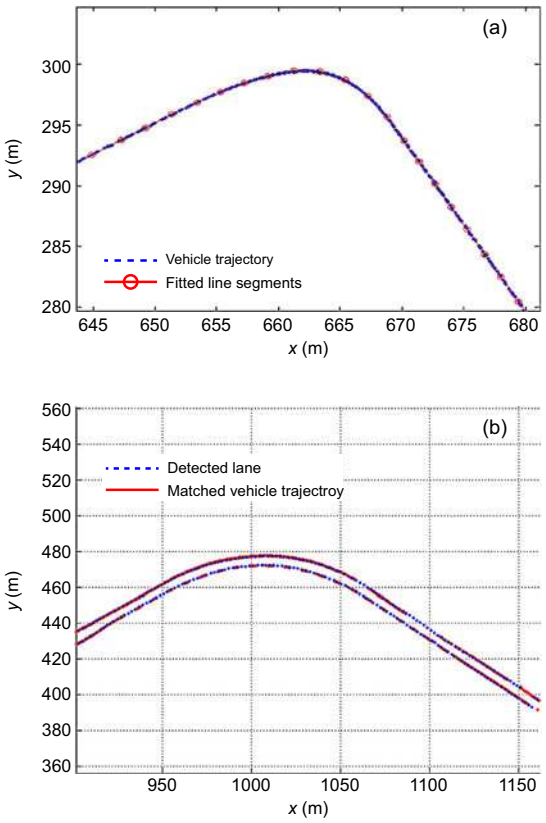

Fig. 9 The fitting result of the vehicle trajectory (a) and the registration result of the vehicle trajectory and detected lane (b)

the corresponding Google Earth map of the road is presented in Fig. 10c for comparison.

5.2 Calibration

Based on Eq. (9) and SVD decomposition, we obtain

$$\boldsymbol{P} = \begin{pmatrix} -340.20 & 1366.47 & -2.56 & -6.46 \\ -166.25 & 52.95 & 1338.32 & -2087.32 \\ -0.6267 & 0.04 & 0.011 & 1.00 \end{pmatrix}.$$

According to the method in Hartley and Zisserman (2004), $\boldsymbol{P} = \boldsymbol{K}[\boldsymbol{R}| - \boldsymbol{R}\tilde{\boldsymbol{C}}]$, where $\boldsymbol{R}$ and $\boldsymbol{R}\tilde{\boldsymbol{C}}$ denote rotation and translation of the camera coordinate system with respect to the vehicle coordinate system, respectively, and $\boldsymbol{K}$ denotes the intrinsic parameters of the camera, and these form an upper triangular matrix:

$$\boldsymbol{K} = \begin{pmatrix} \alpha_z & s_\mathrm{d} & z_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{15}$$

where α_z and α_y are the scale factors in the Z_c and Y_c directions, respectively (Fig. 4b), s_d the distortion

parameter, and $(z_0, y_0)^{\mathrm{T}}$ the principal point. Using RQ decomposition, we obtain

$$\boldsymbol{K} = \begin{pmatrix} 2135.7 & 51.39 & 679.46 \\ 0 & 2126.51 & 307.20 \\ 0 & 0 & 1 \end{pmatrix},$$
$$\tilde{\boldsymbol{C}} = (1.65 \;\; 0.42 \;\; 1.75),$$
$$\theta_x = 3.44°, \theta_y = 2.28°, \theta_z = 0.58°,$$

where θ_i denotes the rotation angle turning around the i-axis. The camera we used has a focal length $f = 8$ mm and a pixel size $m_x = 3.75$ μm, and we can estimate the intrinsic parameters as $\alpha_z = \alpha_y = f/m_x = 2133$. The translation vector $\tilde{\boldsymbol{C}} = (1.63, 0.45, 1.74)$ can be measured directly. The rotation angles are difficult to measure directly, can be estimated only by visual inspection, and are all almost 0°. The calibration result is presented in Table 1. We can find that the calibration parameters estimated are almost as accurate as those from the camera calibration method in Zhang (2000).

5.3 Pedestrian detection

We collected a short video using our robotic car and label pedestrians manually per frame, and use it as the ground truth. To validate the robustness of the fusion method, we test it in a real road environment with uneven ground.

There are 427 labeled pedestrians appearing in the video. The thresholds are set as in Table 2. The precision-recall curves with different pedestrian detection methods are plotted as shown in Fig. 11a. The ORG denotes the original detection results, GEO for the fusing results in which only image is used, and LDR for fusing results which using both LIDAR data and image. It shows that LDR can increase the precision up to 10% in case of recall > 0.7, compared with ORG. In Fig. 11b, the OLP denotes the results when using an overlapped area to filter out the false alarm. Obviously, LDR plus OLP increases precision slightly, while GEO plus OLP decreases precision heavily. This is because OLP cannot cope with an uphill road environment.

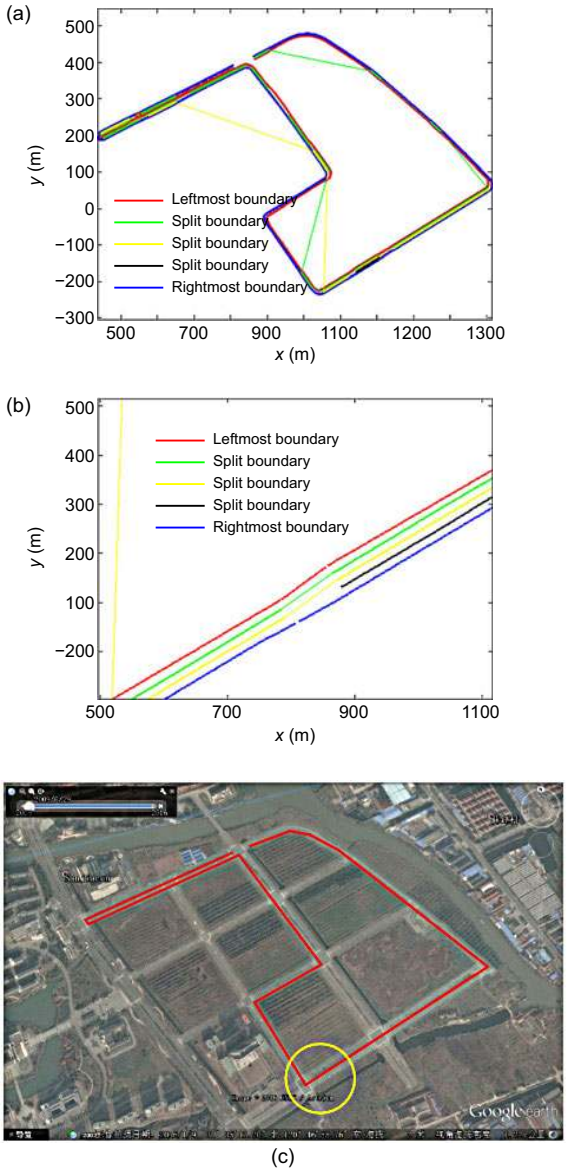

Fig. 10 The results of the multi-lane mapping: (a) global multi-lane map obtained by our method; (b) magnification of the local multi-lane map; (c) corresponding road in Google Earth, where the red points are robotic car's trajectory for generating the global multi-lane map in (a) and the yellow circle indicates a coarse position of the local multi-lane map in (b) (References to color refer to the online version of this figure)

Table 1 Calibration results of different methods

Method	α_z	s	Z_0	y_0	$\tilde{\boldsymbol{C}}$	$\boldsymbol{\theta}$
Measured	2133	0	***	***	(1.63, 0.45, 1.74)	(0.00, 0.00, 0.00)
Zhang (2000)	2303	0	673	622	***	***
Ours	2135	51.39	679	307	(1.65, 0.42, 1.75)	(3.44, 2.28, 0.58)

'***' means that the parameter cannot be estimated via the corresponding method

Table 2 Threshold setting of mentioned methods

Method	Threshold
Pedestrian's height interval	[0.8, 2.1] m
LIDAR's search region	$\delta = 40$
False alarm reduction	$\alpha > 0.9$ && $\beta < 0.6$

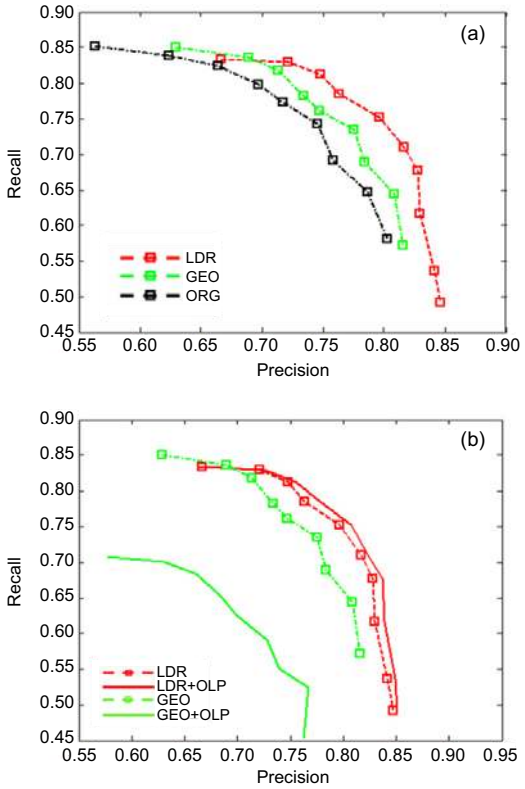

Fig. 11 Precision-recall curves for LDR, GEO, and ORG (a) and precision-recall curves after adding the OLP method (b) (References to color refer to the online version of this figure)

The proposed method may not work as expected under some extreme situations. Fig. 12 presents a failure of our method. The inaccurate mapping of LIDAR data is caused by uphill. In the case that there is a steep uphill, pedestrian detection may be discarded completely, just as illustrated in Fig. 12.

5.4 Multi-sensor fusion

To test the effectiveness of the multi-sensor fusion algorithm, KUAFU-1 was autonomously driving in the campus of Xi'an Jiaotong University, China, to collect data. The fusion results in Fig. 13 show that two pedestrians and a car are correctly detected by fusion of camera, LIDAR, and radar. The two small images in the second row of Fig. 13 show that multiple obstacles (from either LIDAR or radar) are

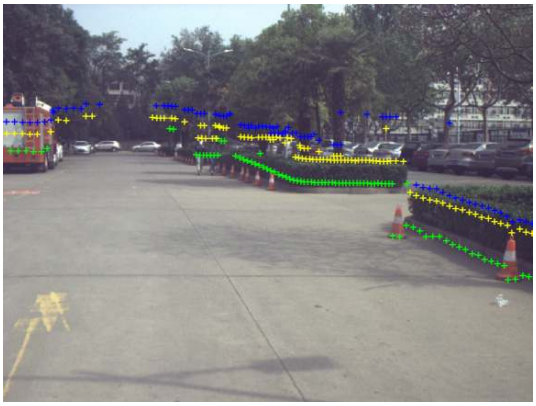

Fig. 12 A scenario in which the pedestrian in the front was discarded incorrectly because the road is uphill, which causes an incorrect mapping of LIDAR data to the image (References to color refer to the online version of this figure)

Fig. 13 The effect of the multi-sensor fusion algorithm on a campus environment. The yellow and cyan boxes are pedestrians and the car detected via camera and LIDAR/radar, respectively, the red circle is a point obstacle returned by radar, and the blue box is a box obstacle returned by LIDAR (References to color refer to the online version of this figure)

projected into the same region of the image. Our multi-sensor fusion method successfully dealt with it by using prior information including: (1) the size of the bounding box of the obstacle is bounded, and (2) range information and class information are assigned only to the nearest obstacle.

6 Conclusions

In this paper, we presented a vision-centered environment perception framework for robotic cars driving in urban environments. We applied machine

vision algorithms as the core of the environment perception system, and proposed algorithms including vision-centered mapping and localization, as well as vision-centered obstacles detection and recognition.

We found that the proposed vision-centered multi-sensor fusing method works well through a long-term test based on the robotic car KUAFU-1 autonomously driving in various real urban traffic environments. We showed the performance of the environment perception by integrating machine vision technologies with LIDAR and radar, as well as our efforts in making most of the camera system in robotic cars. We believe that the camera is the ideal sensor for a robotic car.

We demonstrated the performance of the vision-centered multi-sensor fusing approach from two aspects. The first aspect is vision-centered multi-sensor fusing for self-localization. We successfully constructed the hybrid map by a vision-centered mapping method. The hybrid map indeed improves the accuracy of self-localization from the meter level to the centimeter level in a real urban traffic environment. It also shows that using the constructed hybrid map can greatly improve the perception of lane markings, road shapes, traffic lights, and obstacles. The second aspect is vision-centered multi-sensor fusing for processing of obstacles surrounding the robotic car. Experimental results showed that both accuracy and robustness of obstacle detection and tracking have been improved greatly by our method.

References

Aeberhard, M., Rauch, S., Bahram, M., *et al.*, 2015. Experience, results and lessons learned from automated driving on Germany's highways. *IEEE Intell. Transp. Syst. Mag.*, **7**(1):42-57. http://dx.doi.org/10.1109/MITS.2014.2360306

Blanco, J.L., Fernádez-Madrigal, J.A., González, J., 2007. A new approach for large-scale localization and mapping: hybrid metric-topological SLAM. Proc. IEEE Int. Conf. on Robotics and Automation, p.2061-2067. http://dx.doi.org/10.1109/ROBOT.2007.363625

Blanco, J.L., Fernádez-Madrigal, J.A., González, J., 2008. Toward a unified Bayesian approach to hybrid metric-topological SLAM. *IEEE Trans. Robot.*, **24**(2):259-270. http://dx.doi.org/10.1109/TRO.2008.918049

Brubaker, M.A., Geiger, A., Urtasun, R., 2016. Map-based probabilistic visual self-localization. *IEEE Trans. Patt. Anal. Mach. Intell.*, **38**(4):652-665. http://dx.doi.org/10.1109/TPAMI.2015.2453975

Buehler, M., Iagnemma, K., Singh, S., 2009. The DARPA Urban Challenge: Autonomous Vehicles in City Traffic. Springer. http://dx.doi.org/10.1007/978-3-642-03991-1

Cho, H., Seo, Y.W., Kumar, B.V.K.V., *et al.*, 2014. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. IEEE Int. Conf. on Robotics and Automation, p.1836-1843. http://dx.doi.org/10.1109/ICRA.2014.6907100

Cui, D., Xue, J., Du, S., *et al.*, 2014. Real-time global localization of intelligent road vehicles in lane-level via lane marking detection and shape registration. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, p.4958-4964. http://dx.doi.org/10.1109/IROS.2014.6943267

Cui, D.X., Xue, J.R., Zheng, N.N., 2016. Real-time global localization of robotic cars in lane level via lane marking detection and shape registration. *IEEE Trans. Intell. Transp. Syst.*, **17**(4):1039-1050. http://dx.doi.org/10.1109/TITS.2015.2492019

Darms, M., Rybski, P., Urmson, C., 2008. Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments. IEEE Intelligent Vehicles Symp., p.1197-1202. http://dx.doi.org/10.1109/IVS.2008.4621259

Darms, M., Rybski, P., Baker, C., *et al.*, 2009. Obstacle detection and tracking for the urban challenge. *IEEE Trans. Intell. Transp. Syst.*, **10**(3):475-485. http://dx.doi.org/10.1109/TITS.2009.2018319

Davison, A.J., Reid, I.D., Molton, N.D., *et al.*, 2007. MonoSLAM: real-time single camera SLAM. *IEEE Trans. Patt. Anal. Mach. Intell.*, **29**(6):1052-1067. http://dx.doi.org/10.1109/TPAMI.2007.1049

Dissanayake, M.W.M.G., Newman, P., Clark, S., *et al.*, 2001. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Autom.*, **17**(3):229-241. http://dx.doi.org/10.1109/70.938381

Dollár, P., Appel, R., Belongie, S., *et al.*, 2014. Fast feature pyramids for object detection. *IEEE Trans. Patt. Anal. Mach. Intell.*, **36**(8):1532-1545. http://dx.doi.org/10.1109/TPAMI.2014.2300479

Douillard, B., Fox, D., Ramos, F., 2009. Laser and vision based outdoor object mapping. Robotics: Science and Systems IV, p.9-16.

Du, S.Y., Zheng, N.N., Xiong, L., *et al.*, 2010. Scaling iterative closest point algorithm for registration of *m-D* point sets. *J. Vis. Commun. Image Represent.*, **21**(5-6):442-452. http://dx.doi.org/10.1016/j.jvcir.2010.02.005

Durrant-Whyte, H., Bailey, T., 2006. Simultaneous localization and mapping: part I. *IEEE Robot. Autom. Mag.*, **13**(2):99-110. http://dx.doi.org/10.1109/MRA.2006.1638022

Ess, A., Schindler, K., Leibe, B., *et al.*, 2010. Object detection and tracking for autonomous navigation in dynamic environments. *Int. J. Robot. Res.*, **29**(14):1707-1725. http://dx.doi.org/10.1177/0278364910365417

Fuentes-Pacheco, J., Ruiz-Ascencio, J., Rendón-Mancha, J.M., 2015. Visual simultaneous localization and mapping: a survey. *Artif. Intell. Rev.*, **43**(1):55-81. http://dx.doi.org/10.1007/s10462-012-9365-8

Grisetti, G., Kummerle, R., Stachniss, C., *et al.*, 2010. A tutorial on graph-based SLAM. *IEEE Intell. Transp.*

Syst. Mag., **2**(4):31-43. http://dx.doi.org/10.1109/MITS.2010.939925

Hartley, R.I., Zisserman, A., 2004. Multiple View Geometry in Computer Vision. Cambridge University Press.

Held, D., Levinson, J., Thrun, S., *et al.*, 2016. Robust real-time tracking combining 3D shape, color, and motion. *Int. J. Robot. Res.*, **35**(1-3):30-49. http://dx.doi.org/10.1177/0278364915593399

Hillel, A.B., Lerner, R., Levi, D., *et al.*, 2014. Recent progress in road and lane detection: a survey. *Mach. Vis. Appl.*, **25**(3):727-745. http://dx.doi.org/10.1007/s00138-011-0404-2

Hoiem, D., Hays, J., Xiao, J.X., *et al.*, 2015. Guest editorial: scene understanding. *Int. J. Comput. Vis.*, **112**(2):131-132. http://dx.doi.org/10.1007/s11263-015-0807-z

Konolige, K., Marder-Eppstein, E., Marthi, B., 2011. Navigation in hybrid metric-topological maps. IEEE Int. Conf. on Robotics and Automation, p.3041-3047. http://dx.doi.org/10.1109/ICRA.2011.5980074

Li, Q., Zheng, N.N., Cheng, H., 2004. Springrobot: a prototype autonomous vehicle and its algorithms for lane detection. *IEEE Trans. Intell. Transp. Syst.*, **5**(4):300-308. http://dx.doi.org/10.1109/TITS.2004.838220

Mertz, C., Navarro-Serment, L.E., MacLachlan, R.A., *et al.*, 2013. Moving object detection with laser scanners. *J. Field Robot.*, **30**(1):17-43. http://dx.doi.org/10.1002/rob.21430

Montemerlo, M., Thrun, S., Koller, D., *et al.*, 2002. FastSLAM: a factored solution to the simultaneous localization and mapping problem. 8th National Conf. on Artificial Intelligence, p.593-598.

Pan, Y.H., 2016. Heading toward artificial intelligence 2.0. *Engineering*, **2**(4):409-413. http://dx.doi.org/10.1016/J.ENG.2016.04.018

Schueler, K., Weiherer, T., Bouzouraa, E., *et al.*, 2012. 360 degree multi sensor fusion for static and dynamic obstacles. IEEE Intelligent Vehicles Symp., p.692-697. http://dx.doi.org/10.1109/IVS.2012.6232253

Thrun, S., Leonard, J.J., 2008. Simultaneous localization and mapping. Int. Conf. on Artificial Intelligence, p.871-889. http://dx.doi.org/10.1007/978-3-540-30301-5_38

Ulrich, L., 2016. 2016's Top Ten Tech Cars. http://spectrum.ieee.org/transportation/advanced-cars/2016s-top-ten-tech-cars

Xue, J., Zheng, N.N., Geng, J., *et al.*, 2008. Tracking multiple visual targets via particle-based belief propagation. *IEEE Trans. Syst. Man Cybern. B*, **38**(1):196-209. http://dx.doi.org/10.1109/TSMCB.2007.910533

Zhang, Z., 2000. A flexible new technique for camera calibration. *IEEE Trans. Patt. Anal. Mach. Intell.*, **22**(11):1330-1334. http://dx.doi.org/10.1109/34.888718

Zheng, N.N., Liu, Z.Y., Ren, P.J., *et al.*, 2017. Hybrid-augmented intelligence: collaboration and cognition. *Front. Inform. Technol. Electron. Eng.*, in press. http://dx.doi.org/10.1631/FITEE.1700053