# Obstacle Detection and Tracking for the Urban Challenge

Michael S. Darms, Paul E. Rybski, Christopher Baker, and Chris Urmson

***Abstract*—This paper describes the obstacle detection and tracking algorithms developed for Boss, which is Carnegie Mellon University 's winning entry in the 2007 DARPA Urban Challenge. We describe the tracking subsystem and show how it functions in the context of the larger perception system. The tracking subsystem gives the robot the ability to understand complex scenarios of urban driving to safely operate in the proximity of other vehicles. The tracking system fuses sensor data from more than a dozen sensors with additional information about the environment to generate a coherent situational model. A novel multiple-model approach is used to track the objects based on the quality of the sensor data. Finally, the architecture of the tracking subsystem explicitly abstracts each of the levels of processing. The subsystem can easily be extended by adding new sensors and validation algorithms.**

***Index Terms*—Object tracking, obstacle classification, obstacle detection, situational reasoning, system architecture, Tartan Racing.**

## I. Introduction

AUTONOMOUS vehicles that operate in urban environments must contend with a number of challenging perceptual problems. These include, but are not limited to, the detection and tracking of roads; the detection and avoidance of static obstacles; and the detection, tracking, and prediction of other moving objects. The tracking requirements for autonomous vehicle driving in urban settings are arguably more complex than those for other scenarios, such as highway driving or driving through the desert [5]. In urban settings, vehicle speeds are slower than highway driving, but an autonomous vehicle must contend with situations, such as stop-and-go traffic, queueing at traffic signals, merging into and out of moving traffic, and following precedence rules at intersections.

The DARPA Urban Challenge was an autonomous vehicle race held on November 3, 2007, that took place in an urban environment.[1] Successful completion of three different missions culminated in an autonomous run of 60 mi in under 6 h.


Fig. 1. Team Tartan Racing's vehicle "Boss" participating in the 2007 DARPA Urban Challenge.

Driving scenarios that the urban challenge vehicles had to contend with included following and passing other vehicles, safely passing a vehicle against oncoming traffic, obeying the rules of precedence at intersections, and navigating through parking lots. These different scenarios meant that autonomous vehicles had to be able to detect and track other cars coming from any direction, and it was necessary to differentiate between static obstacles (such as cones, walls, and stopped vehicles) and active vehicles that may momentarily stop (such as at an intersection).

This paper describes the vehicle detection and tracking system used by "Boss" (named after Charles F. "Boss" Kettering), which was the autonomous vehicle built by Team Tartan Racing, which won the Urban Challenge in 2007. Boss, as shown in Fig. 1, collected data from more than 13 different environmental sensors and fused that information into a coherent set of dynamic obstacle hypotheses. Because each sensor type [RAdio Detection And Ranging (RADAR) or LIght Detection And Ranging (LIDAR)] had different return and noise characteristics, no single model could be used across all sensors to describe nearby vehicles. Instead, Boss used multiple models and a novel adaptive model switching mechanism to address the challenges of tracking vehicles in urban settings. Section II provides an overview of related vehicle-tracking research. The perception system used on our vehicle and how it models the world is described in Section III. Our tracking system and the specific contributions of this paper are described in Section IV. The performance of the system and examples of how it was used during the Urban Challenge are shown in Section V. This paper is summarized and concluded in Section VI.

[1] http://www.darpa.mil/GRANDCHALLENGE/.

Manuscript received March 6, 2008; revised June 18, 2008. First published April 24, 2009; current version published September 1, 2009. This work was supported in part by General Motors, by Caterpillar, by Continental, and by the Defense Advanced Research Projects Agency under Contract HR0011-06-C-0142. The Associate Editor for this paper was C. Stiller.

M. S. Darms is with the Department of Advanced Engineering, Division Chassis and Safety, Continental Automotive Group, 88131 Lindau, Germany (e-mail: Michael.Darms@continental-corporation.com).

P. E. Rybski and C. Baker are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: prybski@cs.cmu.edu; cbaker@andrew.cmu.edu).

C. Urmson is with the Field Robotics Center, Carnegie Mellon University, Pittsburgh, PA 15217 USA (e-mail: curmson@ri.cmu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2009.2018319

TABLE I
SENSOR SETUP AND CHARACTERISTICS FOR OBJECT TRACKING

| | Sensor | Sensor Type | Max. Range* | Vertical Angle | Horiz. Angle | Returned Feature |
|---|---|---|---|---|---|---|
| **1** | Continental ARS 300 | Scanning Radar (near/far) | 60/200m | $4.3^o$ | $56^o/18^o$ | 2D position, 2D velocity |
| **2** | Continental ISF172 | Fixed Beam Laser | 150m | $4^o$ | $14^o$ | 2D position |
| **3** | SICK LMS 291 | Scanning Laser, 1 level | 80m | $0.25^o$ | $180^o$ | Edge Target, 2D position |
| **4** | IBEO AlascaXT | Scanning Laser, 4 level | 200m | $3.2^o$ | $240^o$ | Edge Target, 2D position |
| **5** | Velodyne HDL-64E | Scanning Laser, 64 beams | 120m | $26.8^o$ | $360^o$ | Edge Target, 2D position Obstacle Maps |

## II. RELATED WORK

Most existing commercial driver-assistance systems with environmental perception are designed for longitudinal traffic in well-structured environments (e.g., adaptive cruise control [6]). Driver-assistance systems, which work in more complex environments and use a multisensor fusion approach, are an active area of research. Examples include systems for intersection assistance or systems that assist drivers in construction sites on a highway [7].

Research in vehicle tracking has been accomplished using a wide variety of different sensors that include, but are not limited to, computer vision [8], [9], LIDAR [10], [11], and RADAR [12], [13]. In our approach, we make use of and fuse information from our vehicle's onboard LIDAR and RADAR sensors to track objects around the vehicle. This fusion is necessary to provide the proper blend of long-range detection with short-range shape estimation. In particular, one of our sensor modules for the SICK LMS 291 (see Table I) uses a vehicle shape-interpretation algorithm that was originally used for vehicle collision warnings for buses [14].

To robustly track vehicles, even in the face of sensor and environment noise, we use a Bayesian-filtering approach [15]. Object tracking using a Bayesian filter formalism relies on an *a priori* model of the object's motion that allows the algorithm to predict the object motion, given noisy observations. One of the most widely used methods for state estimation is the Kalman filter [16], [17], in which the system model is assumed to be linear and the noise is assumed to be Gaussian. Our system uses a more numerically stable variant of the Kalman filter called the square root filter [18].

In our application, the autonomous vehicle must interpret many different sensor returns, potentially from multiple targets, and continually decide which new sensor reading corresponds to an existing tracked object [19], [20]. We make use of an approach known as multiple hypothesis tracking [21], [22], where multiple independent state estimators are used to estimate a multimodal probability density. At each step of the estimation process, new sensor data are associated with the multiple tracks to determine whether the sensor reading corresponds to an existing target or whether a new target should be instantiated in the tracker. Old targets that do not receive additional sensor information to support them will eventually be pruned from the tracked list.

Our approach to vehicle tracking makes use of multiple vehicle models, depending on the kind and quality of information obtained about the object from the vehicle's sensors. A traditional mechanism for incorporating multiple models in tracking is the interacting multiple model filter [23], which uses a weighted mixture of different process models. A more general approach is the switching Kalman filter model [24], which represents multiple independent system state dynamics models and switches between them (or linearly combines them) to best fit the observed (or predicted) nonlinear dynamics of the system being modeled. In our work, model switching is done at a higher level than where the vehicle state is estimated. We propose a novel voting scheme where a tracked object will be represented by the most informative model, depending on the quality of the sensors.

An important and cost-determining part in the process of designing the perception system is the development of the system architecture [25], [26]. Multiple publications that describe general architecture concepts for sensor data-fusion systems exist [20], [25], [27]–[29]. The architecture that we use for object tracking was originally developed for driver-assistance systems [30] and is designed for an efficient development process, allowing a cost-effective extension of the system with new sensors.

## III. PERCEPTION SYSTEM

In our system, vehicle tracking is an integral part of a larger perceptual system, which is responsible for providing a coherent and consistent model of the world. The process of detecting and tracking vehicles is directly influenced by the context in which those vehicles are observed. Fig. 2 shows the flow of how a world model is computed. Raw measurements received by the individual sensors are processed in the measurement phase to be passed to the perception phase, where features and object interpretations are computed. These object hypotheses are passed to the understanding phase, where the larger context in which the object is detected is used to generate an understanding about the global situation surrounding that object.

### A. Modeling the Environment

To properly define the robot's model of the world, we divide the world into three specific domains.

1) *Road structure:* The road structure is a logical interpretation of the environment and defines where and how vehicles are allowed to drive.

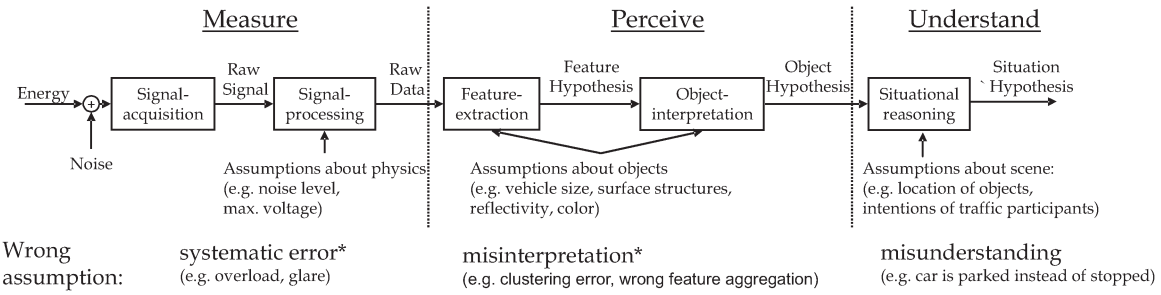


Fig. 2. General perception process that illustrates how raw energy detected from the environment is converted into a situational understanding of the world around the vehicle. This figure also illustrates where noise and errors can be introduced into the system and the modeling assumptions that specify how the information is altered at each step. (Note that the categories of errors marked with $*$ are generally referred to throughout this paper as *artifacts*.)

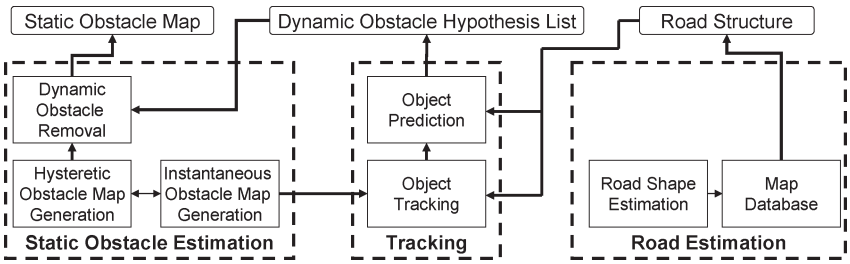


Fig. 3. Perception architecture (see [1]).

2) *Static obstacles:* Static obstacles are obstacles that are assumed not to move during the observation period (on or off road).
3) *Dynamic obstacles:* Dynamic obstacles are obstacles that potentially move during the observation period. For the Urban Challenge, only cars fall into this category. Note, however, that a car does not necessarily have to be a dynamic obstacle, as it can also be a parked vehicle that will never move.

### B. Perception System Architecture

The architecture of the perception system, as shown in Fig. 3, is analogously divided to the world model into the following three subsystems: 1) a road estimation subsystem, which generates information about the road structure; 2) a tracking subsystem, which is responsible for generating dynamic obstacle hypotheses; and 3) a static obstacle estimation subsystem, which estimates the location of static obstacles.

*1) Road Structure:* The road structure is represented as a topological network of segments, intersections, and zones. A segment contains a number of road lanes, and each lane has a specified width and direction. The shape and curvature of a particular lane are determined by a set of points that are spaced roughly 1–2 m apart from each other. Intersections are junctions that explicitly connect lanes from different segments. Zones are free-form open areas, such as parking lots, which have no explicit restrictions on where vehicles can travel.

In our system, the road structure was derived from the given road network definition file and information from a high-precision satellite image. With this, the data were known with high confidence. The information was used inside the tracking system for computational efficiency and to reduce the number of sensor artifacts (see Section IV). The road-estimation system, however, also included a road shape-estimation algorithm, which could deliver information about the road in cases where the map was not sufficient. This, however, was not necessary on race day.

*2) Dynamic Obstacle Hypothesis List:* The classification of an obstacle as dynamic obstacle requires scene understanding (e.g., to distinguish a parked from a temporarily stopped vehicle). The perception system only provides a list of dynamic obstacle hypotheses. These are all obstacles that potentially belong to the class of dynamic obstacles. Dynamic obstacles are represented by a shape model and state variables, such as position, velocity, and acceleration (see Section IV-A).

For every dynamic obstacle hypothesis, two flags are provided as follows: 1) the current movement state, i.e., *moving* and *not moving*, and 2) the movement history, i.e., *observed moving* and *not observed moving*. The flag *moving* is set once the tracking subsystem decides that the object is currently in motion. The flag *observed moving* is set once the tracking system decides that the object has changed its position. With this definition of a dynamic obstacle, it is obvious that, whenever the flags *moving* and *observed moving* are set, the obstacle hypothesis belongs to the class of dynamic obstacles. If only the flag *observed moving* is set, the obstacle may belong to the class of static obstacles (e.g., vehicle stalled). However, in our system, all objects that have only the *observed moving* flag set are directly treated as dynamic obstacles. Testing showed that this is a good approximation for short observation periods.

In certain situations, sensors cannot detect an object. This holds true, for example, if an object is not within the field of view of a sensor or part of the field of view is occluded. (Sensor occlusions were not modeled within our system.) However, this can also occur due to sensor artifacts. Dynamic obstacle hypotheses are only maintained by the tracking system as long as the sensor data can support the estimation of state variables.

This means that only short-term sensor artifacts ("measurement dropouts") are bridged and that no scene understanding is used to argue the lifetime of an obstacle hypothesis. Situations where an object is temporarily lost for a longer amount of time are explicitly handled by the situation assessment algorithms (see Section V).

*3) Static Obstacle Map and Instantaneous Obstacle Map:* Static obstacles are represented in a map decomposed into a regular array of cells. Within this map, dynamic obstacle hypotheses that are flagged as *observed moving* clear out any cells within their footprint. The static obstacle map has a memory, remembering obstacles that are no longer visible.

Evidence of obstacles is calculated as a function of the number of LIDAR points classified as an obstacle in each cell. Evidence decays over time at a rate that is less than the rate at which it accumulates for a reasonably supported obstacle. In this way, the obstacle map is filtered to reduce the effect of spurious sensor measurements.

The perception system also maintains an instantaneous obstacle map. The map contains all obstacles around the robot; dynamic obstacle hypotheses are not removed from the map. The map is used to validate features from planar LIDAR and RADAR obstacles in the tracking subsystem (see Section IV). For this purpose, the information does not have to be as accurate as that for the motion-planning algorithms, which use the static map. A much smaller time constant can be used to filter the sensor input data.

### C. Sensor Setup

Table I shows the autonomous vehicle's sensor configuration used for tracking moving objects and summarizes the characteristics of the sensors [3]. Data are asynchronously collected from each sensor but is timestamped to a common system time. The combination of these sensors provides complete sensor coverage around the vehicle. As will be described, the fusion of the different sensors allows Boss to exploit the advantages of each sensor type to increase overall performance. The redundancy provided by overlapping fields of view enables the system to be robust against false readings or failure of sensors. The sensors are configured to maximize redundancy in front of the vehicle (with between four and seven sensors providing coverage) while also providing redundant sensing around as much of the vehicle as possible (e.g., two sensors, i.e., a LIDAR and a RADAR, provide coverage behind Boss).

Radars are used for detecting objects in the near and far ranges. Doppler shift provides a measure of the relative speed of an obstacle, which gives a low latency and an accurate velocity estimate, which, in turn, can be used to efficiently distinguish static from moving objects. The measurement accuracy of these sensors is sufficient in associating vehicles to lanes on the road up to a detection range of 200 m. While long-range and direct velocity measurement are clear advantages, RADAR sensors provide no notion of the height of the obstacles; instead, they report RADAR cross section. It is common that small angular objects can return significant amounts of energy, comparable with a vehicle. Without complementary sensing, it can be difficult to distinguish between irrelevant clutter on the road and relevant vehicles.

Planar scanning lasers provide information about the geometric cross section of an object. While this information is generally useful, a single cross section can be insufficient in distinguishing between an obstacle and a benign terrain feature. For example, the side of a car may provide a contour that looks very similar to a sloped road. Furthermore, given a fixed angular resolution, the sensors provide sparse spatial information and, thus, insufficient geometric information beyond some range. This motivates the use of a dual-mode representation (bounding boxes and points, as will be shown next).

The Velodyne HDL-64E was the only sensor with a wide-enough vertical sensing angle to deliver reliable 3-D information about objects around the vehicle. However, the effective range of the sensor is not sufficient for all the required autonomous maneuvers, particularly merging and passing maneuvers at 30 mi/h. Despite these shortcomings, the sensor is important for object validation in the near range as the 3-D data allow a ground plane estimation and the creation of the instantaneous obstacle map. In this way, the sensor can be used to effectively reject artifacts caused by the limited vertical field of view of other sensors (e.g., false obstacles generated by perceiving the ground).

The last class of sensors used on the vehicle are fixed-beam LIDAR sensors. Similar to the RADAR sensors, they provide 2-D coordinates of the detection center of obstacles; however, there is no direct measurement of velocity. A fixed-beam LIDAR is mounted redundant with a RADAR on a rotating mount on both sides of the vehicle. By appropriately pointing these sensors, it is possible to situationally focus and extend the vehicle's perception capability.

It is clear that not one of the sensors is sufficient in providing the situational awareness that is necessary for safe urban driving. The variety of sensors provides not only redundant coverage but complementary detection and processing characteristics that provide significant benefits beyond the use of a single sensor as well.

## IV. Tracking Subsystem

The tracking subsystem was responsible for the identification of dynamic obstacle hypotheses for use by the rest of the system. Its design was influenced by both the available sensors and the ways that the obstacle hypotheses are used to implement various traffic-interactive behaviors. The robust implementation of each of these behaviors requires complex situation-specific reasoning that, if applied to the tracking subsystem, would lead to a rigid system that is difficult to adapt or extend to implement new behaviors. Instead, all possible vehicles are tracked the same way, regardless of the scenario, and situation-specific reasoning is isolated in another subsystem.

It is important to note that there was no attempt to do vehicle *classification* in the tracking subsystem. Traditional classification algorithms [31], [32] are generally based on camera data and rely either on heuristics or training data to accurately classify vehicles in the environment. Both approaches work well in relatively constrained scenarios such as highway driving, but they break down somewhat in urban environments, where traffic can come from arbitrary directions.

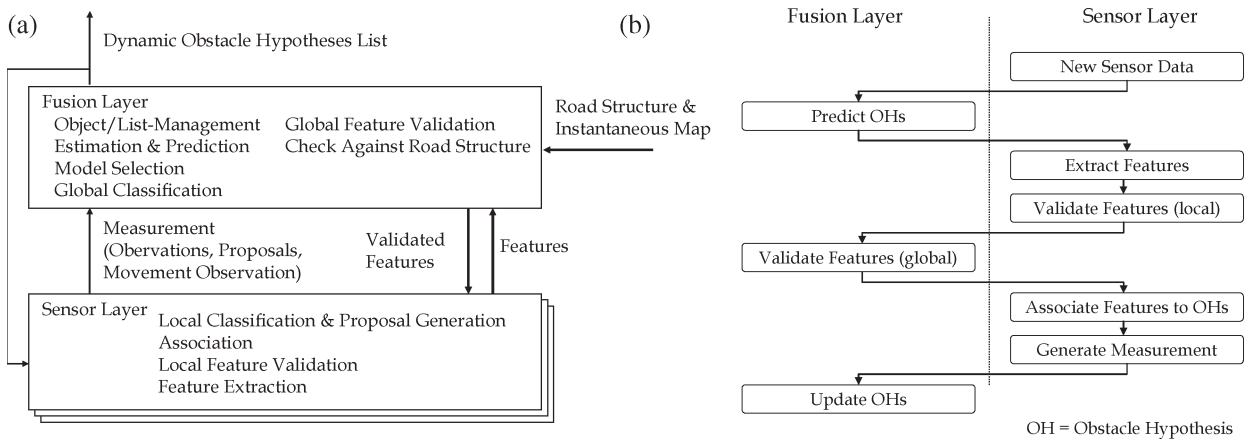


Fig. 4. Tracking subsystem architecture. (a) Two layers that comprise the tracking subsystem. (b) Processing flow between the two layers.

### *A. Modeling Dynamic Obstacles*

An extended Kalman filter [16] is used to estimate the state variables of dynamic obstacle hypotheses. The data available for tracking depend on the type of sensor (e.g., radar versus scanning laser) and the position of an observed object in the sensor's field of view (e.g., increased sensor noise and/or decreased resolution at longer ranges).

To cope with these issues, we use an adaptive model-switching approach [3], wherein the tracking model is selected from different discrete models based on the available sensor information. The tracking subsystem uses the following two models: 1) a box model and 2) a point model. Note, however, that the approach can be extended as necessary to include other models.

The box model describes a dynamic obstacle hypothesis as a rectangular box with a fixed length and width; its dynamics are governed by a simple bicycle model with constant noise parameters, which is similar to that in [33]. The point model, on the other hand, does not include any shape information; its dynamics are governed by a constant-acceleration model with adaptive noise, which is similar to that in [20].

### *B. Architecture*

To support the use of multiple heterogeneous sensors, the tracking subsystem is divided into two layers, as shown in Fig. 4. The Sensor Layer is meant to hide sensor-specific details, providing sensor-independent data to the Fusion Layer, which is responsible for forming and maintaining dynamic obstacle hypotheses. For each type of sensor, a unique sensor-layer module is implemented, and a separate instance of that module is used for each physical device on the robot. Sections IV-C and D discuss the modules for scanning lasers and radar sensors, respectively.

Every time a sensor generates a new measurement, its corresponding Sensor Layer instance first requests a prediction of the current set of dynamic obstacle hypotheses from the Fusion Layer. These predictions are cached for possible use in feature extraction and validation and are eventually used to associate validated features to existing hypotheses. Features are extracted from the raw sensor data according to the algorithm encapsulated within the Sensor Layer module, which searches for features that may be used to describe a dynamic obstacle hypothesis according to either the point model or the box model.

Thereafter, all extracted features go through a two-step validation process to reduce the number of false positives in the extracted feature set. These may be caused by issues, such as ground detection by planar laser scanners, where the intersection of the scanning plane with the ground plane can be misinterpreted as the edge of a vehicle (see, e.g., [4] and [34]). All features that are explicitly validated in at least one of these two steps are used to generate dynamic obstacle hypotheses, reducing the chances of a false negative.

In the first validation step, a sensor-specific validation algorithm that exploits the known properties of the sensor is used to filter out commonly encountered artifacts and to immediately validate particularly strong features, such as when a radar sensor indicates a strong velocity return, as discussed in Section IV-D. In the second validation step, the feature validation interface provided by the Fusion Layer uses the road structure to reject features that are too far away from the road. This reduces the required computation time that is necessary to process the data and can reduce the number of artifacts caused by objects that are close to roads, such as bushes or roadside barriers. Then, the instantaneous obstacle map is used to check whether the remaining features are likely to be caused by a ground return; within the range of the map, features are only validated if they are also found in the map.

All valid features are subsequently used in the next step in the Sensor Layer, i.e., data association. Here, the system attempts to associate the features with the currently predicted dynamic obstacle hypotheses from the Fusion Layer. Sensor-specific heuristics are used for data association. These are designed according to the characteristics of the features extracted from a given sensor. Features that cannot be associated to an existing hypothesis are used to generate new proposals, as described here.

An *observation* is generated for each feature associated with a given hypothesis, encapsulating all information that is necessary to update the estimation of dynamic obstacles in the Fusion Level. This includes the nonlinear equations for mapping the measurements from measurement space to state space, the corresponding Jacobian matrices, and the measurement vector for the Kalman filter. This approach is the key to decoupling

the estimation algorithms in the Fusion Layer from the sensor-dependent details in the Sensor Layer.

Similarly, a *movement observation* is generated per associated feature, indicating whether a given obstacle hypothesis is believed to be moving purely based on sensor information. The movement observation will be one of the scenarios given here.

1) *Movement confirmation*, which means that, from the sensor's point of view, the associated obstacle hypothesis is currently moving;
2) *No movement confirmation*, which means that it is currently not moving. The information is accompanied by a *no movement vector*, which points to the direction in which the sensor detected no movement;
3) *No information available*, which indicates that this sensor cannot provide any information regarding the motion of the feature.

A Sensor Layer instance can also provide advice as to which tracking model should be used for a given feature. These are provided to the Fusion Layer as a *proposal set*, i.e., a set (ordered by quality) of new object hypotheses that can explain the extracted feature. If features are associated to an existing obstacle hypothesis, a proposal can be used to suggest an alternative to the current tracking model (point or box). A sensor-independent voting algorithm (see Section IV-E) is used to determine the best model. For features that are not associated to an existing hypothesis, the proposals are used to initialize a new dynamic object hypothesis. When the according proposals are located on a road, the road structure is used to bias the selection of the best proposal from the set. In parking zones, the proposal ranked with the highest quality is selected. Finally, if no model switch took place, the observations are used to update the state estimate, and the movement properties of each hypothesis are evaluated (see Section IV-F).

The core of the tracking system run on a 2.16-GHz Intel processor. Only the feature extraction algorithms were distributed over other processors or directly performed in sensor-specific electronic control units. The latency of the system is dependent on the number of features that enter the fusion. The typical latency was on the order of 150 ms from the measurement time to the output of an object hypothesis list. The maximum latency was fixed to 300 ms. The data of the sensors were dropped if the latency could not be achieved.

The architecture is robust to sensor failures. If the sensor goes into a fail silent mode in such cases, the fusion system will seamlessly continue to work without its information. The output will, of course, be of reduced quality, depending on which sensor is no longer available.

### C. Sensor Layer: Planar Laser Scanner

As with all sensor modules, processing begins with feature extraction. For a planar laser scanner, *edge target* features are extracted from the laser scanner raw data as in [35]. An edge target consists of either a single edge or two edges joined by a nearly 90° angle, which are assumed to represent either a single side or a corner of a vehicle, respectively. Fig. 5 shows sample edge targets, with artifacts visible due to ground detections and noise in the sensor data. There are no sensor-specific properties

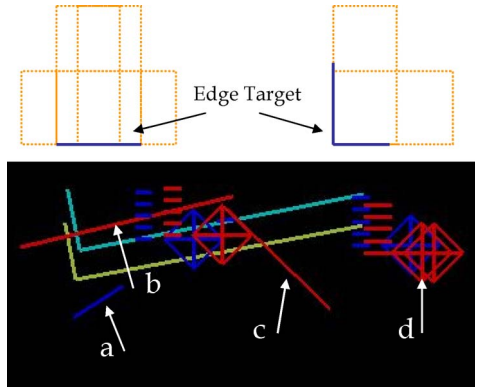


Fig. 5. (Top) Possible box model interpretations of edge targets. (Bottom) Snapshot of a vehicle (bird's-eye view). (Edge target) Laser scanner features. (Diamond) Radar features. (White arrow) Artifacts. (a) and (c) Ground detections. (b) Cluster misinterpretation. (d) Dust. Both laser scanners detected different edges on the vehicle.

that may be exploited for validation; therefore, only the sensor-independent facility provided by the Fusion Layer is used to validate edge targets.

For data association, a set of possible edge targets is generated from the predicted dynamic obstacle hypotheses, i.e., four for each box model and one for each point model. The extracted edges are first compared with the predicted edges for compatibility using a heuristic based on the angle between them. Edges that pass the first heuristic are then associated using a nearest-neighbor approach that minimizes the distance of the corner points.

If an extracted edge target is associated to a predicted point model, then a corresponding observation is set up, and a heuristic is used to decide whether a box model could be used to better explain the extracted feature. If this is the case, then a proposal set is generated, including all possible interpretations of the edge target as the box model, as shown in Fig. 5. The proposed box models are generated according to a heuristic that accounts for the field of view, the sensor's resolution, and the viewing angle, and they are ordered according to a quality measure that reflects the relative confidence in each interpretation.

If an extracted edge target is associated to a predicted box model, all possible interpretations of the edge target as the box model are generated in the same manner. The interpretation that best fits the prediction is chosen to generate the observation. If, however, none of the interpretations is close enough, the edge target is treated as an artifact, and no observation is generated. If there is an interpretation that explains the edge target with high quality and this is not the interpretation chosen to generate the observation, a new proposal is generated based on this interpretation. Finally, the edge target is checked for whether the point model makes more sense, such as when the distance from the sensor to the target is large, and the data can no longer be used to effectively estimate the yaw angle. In this case, a point model proposal is generated and added to the alternate proposal set for the predicted obstacle hypothesis.

Finally, if an extracted edge target is not associated, a heuristic checks if it could be an artifact caused by an already existing object hypothesis. A typical example is an edge target that is located inside a vehicle, such as an edge target on a windshield, instead of on the outer body. For each extracted edge target that

cannot be explained as an artifact of an existing hypothesis, a proposal set is analogously generated similar to the process of alternate proposal selection that was previously described. Edge targets do not include a direct measurement of velocity, and therefore, all movement observations that are associated with the hypotheses or proposals are of the form "no data available."

### D. Sensor Layer: Radar Sensor

The features extracted from raw radar data are called *point targets*. They represent the center of the reflected radar energy and include the relative velocity of the target, as measured by the Doppler shift in the signal. The extraction algorithm is optimized to extract reflections from vehicles but is also susceptible to spurious returns from features, such as manhole covers and metallic road signs.

Fig. 5 includes sample point targets, where, in this case, the highlighted artifacts originate from dust. Any feature with a substantially nonzero absolute velocity that was measured by adding the Doppler measurement to Boss's velocity is immediately validated, as it can be assumed that it originated from a moving object. All features that are not validated by their velocity are checked in the same way as the edge targets via the generic validation interface provided by the Fusion Layer.

For the association step, a set of possible point targets is generated from the predicted dynamic obstacle hypotheses. For a point model, a single-point target is generated. For a box model, a point target is generated for each corner and for the center of each edge. The measured point targets are then associated to the predicted point targets using a nearest-neighbor approach. In contrast to the data association done for scanning laser sensors, multiple extracted point targets may be associated with a single predicted hypothesis. Note, however, that each extracted point target is still associated with, at most, one predicted target, and *vice versa*.

For each associated pair of targets, a heuristic checks whether the predicted velocity and the measured velocity are compatible. If this is the case, an observation is generated. If not, a new point model proposal for the associated obstacle hypothesis is set up. If the associated obstacle hypothesis is a box model, then a point model proposal is set up, regardless of the compatibility, as the sensor cannot support the box model. The remaining unassociated point targets are used to generate new point model proposals. Similar to the laser scanner Sensor Layer, a heuristic checks beforehand if the unassociated target could be an artifact caused by an existing object. Movement observations are generated according to the magnitude of and the confidence in the Doppler velocity measurement.

### E. Best Model Selection

Based on the proposals and observations provided by the Sensor Layer, the Fusion Layer selects the best tracking model (see [3]) for each moving obstacle hypothesis. As with all algorithms in the Fusion Layer, the algorithm is formulated to be sensor independent.

A sensor *supports* a model if the model is observable with observations from that particular sensor only. The laser scanners (planar and 3-D) support both the box and point models, whereas the radar and fixed-beam laser sensors only support the point model. A model counts as *currently supported* by a sensor if it is either the current model and the sensor provides a consistent observation, or it is a model *proposed* by a sensor as an alternative to the current model. In the case of the proposed models, a minimum number of consecutive cycles with identical proposals are required before a proposal is considered in the model-selection process. This increases the system's robustness to spurious misinterpretations.

The best model is selected from the currently supported models according to preference and confidence; the box model is generally preferred over the point model, and confidence is determined by a voting scheme, as will be discussed next. (Only sensors that are currently detecting the object are considered.)

First, the *relative support* for each possible model is determined as the ratio of the number of sensors that are currently supporting the model to the total number of sensors that are supporting the model in principle. A model switch is considered only when the relative support for at least one model is above a configurable minimum support threshold; otherwise, the current model is retained from the previous cycle. Higher threshold values ensure that a switch to a model with higher accuracy will be performed only if there are enough sensors that are currently supporting it. In the Tartan Racing system, the number of sensors supporting the box model increases as the vehicle gets closer to the robot. For example, at a range that is closer than 30 m, up to four sensors can support the box model, but at longer ranges, primarily radar sensors, which only support the point model, detect objects.

The best candidate model for the cycle is the most preferred model with relative support above the minimum threshold. If it differs from the current model, the model is immediately switched and initialized with values from the supporting proposal. If, on the other hand, it coincides with the current model, the current model is still evaluated to handle the case of a valid model with grossly invalid state variables. For example, consider the box model whose heading is orthogonal to the direction of travel. In this case, the ratio of sensors that are proposing alternate models to those that are providing observation of the current model is evaluated, and the model is reinitialized if this ratio exceeds a configurable threshold.

### F. Movement Classification

The Fusion Layer combines the movement observations from all sensors with statistical tests of the motion of the hypothesis to compute these properties.

*1) Moving Versus Not Moving:* The determination of whether an obstacle hypothesis is *moving* begins with an accumulating count of the number of times that the hypothesis has been reported by any sensor to be *confirmed moving*. Each individual movement observation increments this count, and when it exceeds a configurable threshold $th_{\mathrm{moving}}$, the hypothesis is considered to be *potentially moving*.

If the dynamic obstacle hypothesis is not classified as *potentially moving* based on the movement observations, a statistical test on the estimated velocity attempts to reject hypothesis $H_0$: "The absolute velocity is smaller than $v_{\mathrm{min}}$." If this is the case, then the obstacle is classified as *potentially moving*.

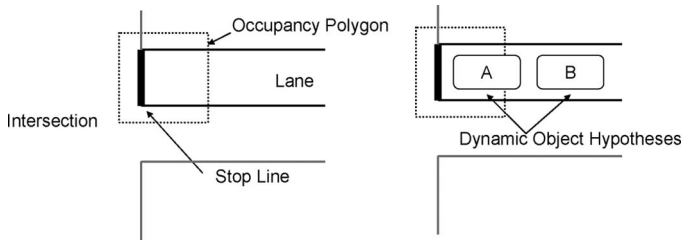


Fig. 6. Example of an occupancy polygon. Dynamic obstacle hypotheses A and B cause the polygon to be considered as occupied and unoccupied, respectively (see [2]).

This statistical test is entirely based on the state variables of the dynamic obstacle hypothesis; therefore, it may be performed independent of the sensors that detect the hypothesis. This test is necessary in situations where none of the detecting sensors can provide a direct movement confirmation, such as when the hypothesis is exclusively detected by laser scanners.

The significance level $\alpha$ for rejecting hypothesis $H_0$ and $v_{\min}$ can be tuned, with a higher $v_{\min}$ and a lower $\alpha$ making more conservative estimates. Lowering $\alpha$ leads to longer times until enough data are available to reject $H_0$, delaying the identification of a *moving* obstacle hypothesis. Similarly, increasing $v_{\min}$ leads to less false positives at low speeds, trading off the ability to properly classify slow-moving objects. In terms of hardware, the test can be influenced by the increase in the number of sensors or the increased accuracy of a single sensor.

If a dynamic obstacle hypothesis is classified as *potentially moving*, the system performs a cross check against the no-movement vectors provided by the Sensor Layers. The dot product between the normalized current velocity (unit vector) estimate and the no-movement vectors (unit vectors) is checked against a configurable threshold. If this check indicates a strong inconsistency between the estimated velocity vector and any no-movement vector, the obstacle is classified as *not moving*, providing the opportunity for a sensor with information about confinement in a particular direction, such as a RADAR sensor, to veto the *potentially moving* state. Otherwise, all remaining *potentially moving* hypotheses are marked as *moving*.

*2) Observed Moving Versus Not Observed Moving:* The determination of whether a dynamic obstacle hypothesis has been *observed moving* begins with an evaluation of the distance that it has traveled since it was first classified as *not observed moving*. If this distance is above a configurable threshold and the hypothesis would be classified as *moving* solely based on the movement observations from the Sensor Layer, then the obstacle is classified as *observed moving*. If it is not directly supported by movement observations, then the obstacle hypothesis must be continuously classified as *moving* for a time period $t_{\min,1}^{\text{obm}}$ to be classified as *observed moving*.

The check against the distance traveled increases the robustness against short-term artifacts, particularly during the initialization phase of the filters. It is possible to skip this check for objects that are confirmed to be moving by the sensors alone; however, this did not significantly increase the performance, and therefore, the team favored the more conservative approach.

The *observed moving* property is immediately cleared if the object is not classified as *moving* during the time period leading up to a second threshold $t_{\min,2}^{\text{obm}} > t_{\min,1}^{\text{obm}}$. Once this second threshold is reached, the *observed moving* property is retained for a time period $t_{\max}^{\text{obm}}$ after the object is no longer *moving*. This accounts for traffic that briefly stops, such as when queueing on approach to an intersection, at the potential cost of retaining an invalid *observed moving* classification for up to $t_{\max}^{\text{obm}}$.

## V. Situational Reasoning

In this section, we describe two different scenarios that were commonly encountered during the Urban Challenge (see also [2]). First, we discuss intersection handling, where Boss must determine whether to enter an intersection or to wait for other vehicles to proceed first. Then, we discuss distance keeping, where Boss tracks a vehicle in front of it to match its speed and maintain a minimum safety distance. Both demonstrate the application of situation-specific reasoning to the situation-independent dynamic obstacle hypotheses.

### A. Intersection Handling

As the robot approaches and waits at an intersection, the precedence order among vehicles at that intersection must be determined to identify the point in time when the robot may proceed. Other vehicles may be static on arrival or for long periods of time after arrival, and therefore, all dynamic obstacle hypotheses are used to determine the precedence order, regardless of their movement flags. This requires a situational treatment of false positives, such as a parked vehicle or other obstacles close to the intersection, and false negatives caused by measurement dropout or transient occlusion. These possible errors prevent the assignment of precedence to specific vehicles, and therefore, the robot instead uses an intersection-centric precedence estimation algorithm.

For each lane entering the upcoming intersection, an occupancy polygon is generated based on the road structure, encompassing the entry point and the area backward along that lane for some configurable distance, as shown in Fig. 6. A given polygon is considered "occupied" when the estimated front bumper of either Boss or any dynamic obstacle hypothesis is inside. For polygons in the "occupied" state, the system maintains two pieces of temporal data, i.e., the time of the first occupancy and the time of the most recent occupancy. The former is used in the determination of the precedence order, which is a simple matter of sorting the occupied polygons in ascending order by first occupancy. The latter is used to implement a temporal delay on when a polygon becomes "unoccupied," such that it must persistently be empty over some span of time to transition

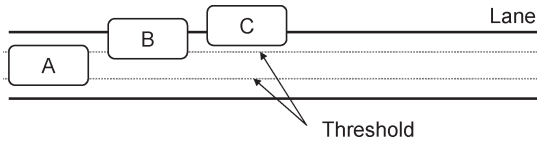


Fig. 7. Different interpretations of the dynamic obstacle hypotheses, depending on the position in a lane. All dynamic obstacle hypotheses would be taken into consideration for distance keeping if they have the *observed moving* property. If not, A and B would still be considered, but C would not (see [2]).

to the "unoccupied" state. This delay maintains the estimated precedence ordering when dynamic obstacle hypotheses briefly disappear or drift outside the polygon, providing robustness to transient tracking errors, such as measurement dropout or short-time occlusions caused by vehicles traveling through the intersection.

Beyond strict adherence to precedence at an intersection, the Urban Challenge rules also specified explicit treatment of stalled vehicles and misperceived precedence ordering, requiring the robot to proceed after 10 s of inactivity. The robust implementation of this special case also provides coverage of the potential false positives previously described. If an occupied polygon consecutively retains precedence for 10 s, it is subsequently ignored for the purpose of precedence estimation. This simultaneously implements the required treatment of stalled vehicles and discards any false-positive dynamic obstacle hypotheses at the cost of a 10-s delay. If the robot proceeds through an intersection under this premise, its speed and acceleration are reduced in an attempt to guarantee the safety of the maneuver. Testing showed that the probability of an obstacle close to an intersection triggering a false positive is comparatively low, and the difference in behavior, i.e., a 10-s delay, followed by movement at reduced speed, is acceptable from an outside perspective, basically reflecting cautious action by the robot.

### *B. Distance Keeping*

The Distance Keeping behavior is active while driving along a road behind a slower vehicle. It aims to preserve a minimum safety gap to the lead vehicle, eventually matching speed to maintain that gap in the steady state. The set of dynamic obstacle hypotheses is filtered according to the situation to limit the occurrence of spurious braking due to false positives, and the lead vehicle is selected as the closest of those that remain. First, all hypotheses overlapping the lane of travel with the *observed moving* property are included, reflecting a high confidence that each will be an active participant in local traffic. Hypotheses that do not have the *observed moving* property are only included if they are close enough to the center of the lane according to a threshold that depends on the local width off the road (see Fig. 7). The goal is to reject parked vehicles and other curbside obstacles while still considering vehicles stopped at the middle of the road in this situation.

Even with this filtering, it is still possible to interpret an invalid dynamic obstacle hypothesis as relevant for distance keeping. For example, a traffic cone or barricade at the center of the lane may generate a dynamic obstacle hypothesis that will be treated as a stopped vehicle. Similar to the requirement of overriding the precedence order at the intersection, the Urban Challenge rules required the robot to handle a stalled vehicle on the road by coming to a safe stop behind it, waiting to determine that it has stalled, and then safely circumventing the vehicle through an adjacent lane.

To provide robustness to sensor noise and measurement dropouts, the absolute position and velocity of the closest relevant obstacle hypothesis are retained over an adaptive span of time, which is similar to the time delay discussed in Section V-A. The obstacle's distance and velocity relative to the robot is considered in the computation of the time delay as an estimated time to collision. A dynamic obstacle hypothesis close to the robot with a small estimated time to collision is retained for a comparatively long span of time, reflecting an increased risk associated with transient measurement dropouts. Farther hypotheses with longer expected collision times represent lower risks to the robot and are retained for accordingly shorter durations.

In the case of hypotheses farther from the robot, the perception system heavily relies on long-range RADAR sensors as they are more effective at detecting distant objects than the LIDAR sensors that contribute to the static obstacle map, which is used to validate the data. However, RADARs are also sensitive to metal objects on the ground, such as a manhole cover, which can safely be driven over by the robot. If such an object is detected with the RADAR sensors at long range, the robot slightly slows down until the object enters the validation area of the instantaneous obstacle map. Then, the RADAR feature will be invalidated, and the associated dynamic obstacle hypothesis will be removed from the list (see also [4]). Since the object is still far away from the robot, the associated risk of collision is low, the time span for retaining it as the relevant obstacle is small, and the robot is allowed to accelerate again after a comparatively short delay.

From an outside perspective, this is again perceived as acceptably cautious behavior, and adjusting parameters for the retention delay allows for selection between cautious and aggressive driving. Longer delays cause the robot to very conservatively react to false positives at long ranges, and high-speed autonomous driving is often not possible. With shorter delays, the robot more readily discards long-range readings but can often be forced to perform abrupt braking maneuvers when approaching real vehicles.

## VI. Conclusion and Future Work

In this paper, we have presented the obstacle detection and tracking system of the autonomous robot "Boss," which is the Team Tartan Racing's winning entry into the 2007 DARPA Urban Challenge. This system was successfully used in a number of different situations; intersections, distance keeping, and parking lots were successfully handled. We have provided illustrative examples in this paper of a few of these situations to describe the process by which the objects were tracked.

One issue that had to be addressed by the tracking subsystem was the need to distinguish between moving and nonmoving obstacles. A sensor-independent algorithm that combines sensor-specific movement observations and a sensor-independent hypothesis test into a robust classification of the movement state of an object has been presented.

Another issue was that no one sensor could provide all of the information necessary to track the vehicles around Boss. Intelligent mechanisms for fusing the data from RADAR and LIDAR were critical for the successful operation of the whole system. In the fusion process, we proposed an adaptive model switching approach where the tracking model for an object is selected based on the available sensor information.

The vehicle-tracking system takes the raw data returned from the sensors and interprets them at multiple layers. The final interpretation is left to the robot's higher level decision processes, which use the situational context to understand the scene and form a situation hypothesis. We found that this greatly improved the performance of the overall system since the tracking system stays independent of the situational context and the decision algorithms of the robot can be adapted to the quality of the perception data and the specific situation.

The general tracking architecture that separates the Sensor Layer from the Fusion Layer allowed individual new sensors to be folded into an entire system as they were brought online while minimizing the need to modify the other components of the subsystem. Additionally, new scenarios can easily be added to the system without requiring additional code.

In future work, we see that the most beneficial work can be accomplished by focusing on improving the low-level sensor details. For instance, more intelligent feature-extraction algorithms for LIDARs could help to more robustly reject spurious returns caused, e.g., by bushes near the roads. In addition, sensors with a wide vertical field of view could be employed to better identify the location of the ground plane.

## ACKNOWLEDGMENT

Parts of this paper were initially published in local German conference proceedings, i.e., passages about the perception architecture in [1], the environment model and situational reasoning in [2], and the model switching approach in [3]. The movement classification was first published in [4]. This work would not have been possible without the dedicated efforts of the Tartan Racing team.

## REFERENCES

[1] M. Darms, P. Rybski, and C. Urmson, "Vehicle detection and tracking for the urban challenge," in *Automatisierungs-, Assistenz- und Eingebettete Systeme für Transportmittel Symp.*, Braunschweig, Germany, 2008.

[2] M. Darms, C. Baker, P. Rybski, and C. Urmson, "Vehicle detection and tracking for the urban challenge," in *Proc. 5. Workshop Fahrerassistenzsysteme FAS2008*, Apr. 2008, pp. 57–67.

[3] M. Darms, P. Rybski, and C. Urmson, "An adaptive model switching approach for a multisensor tracking system used for autonomous driving in an urban environment," in *Steuerung und Regelung von Fahrzeugen und Motoren—AUTOREG*. Duesseldorf, Germany: VDI-Verlag, 2008.

[4] M. Darms, P. Rybski, and C. Urmson, "Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments," in *Proc. IEEE Intell. Vehicles Symp.*, Eindhoven, The Netherlands, Jun. 2008, pp. 1197–1202.

[5] C. Urmson, C. Ragusa, D. Ray, J. Anhalt, D. Bartz, T. Galatali, A. Gutierrez, J. Johnston, S. Harbaugh, H. Y. Kato, W. Messner, N. Miller, K. Peterson, B. Smith, J. Snider, S. Spiker, J. Ziglar, W. R. Whittaker, M. Clark, P. Koon, A. Mosher, and J. Struble, "A robust approach to high-speed navigation for unrehearsed desert terrain," *J. Field Robot.*, vol. 23, no. 8, pp. 467–508, Aug. 2006.

[6] H. Winner, "Adaptive cruise control," in *Automotive Electronics Handbook*, R. K. Jurgen, Ed. New York: McGraw-Hill, 1999.

[7] R. Bishop, *Intelligent Vehicle Technology and Trends*. London, U.K.: Artech House, 2005.

[8] Z. Sun, G. Bebis, and R. Miller, "On-road vehicle detection: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 5, pp. 694–711, May 2006.

[9] C. Tzomakas and W. von Seelen, "Vehicle detection in traffic scenes using shadows," Ruhr Univ. Bochum: Institut fuer Neuroinformatik, Bochum, Germany, IRINI 98-06, 1998.

[10] A. Kirchner and C. Ameling, "Integrated obstacle and road tracking using a laser scanner," in *Proc. IEEE Intell. Vehicles Symp.*, Dearborn, MI, 2000, pp. 675–681.

[11] D. Streller, K. Furstenberg, and K. Dietmayer, "Vehicle and object models for robust tracking in traffic scenes using laser range images," in *Proc. IEEE Int. Conf. Intell. Transp. Syst.*, 2002, pp. 118–123.

[12] M.-S. Lee and Y.-H. Kim, "An efficient multitarget tracking algorithm for car applications," *IEEE Trans. Ind. Electron.*, vol. 50, no. 2, pp. 397–399, Apr. 2003.

[13] R. Mobus and U. Kolbe, "Multi-target multi-object tracking, sensor fusion of radar and infrared," in *Proc. IEEE Intell. Vehicles Symp.*, 2004, pp. 732–737.

[14] R. MacLachlan and C. Mertz, "Tracking of moving objects from a moving vehicle using a scanning laser rangefinder," in *Proc. IEEE Intell. Transp. Syst.*, Sep. 2006, vol. 2006, pp. 301–306.

[15] A. Jazwinsky, *Stochastic Processes and Filtering Theory*. New York: Academic, 1970.

[16] R. E. Kalman and R. Bucy, "New results in linear filtering and prediction theory," *Trans. ASME, J. Basic Eng.*, vol. 83, pp. 95–108, 1961.

[17] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation With Applications to Tracking and Navigation—Theory, Algorithms and Software*. New York: Wiley-Interscience, 2001.

[18] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*. New York: Academic, 1977.

[19] L. Stone, C. Barlow, and T. Corwin, *Bayesian Multiple Target Tracking*. Norwood, MA: Artech House, 1999.

[20] Y. Bar-Shalom and X.-R. Li, *Multitarget Multisensor Tracking—Principles and Techniques*. Storrs, CT: YBS, 1995.

[21] D. B. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Autom. Control*, vol. AC-24, no. 6, pp. 843–854, Dec. 1979.

[22] I. J. Cox and S. L. Hingorani, "An efficient implementation and evaluation of Reid's multiple hypothesis tracking algorithm for visual tracking," in *Proc. Int. Conf. Pattern Recog.*, 1994, pp. 437–442.

[23] Y. Bar-Shalom, K. C. Chang, and H. A. P. Blom, "Tracking a maneuvering target using input estimation vs. the interacting multiple model algorithm," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 25, no. 2, pp. 296–300, Mar. 1989.

[24] K. Murphy, "Learning switching Kalman filter models," Compaq Cambridge Res. Lab., Cambridge, MA, Tech. Rep. 98-10, 1998.

[25] D. L. Hall, *Handbook of Multisensor Data Fusion*. Boca Raton, FL: CRC, 2001.

[26] I. Jacobson, G. Booch, and J. Rumbaugh, *The Unified Software Development Process*. Reading, MA: Addison-Wesley, 1999.

[27] L. A. Klein, *Sensor and Data Fusion Concepts and Applications*. Bellingham, WA: SPIE, 1999.

[28] R. R. Brooks and S. S. Iyengar, *Multi Sensor Fusion*. Upper Saddle River, NJ: Prentice–Hall, 1998.

[29] S. S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Boston, MA: Artech House, 1999.

[30] M. Darms, "Eine basis-systemarchitektur zur sensordatenfusion von umfeldsensoren fuer fahrerassistenzsysteme," *Fortschrittberichte VDI Reihe 12*, no. 653, 2007, Duesseldorf, Germany.

[31] F. Dellaert, D. Pomerlau, and C. Thorpe, "Model-based car tracking integrated with a road follower," in *Proc. IEEE Int. Conf. Robot. Autom.*, Leuven, Belgium, May 1998, vol. 3, pp. 1889–1894.

[32] S. Gupte, O. Masoud, R. Martin, and N. P. Papanikolopoulos, "Detection and classification of vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 3, no. 1, pp. 37–47, Mar. 2002.

[33] N. Kaempchen, K. Weiss, M. Schaefer, and K. Dietmayer, "IMM object tracking for high dynamic driving maneuvers," in *Proc. IEEE Intell. Vehicles Symp.*, 2004, pp. 825–830.

[34] J. Effertz, "Tracking und datenfusion im urbanen fahrzeugumfeld," in *Automatisierungs-, Assistenz- und Eingebettete Systeme für Transportmittel, Symp.*, Braunschweig, Germany, 2008, pp. 181–197.

[35] R. MacLachlan, "Tracking moving objects from a moving vehicle using a laser scanner," Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-RI-TR-05-07, Jun. 2005.

**Michael S. Darms** was born in Berlin, Germany, on October 19, 1976. He received the Dipl.-Wirtsch.-Ing. degree in business administration and electrical engineering and the Dr.-Ing. degree (with highest distinction) from Technische Universität Darmstadt, Darmstadt, Germany, in 2002 and 2007, respectively.

From 2002 to 2006, he was a Scientific Assistant with the Chair of Automotive Engineering, Technische Universität Darmstadt, where he worked on project PRORETA, which is a joint research project with Continental Automotive Systems. New approaches for automotive collision avoidance and collision mitigation applications were developed. He was responsible for the environment perception and sensor data-fusion algorithms. In 2006, he joined Continental Automotive Systems, Lindau, Germany. From 2006 to 2007, he joined the Tartan Racing team as Continental's Embedded Engineer and worked on the perception system of the autonomous robot winning the 2007 DARPA Urban Challenge. He is currently a Manager with the Department of Advanced Engineering, Division Chassis and Safety, Continental Automotive Group. His research interests include sensor data fusion and software architecture design.

Dr. Darms is a member of Verein Deutscher Ingenieure. He was the recipient of the Outstanding Paper Award at the 2005 IEEE Intelligent Vehicles Symposium and the 2008 Best Poster Award at the IEEE Intelligent Vehicles Symposium.

**Paul E. Rybski** was born in Austin, TX, on June 25, 1974. He received the interdisciplinary B.A. degree in mathematics/computer science from Lawrence University, Appleton, WI, in 1995 and the M.S. and Ph.D. degrees in computer science and engineering with a minor in cognitive science from the University of Minnesota, Minneapolis, in 2001 and 2003, respectively.

From 2003 to 2005, he was with the Robotics Institute, Carnegie Mellon University (CMU), Pittsburgh, PA, as a Postdoctoral Fellow, working on human activity recognition, perception and modeling for human/robot interaction, and shared state estimation for multirobot teams. In 2005, he joined the faculty of the Robotics Institute as a Systems Scientist and continued his work on human–robot interaction and joined the Tartan Racing team to work on perception for autonomous driving. He is also the Perception Lead for the General Motors/CMU Autonomous Driving Collaborative Research Laboratory.

Dr. Rybski is a member of the IEEE Robotics and Automation Society, the Association for the Advancement of Artificial Intelligence (AAAI), and the Association for Computing Machinery. He has been an organizer for the AAAI Mobile Robotics Competition, the RoboCup@Home League, the Semantic Robot Vision Challenge, and the IEEE International Conference on Robotics and Automation Robotics Challenge.

**Christopher Baker** received the B.S. degree in computer science and the M.S. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, in 2002 and 2004, respectively. He is currently working toward the Ph.D. degree with the Robotics Institute, Carnegie Mellon University.

He is currently researching the challenges of the design and integration of complex robotic systems, under the supervision of Dr. J. Dolan. He has participated in the development of robotic systems for autonomous subterranean exploration and mapping. He was a Subsystem Architect for Carnegie Mellon University's Tartan Racing team, which won the 2007 DARPA Urban Challenge.

**Chris Urmson** received the B.Sc. degree in computer engineering from the University of Manitoba, Winnipeg, MB, in 1998 and the Ph.D. degree in robotics from Carnegie Mellon University, Pittsburgh, PA, in 2005.

He is currently an Assistant Research Professor with the Field Robotics Center, Carnegie Mellon University. He has served as the Director of Technology for the Urban Challenge with Carnegie Mellon University and as a Robotics Research Scientist with the Science Applications International Corporation (SAIC).

Dr. Urmson was the Technical Leader for the Tartan Racing team, which won the 2007 DARPA Urban Challenge. He has been the recipient of several technical awards, including the SAIC Research, Development, Test, and Evaluation Technology Award and the Boeing Red Phantom Award. He is also a Siebel Scholar.