Moving Object Detection with Laser Scanners

Christoph Mertz, Luis E. Navarro-Serment, Robert MacLachlan, Paul Rybski, Aaron Steinfeld, Arne Suppé, Christopher Urmson, Nicolas Vandapel, Martial Hebert, and Chuck Thorpe
The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213
e-mail: cmertz@andrew.cmu.edu, lenscmu@cs.cmu.edu, ram@ri.cmu.edu, prybski@cs.cmu.edu, steinfeld@cmu.edu, suppe@andrew.cmu.edu, curmson@ri.cmu.edu, vandapel@ri.cmu.edu, hebert@ri.cmu.edu, thorpe@ri.cmu.edu
David Duggins
QinetiQ North America, Technology Solutions Group, Pittsburgh, Pennsylvania 15238
e-mail: david.duggins@qinetiq-na.com
Jay Gowdy
Active Safety Electronics Department, Takata Corporation Gibsonia, Pennsylvania 15044
e-mail: Jay.Gowdy@takata.com

Received 15 December 2010; accepted 16 May 2012

The detection and tracking of moving objects is an essential task in robotics. The CMU-RI Navlab group has developed such a system that uses a laser scanner as its primary sensor. We will describe our algorithm and its use in several applications. Our system worked successfully on indoor and outdoor platforms and with several different kinds and configurations of two-dimensional and three-dimensional laser scanners. The applications vary from collision warning systems, people classification, observing human tracks, and input to a dynamic planner. Several of these systems were evaluated in live field tests and shown to be robust and reliable. © 2012 Wiley Periodicals, Inc.

1. INTRODUCTION

Detection and tracking of moving objects (DATMO) is a central problem in many robotics applications. We have built a DATMO system based on laser rangefinder data. The core algorithms have been adapted for many different uses: moving scanners or fixed installations; two-dimensional (2D) rangefinders or slices of data from three-dimensional (3D) sensors; detecting and tracking vehicles or pedestrians; and running indoors or outdoors, on roads, or cross-country. Our applications include safety warning systems for city bus drivers, pedestrian safeguarding for robot military vehicles operating offroad in the vicinity of dismounted troups, vehicle detection for the Urban Grand Challenge, and mapping typical pedestrian paths through an office environment. Of course, in all applications except perhaps the last, the system must be real-time to be relevant; this limits the amount of processing we can do on each scan. The algorithmic steps in each case involve cleaning up the input data, segmenting them, and matching them to data from previous scans. This process is complicated by occlusions, shape-change artifacts, and nonrigid objects such as pedestrians or foliage. Our system deals with these issues by using a number of strategies: removing spurious points such as ground returns, calculating stable features, reasoning about the geometry of occlusions, Kalman filtering, fusing data from multiple sensors, and classifying objects.

In this paper, we first review DATMO systems developed by various groups. Then we give an overview of the sensors and platforms that we use in our systems. This is followed by a description of the algorithm. Next we evaluate the system in two main applications and three smaller installations. The approximate timeline of the development of the various applications for our DATMO is shown in Figure 1.

2. REVIEW OF DATMO SYSTEMS

Many different groups have developed DATMO algorithms. Tables I and II contain a comprehensive summary of the papers discussed below. We will first discuss work with 2D laser scanners and later work with 3D laser scanners. DATMO algorithms are used inside and outside and on fixed and moving platforms. In indoor environments, the ground is usually flat and there is a smaller variety of objects, e.g., there are no cars. DATMO on a moving platform is more challenging than from a fixed one because the viewpoint is constantly changing. The speed of the platform outdoors can be high, e.g., highway speed, whereas indoors the speed is usually similar to human speed. The ones most closely related to this paper are those on moving platforms that operate outside. Fürstenberg and Dietmayer (2004) and Fürstenberg (2005) describe DATMO using the IBEO sensor. The IBEO sensor has four laser planes within an angular spread of about 3° so that it can compensate for the pitching of the vehicle. Their main focus is on automotive safety and comfort applications. Mendes et al. (2004) use their multitarget detection and tracking system for

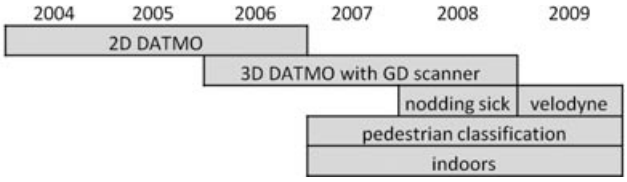

Figure 1. Timeline of the development of the various applications for our DATMO.

collision avoidance for their Cybercar. An outdoor application on a fixed platform, described in Zhao et al. (2006), monitors an intersection. They developed a joint tracking and classification algorithm for moving objects.

Compared to outdoor detection, indoor detection and tracking is simplified because in general the ground is flat, objects are mostly vertical, the environment is limited by walls, and moving cars and vegetation are basically nonexisting. On the other hand, one can encounter large groups of pedestrians. The gait of a pedestrian is commonly used for detection and classification (e.g., Zhao et al. 2007). Lindström and Eklundh (2001) detect nonstatic objects by detecting occupation of previously unoccupied space. This is similar to an occupancy grid method. The key algorithm in Schulz et al. (2001) is a sample-based joint probabilistic data association filter.

Several fixed 270° LADARs are used in Zhao and Shibasaki (2005) to cover a large area and track people in crowds. The system was able to handle 100 trajectories simultaneously. Fod et al. (2002) also use multiple LADARs. They devote a significant amount of work to evaluating the accuracy of their tracking system by comparing it to ground truth obtained by cameras.

Several papers focus on solving particular issues. Xavier et al. (2005) developed a feature detection system for real-time identification of lines, circles, and peoples legs. Castro et al. (2004) used an adaptation of the Hough transform in the feature extraction procedure to interpret scanned segments as primitive features. Montemerlo et al. (2002) presented a conditional particle filter for simultaneously estimating the pose of a mobile robot and the positions of nearby people in a previously mapped environment. Arras et al. (2007) applied AdaBoost to train a strong classifier from simple features of groups of neighboring beams corresponding to legs in range data. Khan et al. (2006) introduced a probabilistic model for interacting targets that addresses both multiple returns from single targets and merged measurements between interacting targets. They provided an algorithm for approximate inference in this model using a Markov chain Monte Carlo (MCMC) based auxiliary variable particle filter. Bruce and Gordon (2004) presented a motion model based on the

Table I. List of DATMO papers with their basic setups, type of laser scanner used, and a very short description of their contribution. The dimension "2+" is a laser scanner with four scanning planes.

Reference	Outdoor	Moving	Sensor	Dimension	Special comments
(Fürstenberg, 2005) (Fürstenberg and Dietmayer, 2004)	×	×	IBEO	2+	Automotive applications
(Mendes et al., 2004)	×	×	SICK	2	Collision avoidance
(MacLachlan and Mertz, 2006)	×	×	SICK	2	Collision warning
(Zhao et al., 2006)	×		SICK	2	Joint tracking and classification
(Zhao et al., 2007)		×	SICK	2	Observing pedestrian gait
(Lindström and Eklundh, 2001)		×	SICK	2	Similar to occupancy grid method
(Kluge et al., 2001)		×	SICK	2	Network optimization techniques
(Schulz et al., 2001)		×		2	Sample-based joint probabilistic data association
(Zhao and Shibasaki, 2005)			LD-A	2	Multiple LADARs, crowds
(Fod et al., 2002)			SICK	2	multiple LADARs, evaluation
(Xavier et al., 2005)		×	SICK	2	line, arc/circle and leg detection
(Castro et al., 2004)		×	SICK	2	Hough transform feature extraction
(Montemerlo et al., 2002)		×	SICK	2	Conditional particle filter
(Arras et al., 2007)			SICK	2	Boosted features
(Khan et al., 2006)			SICK	2	Markov chain Monte Carlo based auxiliary variable particle filter
(Bruce and Gordon, 2004)				2	Efficient trajectories motion model
(Gate and Nashashibi, 2008)			IBEO	2+	Recursive appearance estimation
(Iagnemma et al., 2008)	×	×	Velodyne	3	Urban Grand Challenge teams
(Steinhauser et al., 2008)	×	×	Velodyne	3	Motion segmentation
(Thornton and Patil, 2008)	×	×	GDRS	3	Joint spatial-temporal association

Table II. List of DATMO papers for pedestrian detection.

Reference	Outdoor	Moving	Sensor	Dimension	Special comments
(Wang et al., 2007)	×		SICK	2	Interacting object tracking
(Song et al., 2008)	×		SICK	2	Online supervised learning
(Arras et al., 2008)		×	SICK	2	Multihypothesis leg-tracker with adaptive occlusion probabilities
(Frank et al., 2003)			SICK	2	Sequential Monte Carlo methods
(Cui et al., 2007)			SICK	2	Rao-Blackwellized Monte Carlo data association filter
(Taylor and Kleeman, 2004)			SICK	2	Multihypothesis framework
(Gate and Nashashibi, 2009)	×	×		2	Individual and groups of pedestrians
(Gidel et al., 2009)	×	×	IBEO	2+	Fusion of planes, particle filter
(Glas et al., 2007)	×	×	SICK	2	Parametric human shape model particle filter
(Spinello et al., 2010)	×		Velodyne	3	AdaBoost, geometric and statistical features

intuition that people tend to follow efficient trajectories through their environments rather than random paths. Gate and Nashashibi (2008) proposed a system that recursively estimates the true outlines of every tracked target using a set of segments called Appearance.

Many groups deal with tracking people in crowded environments. In most of the papers, the sensors were either fixed or on moving platforms indoors, i.e., the sensors saw the people at a constant height. Wang et al. (2007) introduced a scene interaction model and a neighboring object interaction model to take long-term and short-term interactions, respectively, between the tracked objects and its surroundings into account. Song et al. (2008) used an online supervised learning-based method for tracking interacting targets with a laser scanner. They compare their method to a joint probabilistic data association filter (JPDAF), a Monte Carlo joint probabilistic data association filter (MC-JPDAF), and a nearest-neighbor standard filter (NNSF). Arras et al. (2008) used a tracker that detects and tracks legs separately with Kalman filters, constant velocity motion models, and a multihypothesis data association strategy. They extended the data association so that it explicitly handles track occlusions. Two approaches are described in Frank et al. (2003) to deal with the multiple target tracking problem. Both are based on sequential Monte Carlo methods and joint probability data association. To track people in dense crowds, Cui et al. (2007) introduced a stable feature extraction method based on accumulated distribution of successive laser frames, and then they introduced a region coherency property to construct an efficient measurement likelihood model. The final tracker is based on a combination of an independent Kalman filter and a Rao-Blackwellized Monte Carlo data association filter (RBMC-DAF). Taylor and Kleeman (2004) focused on tracking legs. They extended a multiple hypothesis framework to allow for both association uncertainty and a switched dynamic model depending on the currently moving leg. Gate and Nashashibi (2009) used a feature-based classification approach that detects and tracks regular pedestrians. It also copes smoothly and efficiently with groups of people. A "Parzen Window" kernel method is described in Gidel et al. (2009) that allows the centralized fusion of information from the four planes of a laser scanner. Additionally, a particle filter with feedback in a laser image is employed for a closer observation of pedestrian random movement dynamics. Glas et al. (2007) also used a particle filter. A parametric human shape model is recursively updated to fit observed data after each resampling step of the particle filter. The laser scanners are mounted at waist height, which allows the model to include torso and arms.

To increase the amount of information available to do detection, tracking, and classification, one can combine the 2D LADAR with other sensors or use a 3D LADAR. A popular approach is to combine LADAR with video (Cui et al., 2008; Fayad and Cherfaoui, 2007; Kämpchen, 2007; Labayrade et al., 2005; Mählisch et al., 2006; Monteiro et al., 2006; Perrollaz et al., 2006; Thornton et al., 2008). In our work, we took the approach of using 3D LADAR data exclusively. Since mobile 3D LADAR scanners with sufficient update rates have only become available in recent years, the field of 3D DATMO is relatively new. Many groups that participated in the Urban Grand Challenge used a Velodyne LADAR Iagnemma et al. (2008). Of the 11 finalists, seven used a Velodyne and one used an actuated SICK LMS. A Velodyne LADAR is also used in Steinhauser et al. (2008). Their system extracts essential information about drivable road segments in the vicinity of the vehicle and clusters the surrounding scene into point clouds representing static and dynamic objects. Thornton and Patil (2008) used their own proprietary LADAR (see experimental unmanned vehicle (XUV) and Suburban in Figure 2). The LADAR data association

Figure 2. Five vehicle platforms on which our DATMO system was running live. From left to right, up to down: GDRS XUV and Suburban, Tartan team Boss, our own test vehicle Navlab 11, and a Pittsburgh transit bus. Furthermore, we had DATMO running live in a fixed indoor setting (lower right). One of the SICK sensors can be seen, indicated by a white arrow.

problem is approached with a joint spatial-temporal solution and several features are used for classification. Spinello et al. (2010) detected people in 3D data. They applied AdaBoost to train a strong classifier from geometrical and statistical features of groups of neighboring points at the same height. In a second step, the AdaBoost classifiers mutually enforce their evidence across different heights by voting into a continuous space. Pedestrians are finally found efficiently by mean-shift search for local maxima in the voting space.

There are also papers that explore subalgorithms of DATMO. Reviews of clustering algorithms can be found in Jain et al. (1999) and Xu and Wunsch (2005). A clustering method for 3D laser data is discussed in Klasing et al. (2008). Premebida and Nunes (2005) describe segmentation and geometric primitives extraction from 2D LADAR data. Nguyen et al. (2007) compare line extraction algorithms. Statistics of natural images and range images can be found in Huang and Mumford (1999) and Huang et al. (2000). Pellegrini et al. (2009) address the question of how to model human behavior to better track people in crowded environments. They developed a social behavior model in which the location and velocity of people around a person are taken into account. Similarly, Luber et al. (2010) integrate a model based on a social force concept into a multihypothesis target tracker. Estimation of the ground bearing surface can be found in Lalonde et al. (2006) and Kelly et al. (2006).

There are several data sets of laser scanner data freely available online. In Pandey et al. (2011), a Velodyne, two push-broom forward-looking Riegl LIDARs, and a Point-Grey Ladybug3 omnidirectional camera system collected data from a moving vehicle. Peynot et al. (2010) describe the Marulan data sets. They contain data from four 2D laser scanners, a radar scanner, a color camera and an infrared camera. The MIT Urban Grand Challenge team made their data available Huang et al. (2010). Their vehicle had a Velodyne, five cameras, and 12 SICK laser scanners. Access to annotated laser scanner data can be found in Yang et al. (2011). This data set was collected with our Navlab11 vehicle (Figure 2).

Lastly, we want to mention some surveys of pedestrian and vehicle detection in general. Gandhi and Trivedi (2007) contains a survey of active and passive pedestrian protection systems. Among other things, it compares pedestrian detection using video, stereo, visible and IR, RADAR, LADAR, and multiple sensors. Enzweiler and Gavrila (2009) has an extensive survey of monocular pedestrian detection. A review of onroad vehicle detection can be found in Sun et al. (2006). Its emphasis is on vision sensors, but it also discusses other sensors such as laser scanners.

3. SYSTEM OVERVIEW

The central sensor for our DATMO system is a laser scanner. We have used many types, such as 2D laser scanners like a SICK LMS or 3D laser scanners like a nodding SICK LMS, a Velodyne, or a General Dynamics Robotic Systems (GDRS) mobility LADAR, which is a custom sensor. The sensors we used and their configurations are listed in Table III.

The sensors can be mounted on vehicles or they can be used on a stationary platform. Figure 2 shows various robotic systems with the sensors. The first vehicle is a Demo III XUV (Shoemaker and Bornstein, 1998) and we used it with 4 SICK LMS LADARs, one on each side, or with a

Table III. Sensors used with our DATMO.

	Resolution	Rate	Field-of-view
SICK LMS	1°	75 Hz	180°
SICK LMS	0.5°	37 Hz	180°
SICK LMS	0.25°	19 Hz	100°
Nodding SICK LMS	1°	2 Hz	110°
Velodyne	0.1°	5 Hz	180°
GDRS 3D mobility LADAR		5–15 Hz	90°–180°

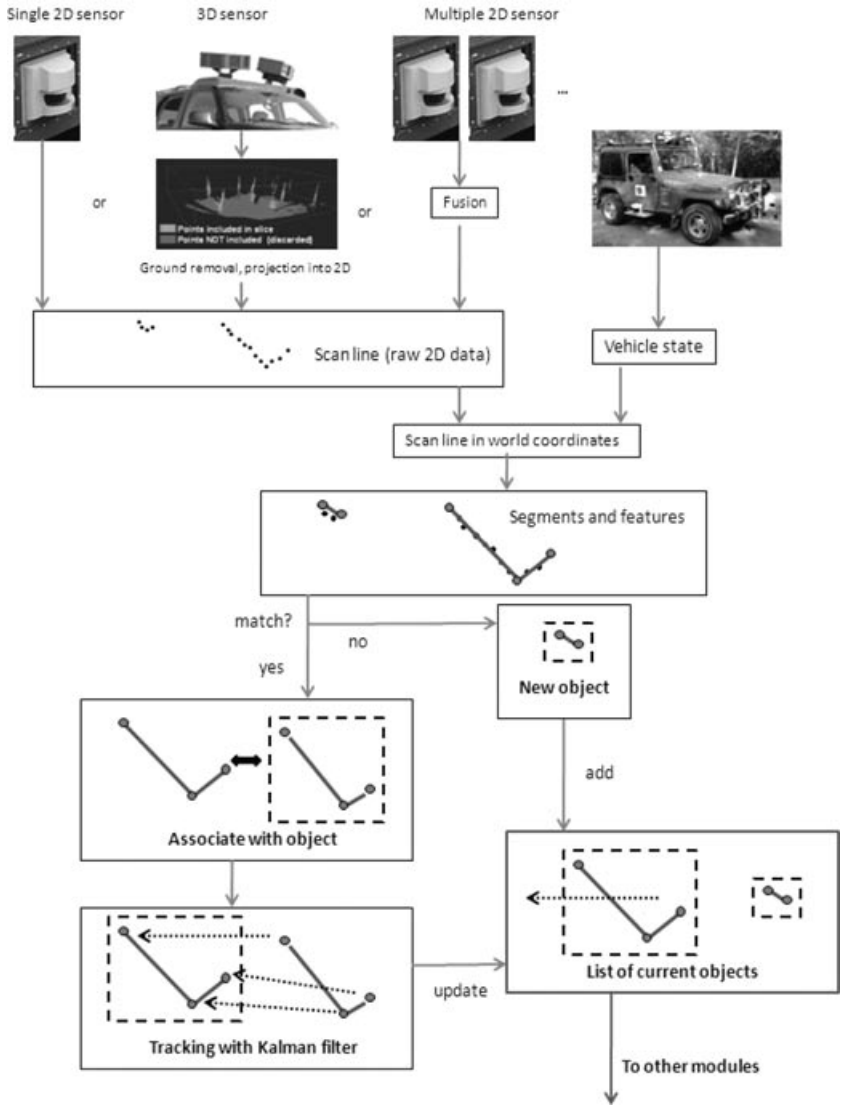

Figure 3. Flow chart of our DATMO.

GDRS 3D LADAR. The next vehicle is a Suburban with a dual GDRS 3D LADAR. We also had a single SICK LMS on the front of it. The next vehicle is Boss, the winner of the Urban Grand Challenge. It ran our system on two SICK laser scanners. Next is a Jeep with 3 SICKs (sides and front) and a transit bus with two SICKs, one on each side. Also, data from nodding SICKs and a Velodyne laser were used as input to our DATMO, and these data were post-processed. The last setup was stationary indoors where 4 SICKs were observing a wide area. In all cases except the post-process data, our system was running live.

The basic modules of our DATMO are illustrated in Figure 3. It begins with scan lines that can be directly obtained from a 2D LADAR scanner, from the projection of a 3D scan onto a 2D plane (Section 5) or from the fusion of multiple sensors (Section 4.2). This scan line is transferred into world coordinates and segmented. Line and corner features are extracted for each segment. The segments are associated with existing objects and the kinematics of the objects are updated with the use of a Kalman filter. Segments that cannot be associated trigger the creation of a new object. The Kalman filter assumes the errors to be Gaussian. This is mostly true with the measurements we are using, but sometimes we are confronted with outliers and special cases. In the next section, these algorithms are discussed in more detail.

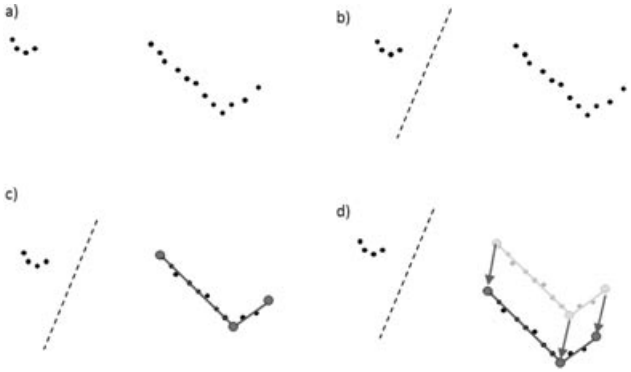

Figure 4. Graphical illustration of our DATMO algorithm: (a) raw data, (b) segmented data, (c) fitting of lines and corners and feature extraction, and (d) tracking of feature points.

4. ALGORITHM DESCRIPTION

We will first describe the core algorithm of our DATMO. In the second subsection, we will show how our DATMO can be used with multiple sensors. Our DATMO algorithm is described in detail in MacLachlan and Mertz (2006), Navarro et al. (2006) and Navarro-Serment et al. (2008).

4.1. Core Algorithm

A laser scanner gives a scan line within a 2D plane, i.e., a set of ordered points with constant angular steps. In the first step, which is the *segmentation* process, the points in the scan line are clustered into segments with a basic region-growing algorithm. A point is added to an existing cluster if its distance to the cluster is less than a threshold. A graphical example is shown in Figures 4(a) and (b), where the raw data are segmented into two clusters. In Figure 10, one can see actual data, the laser data (white dots) that are within a box belong to the same segments.

4.1.1. Occlusions

During simple segmentation, we also detect when background objects are partially occluded by foreground objects. Each point is classified as occluded or normal. A point is occluded if an adjacent point in the scan is in a different segment and in front of this point, or if it is the first or last point in the scan. This flag has several uses:

- When an occluded point appears at the boundary of an object, we consider this to be a false boundary.
- We only count nonoccluded points when determining if there are enough points to create a new track, or if the point density is high enough for a segment to be compact.
- If an occluded point defines the position of a feature, then that feature is regarded as *vague*.

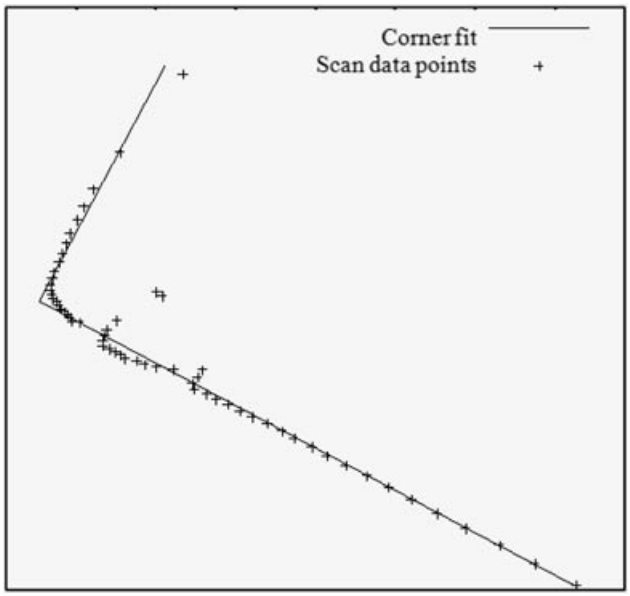

Figure 5. A corner fit to points from a vehicle that include outliers and a rounded corner.

In the segmentation process, missing range returns are treated as points at maximum distance, and are not assigned to any segment. Similarly, if there is a large enough dropout in the middle of an object, it will split the object into two segments (e.g., a person standing between the sensor and a wall will result in splitting the wall in two segments). If any point in a segment is occluded, then the segment is considered partially occluded. Finally, tracks that are not associated on the current tracker iteration, and which were last associated with a partially occluded segment, are considered candidates for total occlusion. These will be removed from the list after failing to be associated for several iterations.

4.1.2. Features

Next we want to extract features from the segments that are stable to viewpoint changes. When the host vehicle drives by extended objects such as other vehicles or buildings, the change in viewpoint modifies the apparent shape of the object. Features such as the center-of-mass (centroid) of the points change their location even if the object is fixed. We found that corners and lines fitted to the points [Figure 4(c)] are much more stable to changes of viewpoint. An example of a corner fitted to the points coming from a vehicle is shown in Figure 5. To make the fit robust to outliers, we first make a trial fit and then we fit again with 20% of the worst points (i.e., largest distance to the fitted line) discarded. We also give a lower weight to points in areas that are densely sampled, i.e., closer to the scanner. We make both a line and a corner fit. The corner fit is chosen if it is significantly

better than the line fit. The ends of the line and the corner of a segment are the feature points [Figure 4(c)].

4.1.3. Data Association

To track the objects we first have to associate them from one scan to the next. This is mainly done by detecting the overlap between the areas of the predicted object and the current segment. For two objects to overlap, it is required that at least one actual measurement point falls inside the track outline, and symmetrically, that when the track's last measured points are updated for predicted motion, that at least one of these predicted points falls inside the segment's outline. The track outline is determined by finding the corners of a bounding box that contains all the points in the segment. Since the points are on the edge of the object, we expand the object outline by 0.8 m, then check if any points fall inside this expanded outline. However, in cases when objects split, merge, or are too close to each other, the association is ambiguous. In these cases, the main point is to determine which segment will best support the existing track. For this purpose, we calculate the *closeness* between features. Closeness is defined as the sum of the inverse distance between corresponding features (e.g., between corners). This measure of closeness differs from the RMS distance in two important ways: First, the result is dominated by the best agreeing features, not the worst, so it is not confused by the gross shape change that happens when tracks split or merge. Second, the result is not normalized by the number of features in correspondence. The more features, the higher the closeness.

In a split/merge situation, we discard the track or segment with less information. If a current segment is not associated with a predicted object, a new object is created. Likewise, if an object cannot be associated with a current segment for several consecutive cycles, it is deleted.

The position, velocity, acceleration, and turn rate of the object is estimated by a Kalman Filter with a constant linear acceleration, constant turn rate, and fixed process noise [Figure 4(d)]. Each feature is considered to be an independent measurement. The turn rate is the derivative of the heading and the heading is usually derived from the orientation of the line or corner. If we do not have a good line or corner fit, then the direction of the linear velocity is used.

Measurement noise. The two dominant sources of feature position uncertainty are angular resolution limits and instability in segmentation or feature extraction. The feature position uncertainty is decomposed into longitudinal (lengthwise, along the scan line) and lateral (normal to the line) uncertainties. We address these two classes of uncertainty by combining two noise estimates: static and adaptive.

The static part is calculated from a single scan and accounts for things like geometric effects, fit quality, or missing returns. For linear features, it is computed from separate longitudinal and lateral variances. At long ranges and shallow incidence angles, the interpoint spacing can exceed the segmentation threshold, causing the line to end at that point even though the object extends on; the longitudinal uncertainty is then effectively infinite. In this case, the line end should not be interpreted as a stable corner feature.

The adaptive part downweighs features that have unstable positions. Its value is derived from the position measurement residue (the difference between prediction and measurement). After a feature has been associated 15 times, the covariance of the position measurement residue (the difference between prediction and measurement) is used as the measurement error. The static noise covariance times 0.3 is added to the adaptive noise to insure a lower bound.

Special cases are when the end point of a line does not correspond to the actual end of the object. This happens when an object in the foreground casts a shadow on an object behind it (occlusion) or when the interpoint spacing exceeds the segmentation threshold at long distances or shallow angles. In such cases, we use a modified Kalman filter model where the longitudinal innovation (i.e., the velocity and acceleration components of a segment running lengthwise) is set to zero to suppress unwanted false apparent motion.

There are still more challenging cases, e.g., when both end points of a line do not correspond to the actual ends the object, ground returns (the scan line strikes the ground instead of an object), missing returns, or feature extraction errors. We only update a track when the innovation is above a given threshold and at the same time within physically plausible limits. Furthermore, we implemented an additional validation procedure that operates independently of the Kalman filter. We collect the last 35 segments (raw measurements) associated with each track, then check how well the track path matches up with the measurements if we project it backward in time from the current state of the object. If the discrepancy is too large, then there are either unmodeled disturbances (rapid change in acceleration or turn rate) or there is a tracking failure due to problems in feature extraction, etc. and the track is invalidated.

4.2. Using Multiple Sensors

There are three main reasons why multiple sensors are used. The first is when one sensor alone cannot cover all the area. For example, in Figure 6 four SICK sensors are needed to detect objects all around the vehicle. The second reason is if you want to combine sensors with different properties to achieve a quality of detection that cannot be achieved with one kind of sensor. Those properties could be different range, resolution, update rate, or even different sensing modalities like LADAR, RADAR, or vision. The third reason is redundancy.

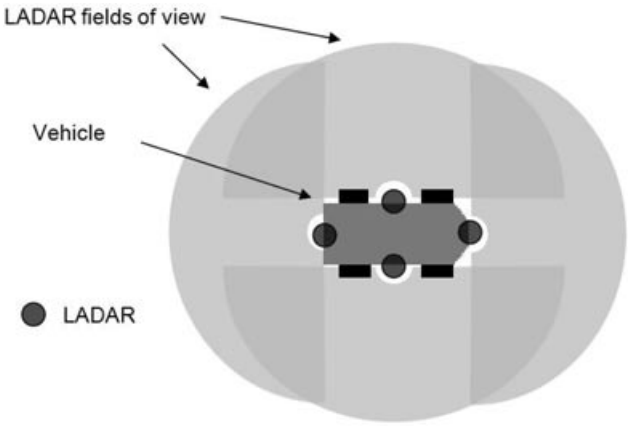

Figure 6. An example of a multiple sensor arrangement. Four LADARs, each with 180° field-of-view, are mounted on the sides of the vehicle to give a complete coverage around the vehicle.

Combining the data from multiple sensors can happen at any level during the detection and tracking process. For synchronization purposes, all the sensor data are time-stamped. Only the most recent data are used. As shown in Figure 3, assume a set of Q sensors $S = \{S_1, S_2, \ldots, S_Q\}$ is used. Define $Z_j^q = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_q}\}$ as the set of N_q points collected at time t_j by the sensor $S_{q\in\{1,2,\ldots,Q\}}$, whose elements are represented by Cartesian coordinates $\mathbf{x} = (x, y, z)$. From these definitions, we now proceed to describe the three different combination approaches we have used in various systems:

Point level. In this mode, the set of raw points from all sensors, $Z_j = Z_j^1 \cup Z_j^2 \cup \cdots Z_j^Q$, is resampled to form a new set of points. To fit our data format, the points in the new set are expressed in polar coordinates where the angles are discretized. The result is a vector of distance measurements coming from consecutive bearings, which resembles the kind of data provided directly by a line scanner such as the SICK sensor. This is referred to as a *virtual laser scanner*. In the resampling step, if there are two or more points in one angular bin but at different radial distances, only the point closer to the origin is used. The origin of the coordinate system is the location of the new virtual laser scanner. The great advantage of this method is that for the virtual laser scanner, the overlap regions look like any other region and therefore the overlap region does not need any further special treatment. This point level fusion works well only when the sensors are relatively close together. Otherwise, the occlusions seen by the real sensors are too different from the occlusions perceived by the virtual sensor. Also, one will inadvertently have to discard some of the original points and therefore not make full use of all the available data. A version of this is described in Section 5 using the dual LADAR on the Suburban in Figure 2.

Segment-to-object level. In this mode, each sensor is treated as a stand-alone sensor up to the point where the segments are formed and the features of the segments are extracted. For each sensor $S_q \in S$, a set of segments is extracted from the corresponding measurement set Z_j^q. Then, a set of features is extracted from each segment [Figure 4(c)]. The feature points of the segments from all sensors S are then used to update one common list of objects O_j, which contains estimates of position and velocity of all the targets at time t_j. Notice that in this approach the segments are not fused together to get a new list of segments. With this method, the occlusion reasoning is fully preserved and therefore one can use sensors at very different locations, e.g., in the system described in Section 8.2 the SICK sensors are located in different corners of the room and yet objects are consistently tracked around the island in the middle of the room. One disadvantage of this method is that at the boundaries between sensor FOVs, the system can be confused when tracking large objects, because the sensors see very different parts of the object.

Object level. Here each sensor is a stand-alone system that produces its own list of objects. For each sensor $S_q \in S$, a set of segments is extracted from the corresponding measurement set Z_j^q. Then, a set of features is extracted from each segment [Figure 4(c)]. However, as opposed to the segment-to-object level approach, only the feature points from sensor S_q are used to update one list of objects for this sensor, O_j^q, which contains estimates of position and velocity of all the targets "seen" by S_q at time t_j. These lists are then combined into one list that includes estimates of all targets, O_j (to facilitate fusing the data, all the position and velocity estimates are expressed in the vehicle reference frame). This method is very general and can even be used when combining very different systems, i.e., systems that might use different sensors and track different feature points. A disadvantage of this method is that at the boundary between two sensors, the object needs to be acquired from scratch by the second sensor. It does not use the knowledge about the object from the first sensor to initialize the object. This method was used on the Urban Grand Challenge vehicle (Section 7.3).

5. DATMO WITH 3D LADAR

As mentioned before, our DATMO is capable of processing range measurements from both line and 3D scanners. The information pipeline used to feed sensor data into our system is the same for both types of sensors, as seen in Figure 3. The use of 3D LADAR improves the performance in uneven terrains, while the line scanners, which usually operate at faster rates, perform better in flatter areas. This

feature allows the system to adapt to a wider range of operation environments by combining the advantages of each sensor type.

The 3D scanners used in the CTA (Collaborative Technology Alliances) project can be seen in Figure 2 on top of the XUV and the Suburban. Our DATMO system uses 2D scans of ordered points. If there are several 3D LADARS, e.g., two of them on the Suburban in Figure 2, the 3D point clouds are first combined to one 3D point cloud and then the projection is performed. This is a variant of the point level combination of multiple sensors mentioned in Section 4.2.

To make use of the 3D point cloud, it first needs to be appropriately converted into a 2D virtual scan, as described next.

5.1. Projection into a 2D Plane

Since it is computationally too expensive to process the entire point cloud, we initially isolate a 2D virtual slice, which contains only points located at a certain height above ground. As shown in Figure 7, a 3D scanner produces a point cloud, from which a "slice" is projected onto the 2D plane, resulting in a virtual scan line. This scan line is a vector of range measurements coming from consecutive bearings, which resembles the kind of data reported directly by a line scanner such as the SICK laser scanner. This projection is done by collapsing into the plane all the points residing within the slice, which is defined by its height above the ground, and then resampling the points. Resampling means we pick for each angular bin the closest point and order the points.

One difficulty in the projection procedure is determining which points belong to the ground plane. In general, the terrain is uneven and it is not sufficient to linearly extend the ground plane underneath the vehicle. It is necessary to maintain a ground elevation map. Common algorithms for estimating the ground surface [e.g., see Lalonde et al. (2006) and Kelly et al. (2006)] are computationally too expensive. We developed an efficient method that is described in the following section.

5.1.1. Elevation Map

The system accumulates processed LADAR measurements in a grid called an *elevation map*, which is always aligned with the vehicle's reference frame and is centered on the vehicle's current position. The elevation map provides a surface description model in which each cell contains an estimate of the ground elevation for the corresponding area. The mean and standard deviation of the heights of all scan points that are inside each cell are computed, and the elevation is then calculated by subtracting one standard deviation from the average height of all the points in the cell. The key properties of this simple algorithm are that mean and standard deviations can be calculated recursively, and that

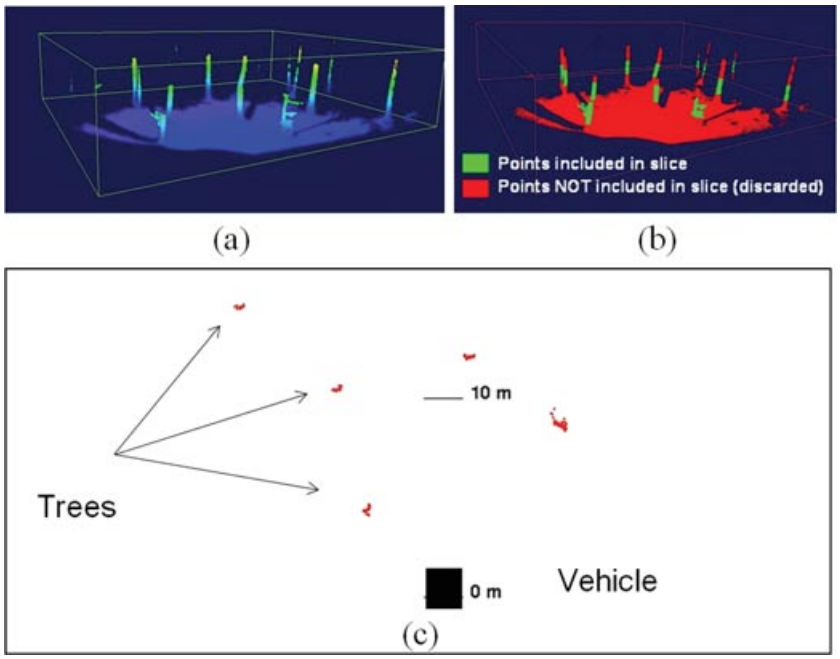

Figure 7. Projection of virtual scan line. A point cloud is collected from a wooded area (a). The points in the cloud located within a certain height from the ground are projected into a 2D plane (b) and processed as if it were a single scan line. The resulting projection is shown in (c), top view. As expected, this measurement resembles what would be obtained using a line scanner.

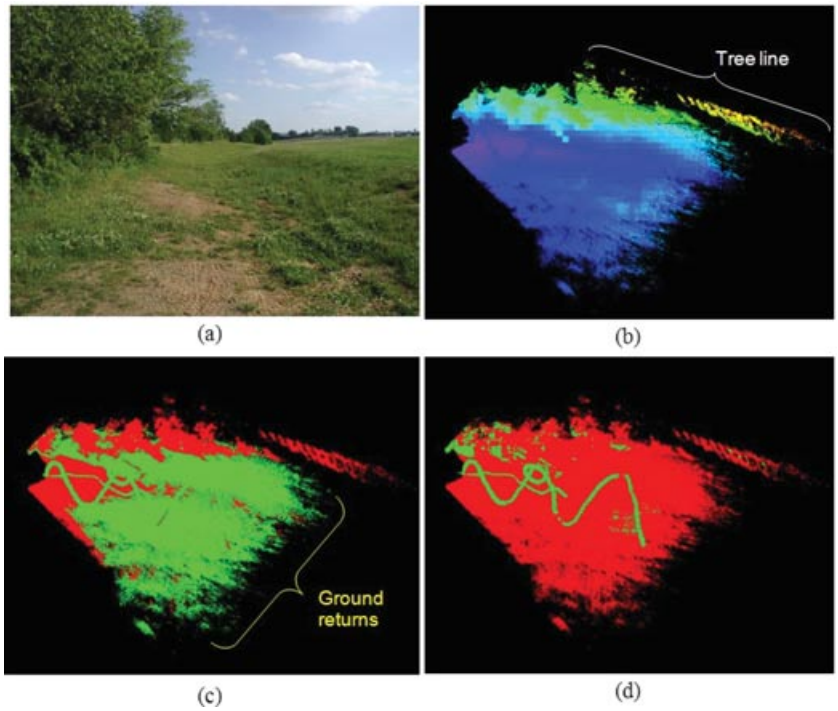

Figure 8. Sample field test using ground elevation to eliminate ground returns. The test was conducted in uneven terrain (a). The corresponding ground elevation map is shown in (b) (viewed from above). The 3D point clouds are shown in (c) and (d). Points used in the slice are shown in green, while points in red are ignored. In (c), the virtual slice is a plane, and the lack of compensation for ground elevation results in significant ground returns. Conversely, in (d) the virtual slice is adjusted depending on ground elevation. Ground returns are greatly reduced, increasing the likelihood of collecting measurements from potential targets, while reducing the probability of false detections resulting from processing ground returns.

the elevation estimate is never below the lowest point while still having about 80% of the points above ground. To keep the computational cost of maintaining the map to a minimum, the map is indexed as a 2D ring buffer using modulo arithmetic. This logically scrolls the map as the vehicle moves by physically wrapping around in memory. In this way, only the cells for which no estimates have been computed are updated when the map is scrolled in response to vehicle movement.

5.1.2. Ground Removal

The elevation map is used to eliminate returns from the ground. We only keep points that are between a lower and an upper height bound. The values of height bounds depend on the environment and the tolerance of the particular application to spurious measurements. Usually the lower bound is 0.5 m. If there is little or no ground vegetation, one can use 0.3 m, and if there is tall grass one might need to go as high as 1 m. The only reason to have an upper height bound is to avoid getting returns from overhangs. It needs to be at least as high as the clearance needed for the robot. We commonly use 3 m. The ground removal is illustrated in Figure 8, where the trajectory of a moving object is easily identified, given the increased amount of ground returns being ignored (red).

6. TESTING OF DATMO COMPONENTS

In this section, we give quantitative measures of the quality of the velocity determination and the tracking, particularly that of humans. The tracking is evaluated in terms of detection distance, velocity error and delay, and track breakup. These numbers will give us a baseline of the performance of our DATMO.

The most commonly used sensor in our systems is the SICK LMS (see Table III). Its range is up to 80 m. The manufacturer's claim that the resolution and accuracy of the SICK laser scanner is 1 cm has been confirmed by our experiments, including variations over space and time [Section 1.32 of PATH and CMU (2004)]. The following derived quantities are for systems with a SICK LMS unless stated otherwise.

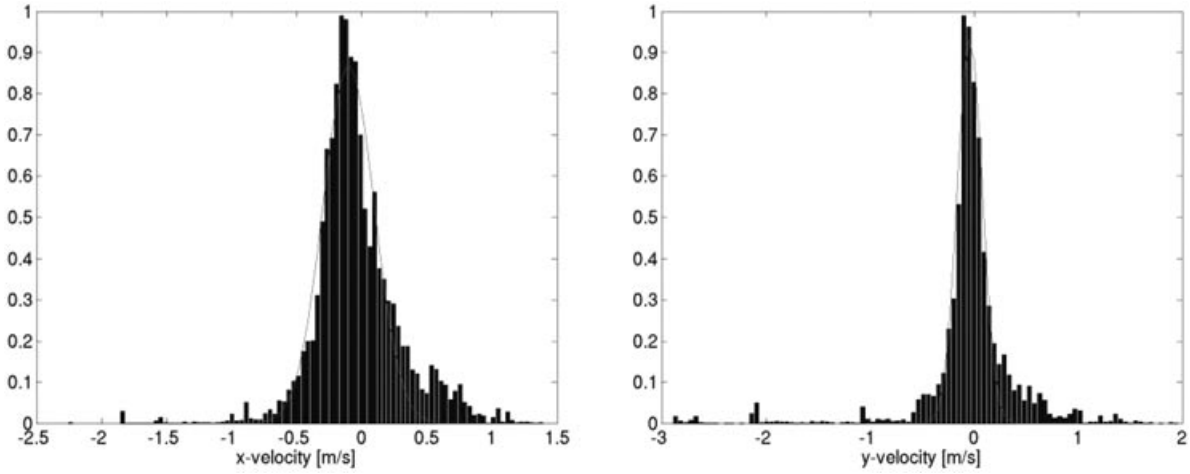

Figure 9. Distribution of errors in velocity determined by our DATMO. x velocity (left figure) is parallel and y velodity (right figure) is perpendicular to the direction of travel of the host vehicle.

6.1. Object Velocity

An important derived quantity is the object velocity. To get a good characterization of its error function we studied a 40-s-long data set from the bus project [Section 2.3.2.2 of PATH and CMU (2006)]. During this time, the bus was driving at about 10 m/s past a series of fixed objects (parked cars, mail boxes, and lamp posts), while our DATMO detected 312 different objects. The distribution of the measured velocities shows the error function.

Figure 9 shows the distributions of the velocity errors in the x and y directions (normalized so that the maximum is 1). The left plot shows the velocity error in the x direction and the right shows the y direction. Gaussian curves were fit to the distributions and gave the parameters shown in Table IV.

The centers of both distributions are not exactly at zero, even though the objects are known to be stationary. The offset for the x direction can be explained by a 1% inaccuracy of the speed of the bus. The offset for the y direction could be due to a misalignment of 0.2° of the laser scanner. Both of these errors are very small and well within the known accuracy of the bus speed and the sensor alignment.

The distributions are fairly well described by the Gaussian curve, except for their tails, which are much stronger. These outliers can come from inconsistent scanner data, e.g., if the scanner sees different parts of an object or does not get any return from certain parts of an object. The bus itself was not completely level and therefore the sensor plane was not exactly parallel to the ground. This would explain why we did not always get consistent returns, i.e., the scanner probed the objects at different heights depending on the distance of the objects.

For the great majority of objects, the velocity determination worked well, i.e., it was accurate enough to analyze a situation and issued appropriate warnings. The few outliers can occasionally cause false warnings (see Section 7.1.3).

Table IV. Parameters of velocity errors

	Center	σ
x velocity	−0.10 m/s	0.20 m/s
y velocity	−0.04 m/s	0.13 m/s

6.2. Tracking tests

To test the tracking of people in more detail, we used Navlab 11 (Figure 2, left center) to detect and track pedestrians. The ground truth for the pedestrian speed was determined by markers on the ground and a stopwatch. A bird's-eye view of a typical run is shown in Figure 10. We are interested in detection distance, velocity error, velocity delay, and track breakup to evaluate the performance of the tracking. Table V shows some of these quantities for five runs with different vehicle and target velocities.

Detection Distance. This is the distance to the object when it is detected for the very first time. This happens when the target returns at least three points. For a 0.5° resolution scanner and a target of about 60 cm cross section, this distance is between 30 and 40 m. The average in Table V is indeed 35 m with only a few meters variation from run to run.

Velocity Error. This is the difference between the velocity measured by the system and the ground-truth velocity. Mean and standard deviation of velocity error are reported in Table V. The mean error is small and is entirely

Table V. Tracking pedestrians from Navlab 11. Five individual runs and their average are shown.

Vehicle speed m/s (mph)	Target velocity m/s	Estimated velocity m/s	Mean velocity error m/s	Target detection distance m	velocity delay s	st. dev. of velocity m/s
9.1 (20.4)	1.62	1.69	−0.066	34.7	1.1	0.046
10.5 (23.5)	2.18	2.1	0.077	36.9	1.8	0.048
11 (24.6)	3.95	4.08	−0.132	34.9	3.5	0.066
14.2 (32)	1.71	1.78	0.061	33.7	1.4	0.075
17.7 (40)	2.89	2.92	0.027	34.3	1.4	0.081
12.5 (28)	2.47	2.51	0.007	34.9	1.8	0.063

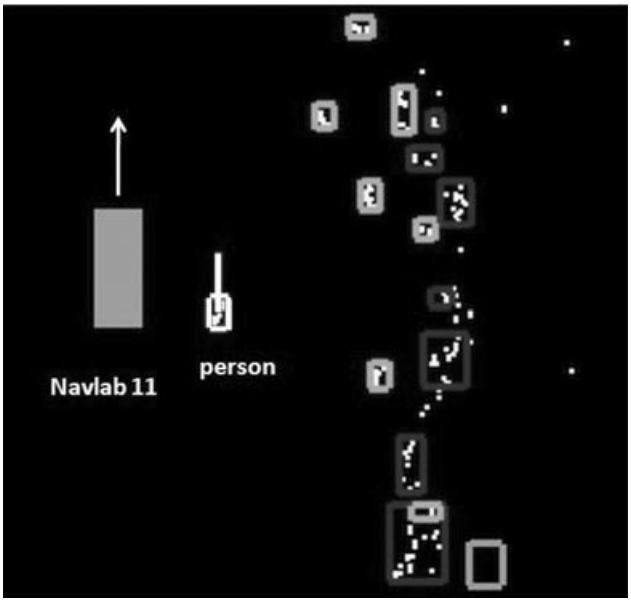

Figure 10. A pedestrian (white) moves at constant velocity. Navlab 11 estimates the pedestrian's velocity while driving at 35 mph. The white line indicates the pedestrian's estimated velocity. The intensity of the other objects indicates their classification: light gray is low probability human, dark gray is ruled out as human, and gray without points inside is a previous object no longer associated with any point. Raw scanner measurements appear as white dots.

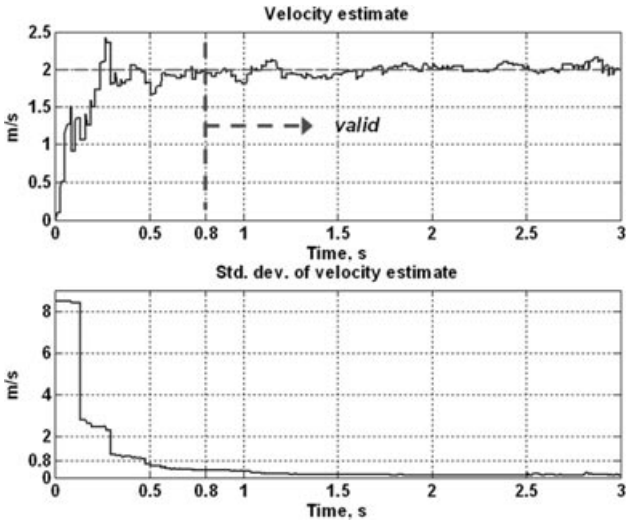

Figure 11. Estimation of target velocity. A pedestrian walking at a constant speed of 2 m/s is tracked. The system reports a valid velocity estimate after 0.8 s, as shown in the top figure. The standard deviation of the velocity estimate, plotted in the bottom figure, is one of several criteria used to validate the track.

explainable by the uncertainties of measuring the ground truth. The standard deviation is about 0.06 m/s. It is smaller than the velocity error reported in Section 6.1, but this is expected because pedestrians are much smaller than vehicles and therefore have a better defined location.

Velocity Delay. This is the delay between the time at which an object is detected and the time at which its velocity is valid. It is valid if certain tests are passed, including the estimated velocity error being below a given threshold. In our examples, the delay varies between 1.1 and 3.5 s with an average of 1.8 s

The location of an object is available after one cycle. The estimation of the velocity takes several cycles to be reliable. It is initiated at 0 m/s and the Kalman filter needs to be updated several times to give a valid measurement. An example is shown in Figure 11 in which it took 0.8 s for the velocity to become valid. Among one of the criteria of the validity test is a threshold on the measurement error from the Kalman filter (Figure 11, bottom).

Track Breakup. The tracking process can be negatively affected by several causes. The system fails to detect a target when it is occluded, when it has poor reflectivity, or when objects are too close to each other and it is not clear

Table VI. Track breakup analysis: four pedestrians walking alongside an XUV moving at low speed.

Target	No. of track breakups	Average velocity delay s	Minimum velocity delay s	Maximum velocity delay s	Time with valid velocity estimate %
A	4	2.10	1.5	2.8	96.0
B	10	3.57	0.2	10.7	79.5
C	9	0.81	0.2	2.9	96.4
D	57	4.95	0.8	15.0	28.0

whether to segment the data as one or more objects. It is difficult to conduct a systematic quantitative evaluation of track breakups because it is very much dependent on the specific situation. We will describe one run with four people walking alongside a manually driven XUV (Figure 2, upper left) in an off-road environment. The vehicle drove at about 1 m/s for 92 m. Table VI summarizes track breakup occurrence. In this experimental run, target A moved always ahead of the vehicle, while periodically crossing from one side to another. Targets B and C always remained behind and close to the XUV (less than 4 m), and were never occluded nor significantly affected by clutter. Target D followed the XUV from slightly farther away and eventually walked across tall grass areas, to the point of being lost in the clutter for extended amounts of time. As shown in the table, the system performed well, reporting valid velocity estimates 96%, 79.5%, and 96.4% of the time for targets A, B, and C, respectively. Similarly, there were few breakups for these three targets, being as low as 4 for target A and as high as 10 for target B. Target D was frequently occluded or cluttered by the tall grass and suffered as many as 57 breakups, which precluded the computation of valid estimates more than 51% of the time. At some point, the system assigned a new track for this target every 0.1 s, since the target walked too close to a patch of tall grass, even though the scanners had an unobstructed view of it.

7. DATMO INTEGRATED INTO LIVE SYSTEMS

Our DATMO has many different applications, as can be seen by the multitude of platforms that used it (Figure 2). In the following sections, we want to describe six different systems. The ones described in Sections 7.1 and 7.3 used SICK laser scanners, whereas the one in Section 7.2 used a 3D LADAR. Section 7.1 describes in detail a warning system and measures its performance with false warning rates. The reason for the false warnings and how they relate to our DATMO are explained. Section 7.2 shows how our DATMO can be used to classify objects as humans and nonhumans. Our DATMO was also employed on an autonomous vehicle for the DARPA Urban Grand Challenge. It was part of several subsystems that detected and tracked moving objects. DATMO's role and function on this vehicle are described in Section 7.3.

7.1. Side Collision Warning System (SCWS) for Transit Buses

In a joint project, University of California PATH and Carnegie Mellon University Robotics Institute developed front and side collision warning systems for transit buses. The purpose of these systems was to warn the bus driver about dangerous situations and thereby avoid collisions. For the side system, two SICK sensors and DATMO were employed. Detailed descriptions of the system and the results can be found in the references PATH and CMU (2006). The locations of the sensors and the computers are shown in Figure 12. On the left and right side of the bus was one retractable SICK LMS. On each upper corner was a video camera. The cameras helped us to analyze collected data, but they were not involved in creating warnings for the bus driver. We also had a laser line striper (Aufrere et al. 2003) consisting of a laser and a camera in the front right bumper. This was used to detect the curb next to the vehicle. The computers were housed in a compartment below the bus driver.

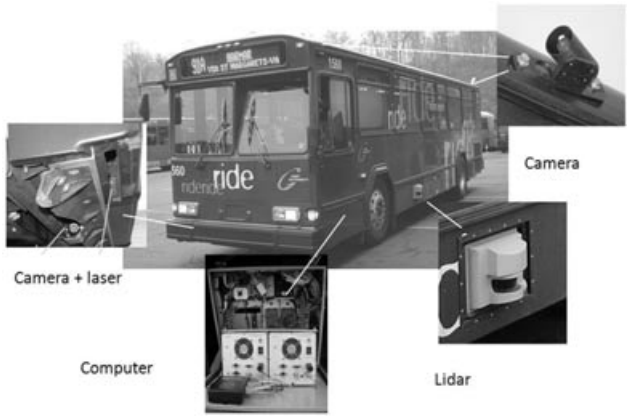

Figure 12. Sensors and computers around the bus.

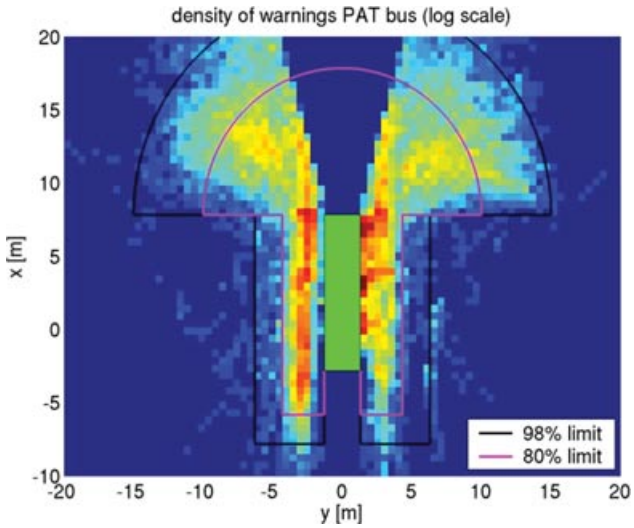

Figure 13. Density (log scale) of warnings around the bus.

7.1.1. Range of Laser Scanner

We wanted to determine the maximum range that the laser scanner needs to cover to detect all the dangerous situations the bus is exposed to during normal operations. For this, we created a density plot of the location of all the warnings, based on the warnings from all good runs. The density plot for the PAT (Port Authority Pittsburgh) bus is shown in Figure 13, where the highest density is dark red and the lowest is dark blue.

Most of the warnings are located alongside the bus. On the right side are mostly pedestrians, who are moving toward the bus when the bus is coming to a stop. On the left side, most of the warnings are caused by passing vehicles. For the PAT bus there is a high concentration of warnings in the middle of the bus adjacent to the right side.

The latter is the location of the laser scanner, and these warnings were generated when the laser scanner was dirty. The warnings in the front right or front left area of the bus are generated when the bus is turning and an object is in the path of the bus.

The figure also shows two enveloping areas, which include 80% and 98% of all warnings, respectively. They can be described as rectangular boxes on the side of the bus extending 3 m (5 m) from the back and 3 m (5 m) to the side and a half-circle in front with a radius of 10 m (15 m). The system covers the area of a half-circle of 50 m radius for large objects (>1 m as viewed from the scanner), which is much larger than the area indicated by the enveloping limits. For pedestrian-sized objects, which are harder to detect and track, the coverage is approximately a half-circle of 20 m radius, which still includes the enveloping area. We can therefore be quite confident that we did not miss warnings because of a lack of coverage.

The enveloped areas give an indication of what the coverage of a commercial system should be. It is desirable for the sensor to have a range somewhat greater than the indicated area, because this enables the detection and tracking of objects before they enter the area.

7.1.2. SCWS Warning Algorithm

The sensors and modules described in the previous sections provide the dynamic quantities of the bus and the observed objects and additional information about the environment. These measurements are combined with preloaded information to analyze the threat level of the situation. In the warning algorithm, the system calculates the probability that a collision will occur within the next five seconds. If the probability of collision exceeds a certain threshold, an appropriate warning is displayed to the driver. In the warning algorithm for the SCWS, we have two warning levels: "alert" and "imminent warning." An "alert" is displayed to the driver when the situation is somewhat dangerous, an "imminent warning" is given if the situation is dangerous enough to inform the driver in an intrusive way. A detailed description of the algorithm can be found in Mertz (2004). A short example is illustrated here.

In Figure 14, a bus turns right while an object crosses its path from right to left (World). The sensors measure the speed and turn rate of the bus and the location and velocity of the object. The algorithm calculates possible paths of the object with respect to the bus (Fixed bus). In this calculation, the paths are distributed according to the uncertainties of the measured dynamic quantities as well as according to models of driver and object behavior. These models are limits on speed, acceleration, turn-rate, etc. and were determined from previously recorded data. Next, the system determines for times up to 5 s into the future which fraction of these paths lead to a collision. In Figure 14, this is shown

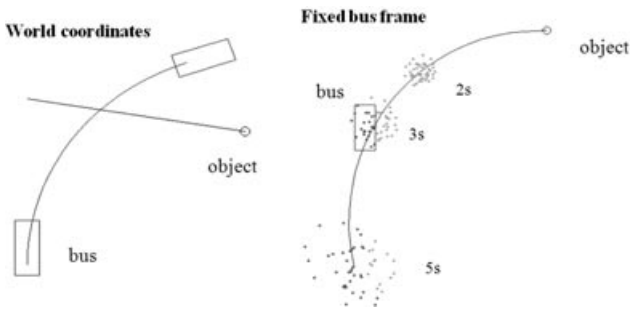

Figure 14. The trajectories of a bus and an object shown in the world coordinate frame (left) and the fixed bus frame (right). In the right figure, possible positions of the object are shown for the times 2, 3, and 5 s in the future. Light gray indicates that no collision has happened; dark gray indicates that a collision has happened.

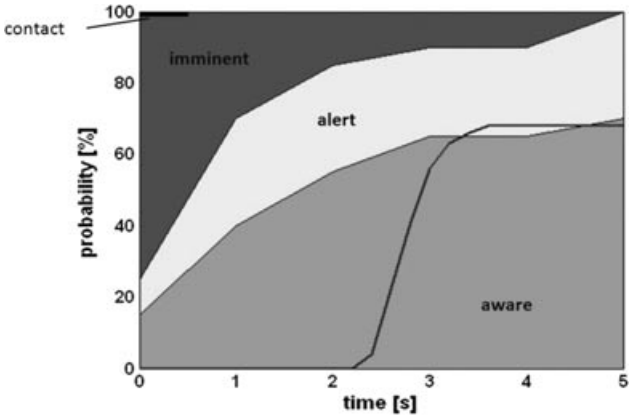

Figure 15. Probability of collision plotted versus time. The three regions correspond to the warning levels aware, alert, and imminent.

for the times 2, 3, and 5 s. This fraction is the probability of collision and is plotted versus time in Figure 15. This graph is divided into three areas, each a different level of threat severity. The area with the severest level that the probability of collision curve reaches determines the warning issued to the driver.

The algorithm can also deal with environmental information. For example, if the object is a pedestrian and is on the sidewalk, there is an enhanced likelihood that the pedestrian will stay on the sidewalk. This is addressed by giving the paths leaving the sidewalk a lower weight.

Underbus warning. Another important alarm is the underbus warning. It is issued when a person falls and has the potential of sliding under the bus. We detect these situations by observing pedestrians who disappear while being close to the bus. The challenge in this algorithm is to distinguish people who disappear through falling and people who only seem to disappear, but in fact either merged with another object or are occluded by other objects.

Notification that a collision occurred. Sometimes the bus can collide with an object, especially a person, and the driver does not notice it. It is therefore important to notify the driver if a collision has occurred. A notification will be triggered if the probability of collision is 100% for times up to 0.5 s.

7.1.3. Missed Warnings

Alert and imminent warnings. The false negative alarm rate (missed warnings) for alerts or imminent warnings is difficult to determine because it is time-consuming to review large sets of data to find situations when warnings should have been given. Instead, we staged collisions to determine how many the system missed. We did not observe any missed warnings in these staged scenarios, which puts an upper limit of 16% on the ratio of missed warnings to correct warnings.

Failures that we observed in other situations that could lead to missed warnings are:

- The system needs time to start up, so during this time no warnings can be issued.
- The system reboots for some reason.
- The laser scanner is retracted.
- The laser scanner is dirty.
- The bus is leaning and therefore the laser scanner does not point horizontally.
- Some objects do not reflect the laser light sufficiently. For example, we observed a situation when a person was wearing a dark outfit on a rainy day and was not seen by the scanner. It could be that wet clothing specularly reflected the laser light or the dark clothing absorbed it.

Contact warnings. Initial evaluation showed that this algorithm is too restrictive, since objects colliding with the bus at small velocities do not trigger a contact warning. The system always gives an object an uncertainty in position and velocity, so the probability-of-collision calculation will not give a result of 100% unless the velocity of the object is sufficiently large and aimed at the bus.

Under the bus warnings. We used the data from these staged scenarios to test and calibrate the under-the-bus warning algorithm. We found that we needed to modify the tracking module (DATMO) to determine if an object went into occlusion or merged with another object. This information was passed to the warning algorithm so that it did not falsely think an object disappeared (and potentially was under the bus) when it in fact became occluded or merged with some other object. We also discovered that we should only consider objects that were detected for at least half a second to suppress alarms for spurious objects. Lastly, objects that are as far as 1.8 m from the bus when they disappear need to be considered. After these modifications and tuning of parameters, all 12 staged events gave correct under-the-bus warnings.

There were not enough staged events to determine a reliable rate of false negative under-the-bus warnings. False negative warnings are possible when a person falls while being occluded by another object or the last part seen by the sensor is close to another object and merges with it. Another possibility is that a person falls under the bus at the front door when the door is open. The system excludes these situations because people routinely disappear from the view of the sensor at that location when they enter the bus.

7.1.4. False Positive Warnings

Alert and imminent warnings. When we interviewed operators about the warning system, they stated that although

Table VII. True and false positive warnings.

	Absolute				Relative (%)				Rate (1/h)			
	Alert		Imminent		Alert		Imminent		Alert		Imminent	
	Right	Left	Right	Left	Right	Left	Right	Left	Right	Left	Right	Left
True	60	94	15	9	59	71	47	26	12.0	18.8	3.0	1.8
Vegetation	10	3	2	0	10	2	6	0	2.0	0.6	0.4	0.0
false velocity	21	28	10	20	21	21	31	57	4.2	5.6	2.0	4.0
no velocity	0	2	1	0	0	2	3	0	0.0	0.4	0.2	0.0
ground return	10	4	3	3	10	3	9	9	2.0	0.8	0.6	0.6
Other	1	2	1	3	1	2	3	9	0.2	0.4	0.2	0.6
Sum	102	133	32	35	100	100	100	100	20.4	26.6	6.4	7.0

the collision warning system is acceptable to them, they still would like a lower false warning rate. The false warning rates discussed below should therefore be considered in the context of an acceptable system, while recognizing that a reduction in the false warning rate is desirable.

We reviewed all alerts and imminent warnings in two runs and determined if they were true or false. One of the runs took place in California and the other in Pennsylvania, and together they were five hours long. We wanted to gather enough data to ensure that we observed the main categories of false warnings. We collected at least 30 warnings for each warning level and each side to ensure an upper limit of 10%, i.e., false warning categories that were not observed have a rate of less than 10% (90% confidence level). Table VII shows the absolute number of warnings, the relative number for each category (percentage of the total number of warnings), and the warning rates, for the left and right sides.

The most common situations that cause true warnings are vehicles passing and fixed objects in the path of a turning bus. On the right side there are additional true warnings caused by pedestrians entering the bus or walking toward the bus when the bus has not yet come to a full stop.

A majority of the alerts are true alerts, whereas a majority of the imminent warnings are false positives. The most common reason for false imminent warnings is that the velocity estimation was incorrect, but as explained below this kind of error is not very serious.

Vegetation. The system cannot distinguish between vegetation (grass, bushes, and trees) and other fixed objects, but the threat posed by vegetation is much smaller than other objects because a bus can come in contact with grass, leaves, or small branches without being damaged. A warning triggered by vegetation can often be considered a nuisance warning. This is the least serious kind of system error because the system functions correctly, but the warning is considered a nuisance. Figure 16 shows a situation when the bus comes close to a bush and an imminent warning is triggered. On the right side, the four images from the four side cameras (see Figure 12) are shown. The bush can be seen in the upper right image, overlaid with a dotted and a thick white box, indicating an alert and an imminent warning for that object. Those boxes can also be seen in the bird's-eye view of the situation on the left. Notice that part of the bush extends over the curb. If an object is off the curb, the warning algorithm will give it a higher probability of collision than if it is on the curb.

False velocity estimation. The velocities of the objects are determined by our DATMO algorithm. Figure 9 shows the error distribution of the velocity estimation in the x and y direction from that report. The distribution is characterized by a Gaussian shape plus some additional outliers. The false velocities that give false warnings are from the tail of the Gaussian distribution or are outliers. An example of a case when a slightly incorrect velocity estimation leads to an alert is shown in Figure 17. The vehicle can be seen in the lower left image with a dotted box on it, indicating an alert. The dotted box also appears in the bird's-eye-view display.

The incorrect velocity estimation increases the probability of collision by enough to cross the warning threshold.

Figure 16. Overhanging bush is close enough to trigger an alert (dotted box) and an imminent warning (thick white box).

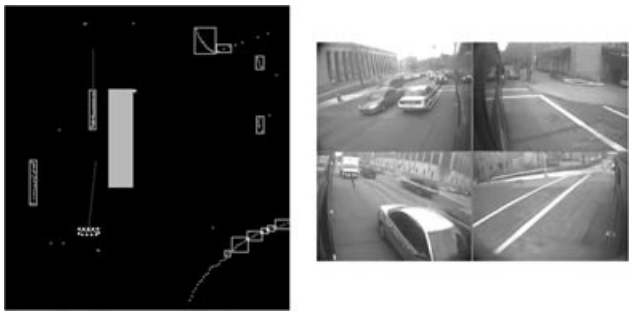

Figure 17. The velocity estimation of the vehicle is slightly off, leading to an alert (dotted box).

It needs to be mentioned that this kind of error is not very serious because the danger level was only slightly overestimated. In most of the cases when an imminent warning was issued because of a false velocity estimation (such as Figure 17), the correct warning level would have been an alert.

No velocity information. Our DATMO needs some time to determine the velocity of an object after its initial detection (see the "velocity delay" discussion in Section 6.2). During that time, the system ignores the velocity of the object, i.e., sets it to zero. In some cases, this can lead to a false warning, especially if the object is in the path of the bus and moving away from the bus. This error is of medium seriousness because an object is present but the threat level is misjudged.

It is possible to avoid these false warnings by waiting until the velocity of the object has been established, but this would introduce latency and therefore false negative warnings.

Ground return. The laser scanners will see the ground if the bus rolls or pitches or if the ground in the vicinity of the bus slopes upward with respect to the plane on which the bus stands. Depending on the location of the ground seen by the sensor, the system might issue a warning, as shown, for example, in Figure 18.

Figure 18. Ground returns seen as an object in the left front of the bus (thick white box).

In this case the bus is turning left. The laser scanner sees the ground and the system thinks an object is directly in the path of the bus and issues an imminent warning. In the left upper image, the ground return is indicated as a thick white (imminent warning) box and in the bird's-eye-view display it is the thick white box in the upper left corner. This is the most serious false positive, because a warning is issued when there is no threat whatsoever.

Other reasons for false positives. There are many other reasons for false positive warnings, which can vary greatly in their frequency from run to run. For example, in some runs a malfunction of the retraction mechanism misaligned the laser scanner and resulted in hundreds of false warnings. Some of the reasons were easily eliminated after their discovery, but they are listed here for completeness.

Retraction malfunction: When the laser scanner is retracted, a switch should signal this fact to the system. In some cases the switch malfunctioned and the sensor was retracted without the system knowing about it.

Dirt on the scanner: Dirt on the laser scanner appears as an object very close to the bus. This problem can be solved by discarding all scanner data points very close to the scanner itself.

Scanner sees the bus: The laser scanner can see parts of the bus if the scanner is slightly misaligned or if some parts of the bus protrude, such as an open door or a wheel when the bus is turning. This problem can be solved by excluding returns from certain areas, but this has the side effect that these areas are now blind spots.

Error in DATMO: There are many ways that our DATMO algorithm can make mistakes. The most common one was already mentioned above, this being the incorrect estimation of the velocity of an object.

Splashes: Splashes of water can be seen by the scanner and trigger a warning. In Figure 19, one can see the development of such a splash, indicated by a circle in the middle image. The outline of the bus is on the left in gray. The splashes are seen by the sensor for only about 0.2 s, but this is enough to be registered as an object and to trigger an alert.

Noise in turn rate: The turn rate of the bus is measured by a gyroscope, which has some noise, so there is a small chance that it gives an erroneous value. Very few cases were observed when these errors led to a false warning.

Dust: A cloud of dust can be produced by the wheels, appearing as an object to the system.

7.1.5. Contact and Under-the-bus Warnings

The rate of contact warnings is very low, about 0.4 warnings/h on the right side and 0.1 warnings/h on the left side. All the contact warnings we observed during normal operations were false positive warnings, i.e., no actual collisions occurred. These warnings were not directly related to our

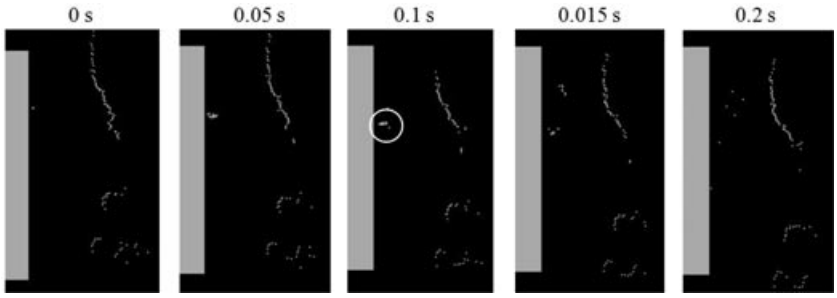

Figure 19. Example of a splash of water appearing on the right side of the bus.

DATMO. They were caused by two different indicator failures. One was the indicator for the retraction of the laser scanner and the other was the bus-door-open indicator. Either indicator would have vetoed the collision warning.

For the under-the-bus warnings, we observed a false positive rate of 1.9 per hour. About half of those were caused by similar indicator failures to those for contact warnings. Most of the rest were cases when the system was not able to interpret a cluttered scene correctly. In two cases, the bus tires created a small dust cloud that disappeared after a short time and triggered a warning.

The total false positive warning rates of 0.5 contact warnings/h and 2 under-the-bus warnings/h are still too high because either warning requires drastic actions from the bus driver, namely stopping the bus and investigating what happened. We therefore did not activate and display these warnings for the operational testing in public service.

The fact that we did not observe any correct under-the-bus warning is no surprise, since people falling under the bus is an extremely rare event. The system would benefit from additional sensors that can positively identify that something is underneath the bus or that something hit the bus.

7.2. Human Detection with CTA System

In the CTA project we used 3D scanners, as seen in Figure 2 on top of the XUV and the Suburban. For this application, the 3D data were converted to a 2D scan line using the procedure described in Section 5. This conversion is done to eliminate ground returns, and also to reduce the computational cost of processing the point cloud. Consequently, object detection and tracking can be done very efficiently. We have achieved rates up to 15 Hz when processing up to 10,000 points per frame coming from two 3D scanners. But in addition to detection and tracking, the system can also classify objects. In this case, we were specifically interested in determining which objects are humans: one of the most important goals of any autonomous system is to avoid colliding with people.

We have developed a simple human detection algorithm, which is described in detail in Navarro-Serment et al. (2008). This algorithm operates as follows: Once the 2D scan line is obtained, and after the DATMO process has been executed, four basic measures are determined for each object being tracked: the object's size, the distance it has traveled, and the variations in the object's size and velocity. These measures are tested to score the extent to which the target is believed to be a human. The size test eliminates large objects such as buildings and vehicles. The distance traveled test discriminates against stationary objects such as barrels and posts. Finally, both noise variation tests discriminate against vegetation, since their signature in the scan line typically changes considerably due to their porous and flexible nature.

All objects are evaluated by computing their *Strength of Detection* (SOD), based on the four measures described before. The SOD is a measure of how confident the algorithm is that the object is actually a human. The SOD is the product of the scores of the size and distance traveled test, and the square root of the scores of the variance tests. The objects evaluated with a SOD larger than a given threshold are classified as humans. A sample result is illustrated in Figure 20, where a snapshot of the point cloud contained in one frame is shown. Tracked objects classified as humans are colored from green (low) to red (high) according to their SOD; objects in blue have an SOD of zero. The rest of the points, which do not belong to any object being tracked,

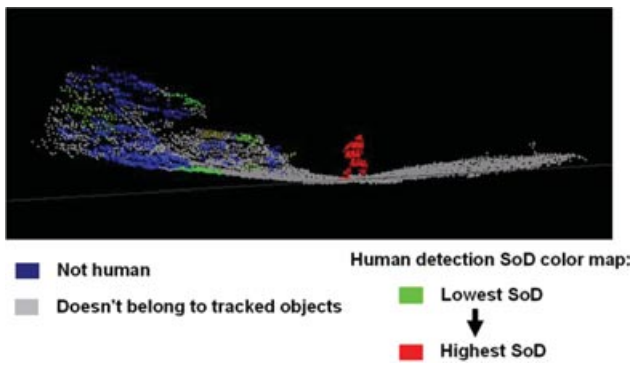

Figure 20. Segmentation of points based on tracking information.

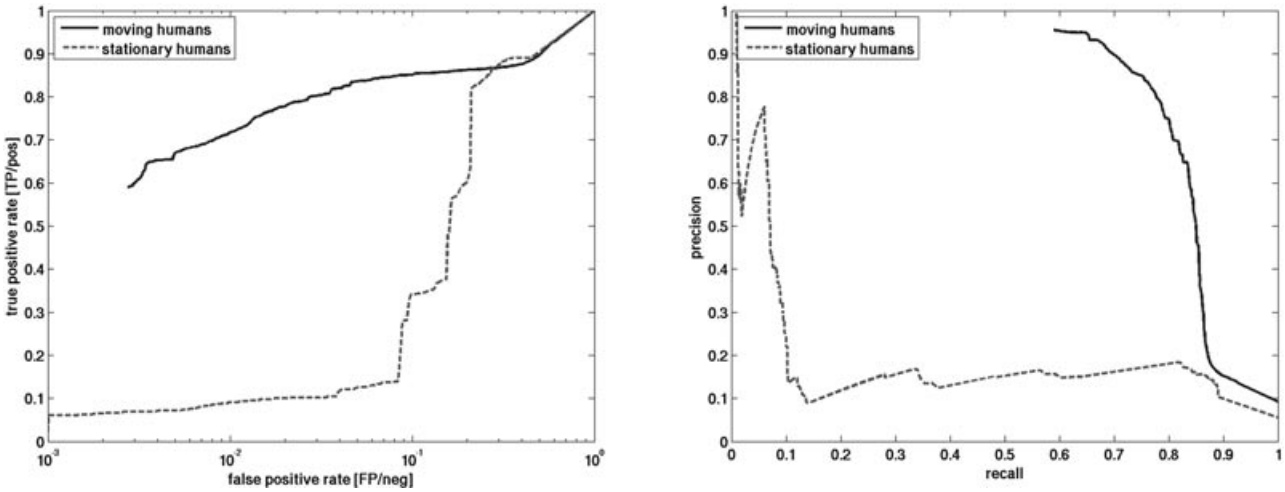

Figure 21. ROC and precision-recall curves for human classification, comparing performance for static and moving humans. The data used to produce these plots come from various runs using the GenIII LADAR. The variations included static and moving vehicles, pavement and off-road driving, and pedestrians standing, walking, or jogging.

are colored in gray. One can see that the human has a high SOD whereas the other objects have a low SOD or an SOD of zero.

To evaluate our detection algorithm, we hand-labeled the objects reported by DATMO as humans and non-humans and used this as ground truth to calculate the performance curves. Figure 21 shows ROC and the precision-recall curves for both static and moving humans. Each object in each cycle counts as one example. For the plot with moving humans, there were 1,793 cycles, 5,605 positive, and 55,241 negative examples. For the static humans, there were 380 cycles, 577 positive, and 9,935 negative examples.

This classifier performs fairly well when dealing with moving humans. It has a 75% precision (or 2.7% false positive rate) at 80% recall (or true positive rate), but it has difficulties with stationary humans, as can be seen in Figure 21. This issue was the focus of subsequent research, where a classifier was designed to exploit the information contained in the 3D point cloud of each object (Navarro-Serment et al., 2010). This work demonstrated improved classification performance of both moving and static pedestrians. The improvements were most significant in the case of static humans. Alas, given the strict 3D nature of this algorithm, they cannot be applied to 2D laser scanners.

7.3. Urban Grand Challenge System

Our DATMO played an important part of the detection and tracking of moving cars in Tartan Racing's entry into the Urban Challenge in 2007 (Urmson et al., 2008). Boss, the robot fielded by CMU, had a wide range of sensors that were used for its perception systems, including eight planar LADARs, two point LADARs, one 3D LADAR, and five RADARs. For the detection and tracking of moving obstacles perception subsystem, Boss made use of three SICK planar LADARs mounted on the bumpers (two in front and one in back), two IBEO AlaskaXT planar LADARs mounted on the front bumpers, two point LADARs that were designed for automatic cruise control (ACC) applications, and five long-range RADARs. Each sensor type used in the moving obstacle tracking subsystem passed its raw data through a series of filters and data interpretation systems that were responsible for converting the data into a form that was compatible with the other sensors. The data generated from each of these sensors were then fused together into a unified multitarget tracking module that reported the existence and statistics of all moving obstacles around Boss to the behavior and planning subsystems (Darms et al., 2009).

In the early stages of Boss's development, the only sensors to be integrated into the system were the three planar SICK LADAR mounted on the robot's bumpers. Before development began on the multisensor moving obstacle fusion system, our DATMO was used to interpret the laser point readings from the SICKs. The laser data from all three SICKs were fused into a single 2D point cloud and this was given to our DATMO for processing. DATMO clustered the points into edge and corner-shaped targets, computed their velocities, and returned this to the rest of the system. This worked fairly well in the initial stages of testing, however there were some immediate shortcomings. First, our DATMO is optimized for fitting lines and "L"-shaped corners to moving obstacles such as cars. When Boss was following a vehicle on the road, the front two bumper SICKs would be able to see both rear corners of the vehicle. This

created a "U"-shaped target in front of the robot that sometimes did not resolve properly in our DATMO system. The second issue was that our DATMO generates very conservative estimates for an object's velocity. This is to ensure that when a velocity is reported, it is accurate and reliable. However, for an application like the Urban Challenge, a reliable velocity estimate was needed almost immediately once a moving object was detected.

The decision to develop a more complex moving obstacle fusion system above and beyond the capabilities provided by our DATMO was three fold. First, our DATMO had no mechanism for processing non-LADAR data from sensors such as a RADAR. Secondly, the configurations of the SICKs on the front of Boss did not lend themselves well to generating a single set of laser points for our DATMO to work with because of the overlap in the front that would generate "U"-shaped targets. Finally, rapidly obtaining an accurate estimate of a target's velocity once it was detected was a high priority for the team. The third issue would be addressed by fusing the Doppler RADARs into the system, which provided a direct measurement of velocity along the axis of the emitted RADAR energy.

To combat the problem of the "U"-shaped targets, the data from each planar SICK were fed into its own instance of our DATMO, which clustered the points into edge and "L"-shaped targets. These targets were fed into the moving obstacle fusion system and were merged with the other targets generated from the other sensors. Ultimately, the velocity estimate from our DATMO was not used as velocity was estimated using the Kalman filters in the moving obstacle fusion system. As stated before, direct velocity measurements taken by the multiple RADAR assisted with obtaining a rapid and accurate estimate of target velocities.

One of the important functions provided by our DATMO was the clustering of the LADAR points into lines and "L"-shaped targets. The moving obstacle fusion system made use of two different kinematics models for tracking cars. The first is known as the "simple bicycle model" and is a center-pivot steering model. The second is a simple point model and is used by the RADARs as the targets returned by those systems have no shape. When our DATMO returns a line or an "L"-shape because the plane of the LADAR traces along the side and/or the front of the car, the simple bicycle model is used (Figure 22). Without the shape of the object being resolved by our DATMO, this model could not be used as the velocity vector of the object is directly coupled to its orientation.

Below is a sequence of images that illustrates the different stages of the moving obstacle fusion system used by Boss in the Urban Challenge.

Figure 23 shows a representation of the raw data returned by all of the different sensors. Individual laser points highlight not only the nearby cars but also the edges of the road. The diamond-shaped objects in this image are targets returned by the RADARs.

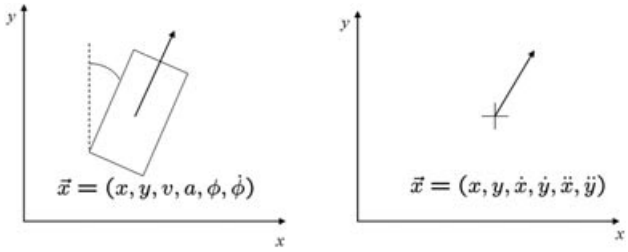

Figure 22. Tracking models used by the moving obstacle fuser. On the left is the simple bicycle model, which relies on the shape of the tracked object to determine its heading. On the right is the simple point model used when the shape of the tracked car cannot be resolved (Darms et al., 2008).

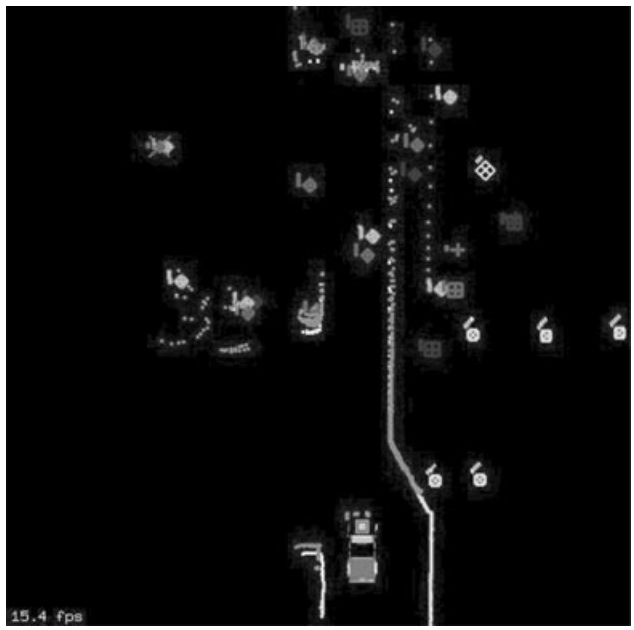

Figure 23. A top-down virtual view showing Boss (lower center) and the raw data from its forward-mounted LIDAR and RADAR sensors. Point clouds from the LIDAR sensors are shown as dots. RADAR targets are shown as boxes and diamonds. A set of curbs is visible in the LIDAR data.

The next step is to use modules such as our DATMO on the SICK LADARs and the proprietary algorithms used by the IBEO LADARs to cluster the individual LADAR points into a sequence of line and "L"-shaped targets (Figure 24). All other LADAR points are discarded as belonging to non-car objects.

After clustering the LADAR points, all targets, regardless of whether they were from LADARs or from RADARs, were validated to remove false positives. For the Urban Challenge domain, all moving objects of interest were found on the roads. Thus, any target that did not appear on

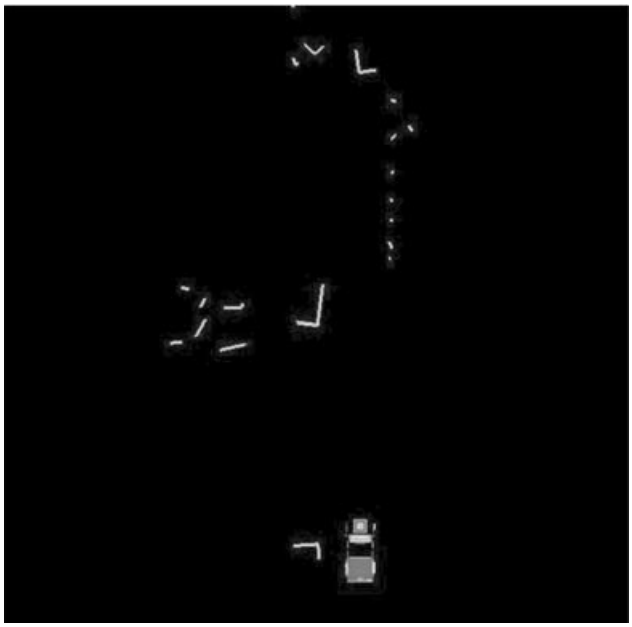

Figure 24. Lines and L-shaped targets.

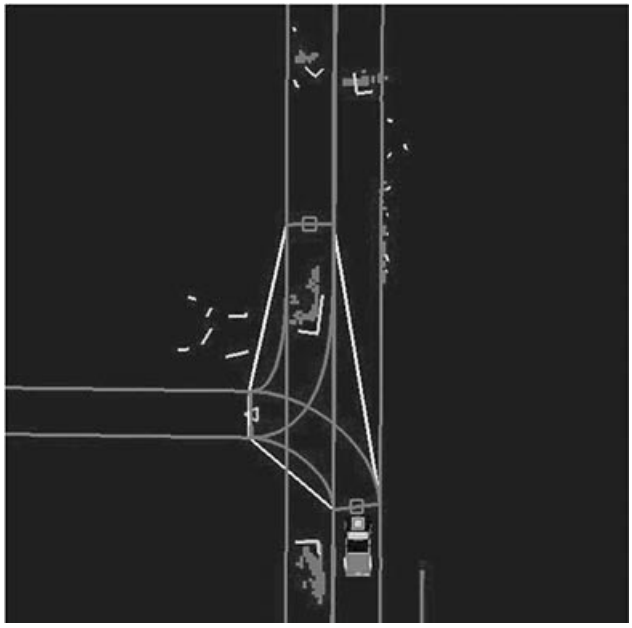

Figure 25. Target on and off the road.

the road world model (illustrated under Boss in Figure 25) was filtered out from consideration. A second filtering step was employed whereby the points returned by the Velodyne 3D LADAR were used to check whether a line target, an "L"-shaped target, or even a RADAR target actually belonged to a car. If the 3D point cloud corresponding to the same location as the line points did not have the expected volume and shape, then that potential target would also be removed from consideration.

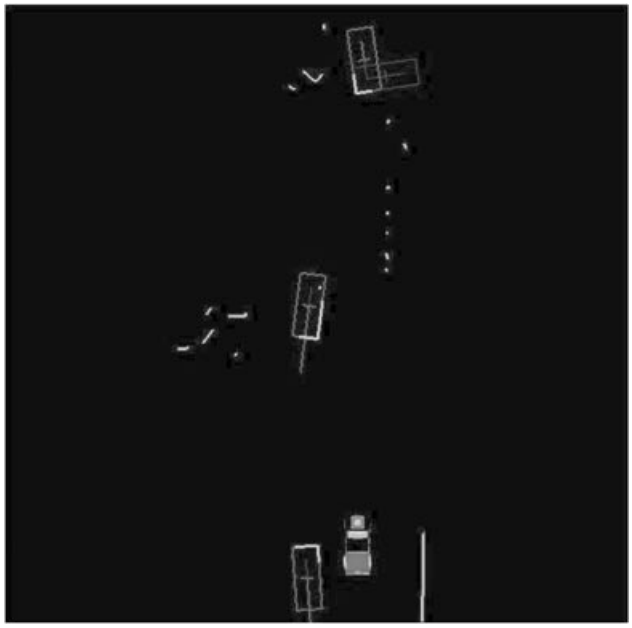

Figure 26. Lines and L-shapes merged with existing targets.

Each of the targets that survived the filtering process would then be associated with existing targets already being tracked, as shown in Figure 26. The line and "L"-shaped targets were compared against the known shape of the different targets to determine the proper alignment of the measurements. Any measurements that did not correspond to a preexisting target would be a candidate for the creation of a new target.

Finally, the set of measurements would be applied to the target list, and statistics about the position, velocity, and movement history would be generated for each target (Figure 27). This information then was passed on to the behavior and planning modules.

8. ADDITIONAL CONFIGURATIONS AND APPLICATIONS

Two more systems with 3D scanners are found in Sections 8.1.1 and 8.1.2, each with different scanning patterns and update rates from the one in Section 5. The DATMO results were used for a dynamic planner and for human detection. With the final system in Section 8.2, we have a fixed indoor setting where we demonstrate the monitoring of a wider area over long periods of time and the use of multiple sensors.

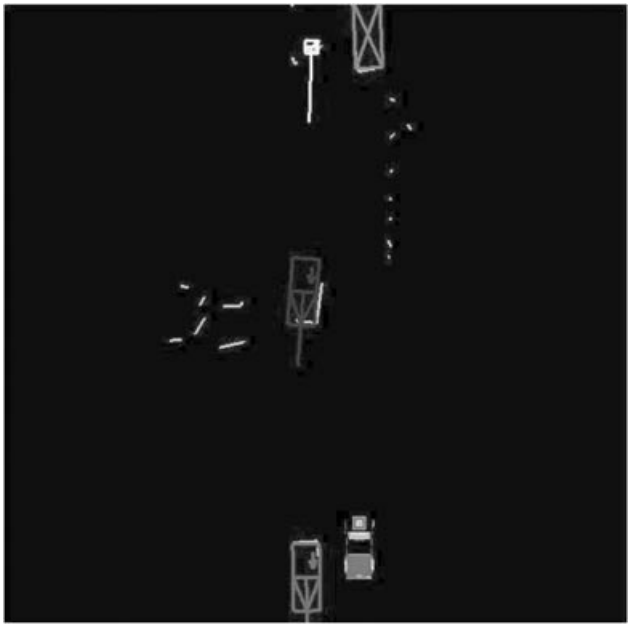

Figure 27. Final list of objects.

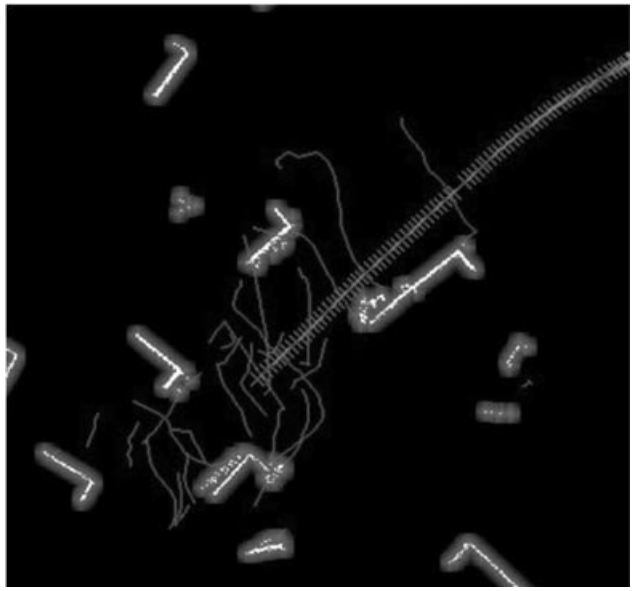

Figure 29. Fixed objects (white) and tracks (gray) observed by the nodding SICKs and our DATMO. The path of the host vehicle is shown as X's.

Figure 28. Picture of a situation seen by the nodding SICKs.

8.1. NREC Data

The Natinal Robotics Engeneering Center (NREC) provided us with data containing people and vehicles that we analyzed off-line. In Figure 29, one can see a bird's-eye view of the scenario derived from the data. The white objects are the stationary objects, mostly containers, visualized as a cost map. The tracks of the people are red lines and the path of the vehicle are the purple X's.

8.1.1. Nodding SICK

The nodding SICKs are shown in Figure 2 in the lower left. One of the scenarios seen by the sensors can be seen in Figure 28. We analyzed these data as part of a project where we wanted to show that data from such a sensor arrangement can be used for a dynamic planner. In this particular run, there were only pedestrians and no moving vehicles.

The points from one nodding cycle were accumulated into one point cloud. The point cloud was treated like the point cloud from the Gen III LADAR (Section 5), the ground points were removed, and the slice was taken from the remaining data. The resulting tracked objects are shown in Figure 29.

The main difference between this system and the Gen III LADAR is that the update rate is very low, only about 2 Hz (Table III). Also, the point pattern is different from the Gen III LADAR. One effect of the low update rate is that there is some smear in the point cloud when a person is moving. But as is evident in Figure 29, we are able to track the people over long distances so that we are able to feed these data into a dynamic planner. The paths of people that overlap static objects are cases when an object is kept alive for a while after it goes into occlusion. The static objects were used to build a static cost map. In Figure 29, the costs are indicated with gray scale. Black is no cost and white is highest cost. The updates of position and velocity of the dynamic objects were passed to the planner at each cycle.

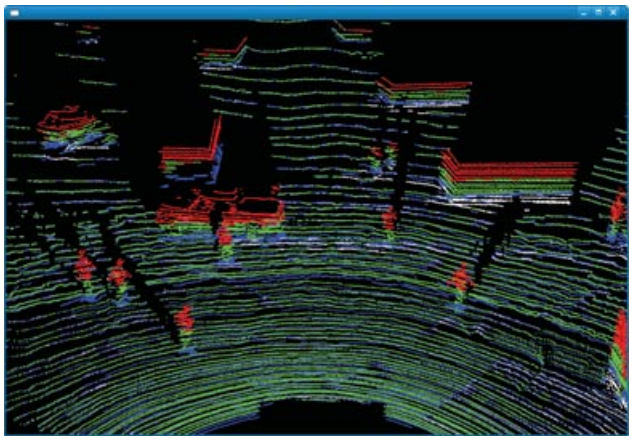

Figure 30. A snapshot of the 3D Velodyne data. The estimated ground plane is blue, points below the ground are white, above it green, and the slice is red. One can see several humans, two vehicles, and containers.

8.1.2. Velodyne

Velodyne data were recorded at the same time as the data for the nodding SICKs. The Velodyne sensor has a 360° horizontal field-of-view. However, in this case the back view was obscured by other equipment and we only used the front 180°. An example of a point cloud is shown in Figure 30. One can see humans, two vehicles, and rectangular containers.

The 3D data were analyzed as described in Section 5, the ground points were removed, and the slice was taken from the remaining data. This can be seen in Figure 30, where the ground plane is indicated by blue points, points below the plane are white, those above the plane are green, and the slice is red. Next the slice is projected into 2D and analyzed by the detection and tracking algorithm. It should be pointed out that we did not have to adjust any parameters to make the system work except the field-of-view.

One measure on how well the system works with these particular data is to evaluate its classification performance, as was done for Gen III LADAR data in Section 7.2. Objects are classified as humans or nonhumans by looking at their size, movement, and noise characteristics. As in Section 7.2, we hand-labeled the objects coming to get the ground truth and calculated the performance curves. In Figure 31, one can see the ROC and the precision-recall curves. They are plotted for distance smaller and greater than 35 m. Each object in each cycle counts as one example. There were 943 cycles. For distances less than 35 m, we had 4,468 positive examples and 10,883 negative examples; for those greater than 35 m, we had 644 positive examples and 13,562 negative examples.

The curves for distances smaller than 35 m are qualitatively similar to the ones in Figure 21 and thereby indicate that our DATMO gives similar results to these two different 3D LADARS. A more detailed quantitative comparison is not possible, because the scenarios were different. The run with the Velodyne had no vegetation or stationary humans, but instead containers and more vehicles.

We were able to detect humans up to 60 m. The classification is better at shorter distances, which is not surprising. One can see the difference in Figure 31, where we plotted the ROC and the precision-recall curves for distance smaller and greater than 35 m.

8.2. Indoor

Our DATMO system can also be used indoors in a fixed setting. In contrast to scanners mounted on a vehicle, which

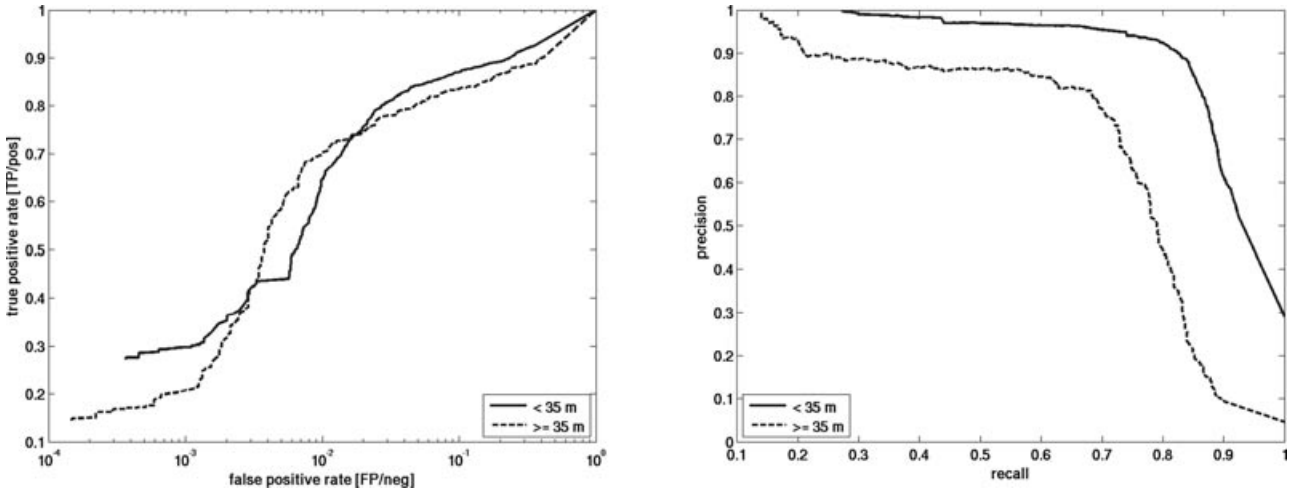

Figure 31. ROC and precision-recall curves for human classification with Velodyne LADAR.

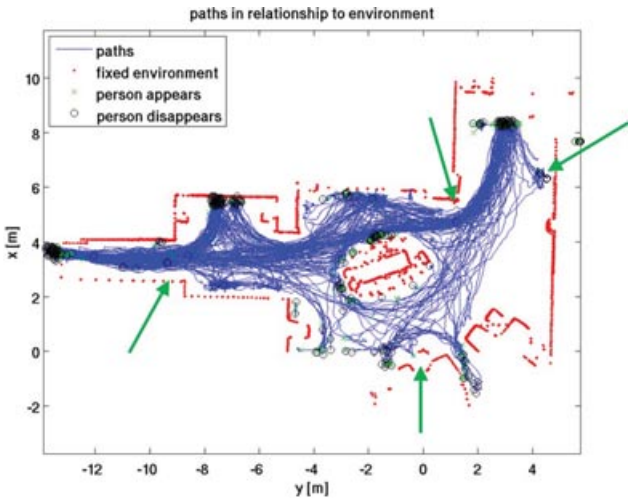

Figure 32. The tracked paths in the indoor environment. The locations of the SICK sensors are indicated by the green arrows. The fixed objects (walls, furniture, etc.) are marked in red. The paths of individual people are shown in blue. The locations where people appear and disappear are indicated with green X's and black O's, respectively.

are relatively close together and face away from each other, the scanners in this setup are distributed over a wide area and face each other. Figure 2 (lower right) shows the indoor environment; one of the SICK sensors can be seen. The locations of the other three are indicated in Figure 32. That figure displays the tracks from one run. The significance in this setup is that we were able to track people walking around the island in the middle of the room without interruption. The people were handed off from one sensor to the next. This was done by fusing the sensors at the segment-to-object level (see Section 4.2). Each scan from each sensor was segmented individually and the segments were used to update an object list that was common to all sensors.

We collected data in this setting almost continuously for over a month. The only interruptions were the periodic retrieval of the data. The tracks generated by our DATMO were used in a project to learn and predict the behavior of people using a Markov decision process framework (Ziebart et al., 2009). To select only the tracks that belong to single humans, we rejected tracks that merged or split with other tracks and we required a high score in the human classification (see Section 7.2). In Figure 32, one can see the tracks for one run. One can also see the locations where tracks started (green "X") and terminated (black circle). These locations are used as the origins and possible goals in the Markov decision-process framework. Our system provided high-quality data to train and test the prediction model.

9. CONCLUSION AND OUTLOOK

We have shown that our DATMO can be employed on a variety of platforms with different kinds of 2D and 3D laser scanners. The data from multiple scanners can be combined on the raw data level, on the segment to object level, or on the object level. The system is able to track on the order of 100 objects simultaneously. The applications that used our DATMO included a collision warning system, pedestrian detection and classification, autonomous driving, human track learning and prediction, and input to a dynamic planner. We have also characterized the accuracy and the limitations of the system.

Our DATMO is now a tool with which we can build many more future applications. Nevertheless, there are areas where our DATMO can be improved. It would be desirable to have a more systematic way of optimizing the configuration parameters of DATMO for particular purposes. There could also be a use for a greater variety of each of the subalgorithms: ground estimation, segmentation, feature extraction, association, motion prediction, stochastic filters, and classification. In particular, one can make more use of 3D information. However, we found [e.g., in Navarro-Serment et al. (2010)] that 3D algorithms need a lot of computational time and one needs to be careful in how to implement them while retaining the ability to run the system in real time. Another of our current research thrusts is to do much better predictions of pedestrian behavior (Ziebart et al., 2009). Better predictions will allow us to keep better track of a person who temporarily went into occlusion.

A particularly interesting and challenging topic is dynamic planning, i.e., planning in environments with moving objects. Such planning would already be difficult if the sensors gave a perfect representation of the environment. But the additional challenge for the planner is to deal with the uncertainties that will come from DATMO. For DATMO, on the other hand, the challenge is to make the right tradeoffs between different errors (over- and undersegmentation, different kinds of misassociations, false and missed detections, etc.).

ACKNOWLEDGMENTS

Several projects contributed to this work. Part of it was conducted through collaborative participation in the Robotics Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-209912. We would like to thank General Dynamics Robotic Systems for their support. This work was also supported in part by the U.S. Department of Transportation under Grant 250969449000 and PennDOT grants PA-26-7006-02 and PA-26-7006-03. We thank the Tartan team and their sponsors for their contributions. We further thank the NREC for making the Velodyne and nodding SICK data available to us.

REFERENCES

Arras, K., Grzonka, S., Luber, M., & Burgard, W. (2008). Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In IEEE International Conference on Robotics and Automation, ICRA 2008, May 19–23, 2008, Pasadena, California, USA.

Arras, K., Mozos, O., & Burgard, W. (2007). Using boosted features for the detection of people in 2d range data. In IEEE International Conference on Robotics and Automation, ICRA 2007, 10–14 April 2007, Roma, Italy.

Aufrere, R., Mertz, C., & Thorpe, C. (2003). Multiple sensor fusion for detecting location of curbs, walls, and barriers. In IEEE Intelligent Vehicles Symposium, Columbus, OH, USA, June 9–11, 2003.

Bruce, A., & Gordon, G. (2004). Better motion prediction for people-tracking. In IEEE International conference on robotics and automation, April 26–May 1, 2004, New Orleans, LA, USA.

Castro, D., Nunes, U., & Ruano, A. (2004). Feature extraction for moving objects tracking system in indoor environments. In 5th IFAC/EURON Symposium on Intelligent Autonomous Vehicles.

Cui, J., Zha, H., Zhao, H., & Shibasaki, R. (2007). Laser-based detection and tracking of multiple people in crowds. Computer Vision and Image Understanding, 106(2-3), 300–312.

Cui, J., Zha, H., Zhao, H., & Shibasaki, R. (2008). Multi-modal tracking of people using laser scanners and video camera. Image Vision Comput., 26(2), 240–252.

Darms, M., Rybski, P., & Urmson, C. (2008). An adaptive model switching approach for a multisensor tracking system used for autonomous driving in an urban environment. In AUTOREG 2008, Steuerung und Regelung von Fahrzeugen und Motoren: 4. Fachtagung, Duesseldorf, VDI-Verlag (pp. 521–530).

Darms, M. S., Rybski, P., Baker, C., & Urmson, C. (2009). Obstacle detection and tracking for the urban challenge. Trans. Intell. Transport. Sys., 10(3), 475–485.

Enzweiler, M., & Gavrila, D. M. (2009). Monocular pedestrian detection: Survey and experiments. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(12), 2179–2195.

Fayad, F., & Cherfaoui, V. (2007). Tracking objects using a laser scanner in driving situation based on modeling target shape. In Proceedings of the IEEE Intelligent Vehicles Symposium.

Fod, A., Howard, A., & Matari, M. J. (2002). Laser-based people tracking. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).

Frank, O., Nieto, J., Guivant, J., & Scheding, S. (2003). Multiple target tracking using sequential Monte Carlo methods and statistical data association. In Proceedings IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS).

Fürstenberg, K. (2005). Pedestrian protection using laserscanners. In Proceedings of Intelligent Transportation Systems, pp. 437–442.

Fürstenberg, K., & Dietmayer, K. (2004). Object tracking and classification for multiple active safety and comfort applications using a multilayer laser scanner. In Proceedings of Intelligent Vehicles Symposium.

Gandhi, T., & Trivedi, M. (2007). Pedestrian protection systems: Issues, survey, and challenges. Intelligent Transportation Systems, IEEE Transactions, 8, 413–430.

Gate, G., & Nashashibi, F. (2008). Using targets appearance to improve pedestrian classification with a laser scanner. In Proceedings of IEEE Intelligent Vehicles Symposium, pp. 571–576.

Gate, G., & Nashashibi, F. (2009). Fast algorithm for pedestrian and group of pedestrians detection using a laser scanner. In Proceedings of Intelligent Vehicles Symposium.

Gidel, S., Blanc, C., Chateau, T., Checchin, P., & Trassoudaine, L. (2009). A method based on multilayer laserscanner to detect and track pedestrians in urban environment. In Proceedings of Intelligent Vehicles Symposium.

Glas, D., Miyashita, T., Ishiguro, H., & Hagita, N. (2007). Laser tracking of human body motion using adaptive shape modeling. In Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference, pp. 602–608.

Huang, A. S., Antone, M., Olson, E., Fletcher, L., Moore, D., Teller, S., & Leonard, J. (2010). A high-rate, heterogeneous data set from the darpa urban challenge. The International Journal of Robotics Research, 29(13), 1595–1601.

Huang, J., Lee, A., & Mumford, D. (2000). Statistics of range images. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, p. 1324.

Huang, J., & Mumford, D. (1999). Statistics of natural images and models. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Vol. 1, p. 1541.

Iagnemma, K., Buehler, M., & Singh, S., editors (2008). Workshop: The 2007 DARPA Urban Challenge: From Algorithms to Autonomous Vehicles, IEEE International Conference on Robotics and Automation (ICRA).

Jain, A., Murty, M., & Flynn, P. (1999). Data clustering: a review. ACM Computing Surveys, 31(3), 264–323.

Kämpchen, N. (2007). Feature-Level Fusion of Laser Scanner and Video Data for Advanced Driver Assistance Systems. Ph.D. thesis, Ulm University.

Kelly, A., Stentz, A., Amidi, O., Bode, M., Bradley, D., Diaz-Calderon, A., Happold, M., Herman, H., Mandelbaum, R., Pilarski, T., Rander, P., Thayer, S., Vallidis, N., & Warner, R. (2006). Toward reliable off road autonomous vehicles operating in challenging environments. The International Journal of Robotics Research, 25(5-6), 449–483.

Khan, Z., Balch, T., & Dellaert, F. (2006). MCMC data association and sparse factorization updating for real time multitarget tracking with merged and multiple measurements. IEEE Trans. Pattern Anal. Mach. Intell., 28(12), 1960–1972.

Klasing, K., Wollherr, D., & Buss, M. (2008). A clustering method for efficient segmentation of 3d laser data. In IEEE International Conference on Robotics and Automation (ICRA).

Kluge, B., Koehler, C., & Prassler, E. (2001). Fast and robust tracking of multiple moving objects with a laser range

finder. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), pp. 1683–1688.

Labayrade, R., Royere, C., Gruyer, D., & Aubert, D. (2005). Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner. Auton. Robots, 19(2), 117–140.

Lalonde, J.-F., Vandapel, N., Huber, D. F., & Hebert, M. (2006). Natural terrain classification using three-dimensional ladar data for ground robot mobility. Journal of Field Robotics, 23(10), 839–861.

Lindström, M., & Eklundh, J.-O. (2001). Detecting and tracking moving objects from a mobile platform using a laser range scanner. In Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, Maui, Hawaii (pp. 1364–1369).

Luber, M., Stork, J., Tipaldi, G., & Arras, K. (2010). People tracking with human motion predictions from social forces. In Robotics and Automation (ICRA), 2010 IEEE International Conference, pp. 464–469.

MacLachlan, R., & Mertz, C. (2006). Tracking of moving objects from a moving vehicle using a scanning laser rangefinder. In Proceedings of Intelligent Transportation Systems (ITSC), IEEE (pp. 301–306).

Mählisch, M., Schweiger, R., Ritter, W., & Dietmayer, K. (2006). Sensorfusion using spatio-temporal aligned video and lidar for improved vehicle detection. In Proceedings of 2006 IEEE Intelligent Vehicles Symposium.

Mendes, A., Bento, L., & Nunes, U. (2004). Multi-target detection and tracking with laser range finder. In IEEE Intelligent Vehicles Symposium.

Mertz, C. (2004). A 2d collision warning framework based on a Monte Carlo approach. In Proceedings of ITS America's 14th Annual Meeting and Exposition.

Monteiro, G., Premebida, C., Peixoto, P., & Nunes, U. (2006). Tracking and classification of dynamic obstacles using laser range finder and vision. In IEEE IROS Workshop on Safe Navigation in Open Environments.

Montemerlo, M., Thrun, S., & Whittaker, W. (2002). Conditional particle filters for simultaneous mobile robot localization and people-tracking. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).

Navarro, L., Mertz, C., & Hebert, M. (2006). Predictive mover detection and tracking in cluttered environments. In Proceedings of the 25th Army Science Conference.

Navarro-Serment, L., Mertz, C., Vandapel, N., & Hebert, M. (2008). Ladar-based pedestrian detection and tracking. In Proceedings 1st. Workshop on Human Detection from Mobile Robot Platforms, ICRA. IEEE.

Navarro-Serment, L. E., Mertz, C., & Hebert, M. (2010). Pedestrian detection and tracking using three-dimensional ladar data. The International Journal of Robotics Research, 29(12), 1516–1528.

Nguyen, V., Gächter, S., Martinelli, A., Tomatis, N., & Siegwart, R. (2007). A comparison of line extraction algorithms using 2d range data for indoor mobile robotics. Auton. Robots, 23(2), 97–111.

Pandey, G., McBride, J. R., & Eustice, R. M. (2011, November). Ford campus vision and lidar data set. The International Journal of Robotics Research, 30(13), 1543–1552.

PATH, U. o. C., & CMU, R. I. (2004). Integrated collision warning system final technical report fta-pa-26-7006-04.1. Technical report, UoC PATH and CMU RI.

PATH, U. o. C., & CMU, R. I. (2006). Integrated collision warning system final evaluation report fta-pa-26-7006-06.1. Technical report, UoC PATH and CMU RI.

Pellegrini, S., Ess, A., Schindler, K., & van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. In Computer Vision, 2009 IEEE 12th International Conference, pp. 261–268.

Perrollaz, M., Labayrade, R., Royere, C., Hautiere, N., & Aubert, D. (2006). Long range obstacle detection using laser scanner and stereovision. In Proceedings of the Intelligent Vehicles Symposium, pp. 182–187.

Peynot, T., Scheding, S., & Terho, S. (2010). The marulan data sets: Multi-sensor perception in a natural environment with challenging conditions. The International Journal of Robotics Research, 29(13), 1602–1607.

Premebida, C., & Nunes, U. (2005). Segmentation and geometric primitives extraction from 2d laser range data for mobile robot applications. In Robotica 2005—Scientific Meeting of the 5th National Robotics Festival.

Schulz, D., Burgard, W., Fox, D., & Cremers, A. (2001). Tracking multiple moving objects with a mobile robot. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).

Shoemaker, C., & Bornstein, J. (1998). The demo iii ugv program: A testbed for autonomous navigationresearch. In Proceedings of the IEEE ISIC/CIRA/ISAS Joint Conference.

Song, X., Cui, J., Wang, X., Zhao, H., & Zha, H. (2008). Tracking interacting targets with laser scanner via on-line supervised learning. In IEEE International Conference on Robotics and Automation (ICRA).

Spinello, L., Arras, K. O., Triebel, R., & Siegwart, R. (2010). A layered approach to people detection in 3d range data. In Proc. of the AAAI Conference on Artificial Intelligence: Physically Grounded Track.

Steinhauser, D., Ruepp, O., & Burschka, D. (2008). Motion segmentation and scene classification from 3d lidar data. In Proceedings of IEEE Intelligent Vehicles Symposium.

Sun, Z., Bebis, G., & Miller, R. (2006). On-road vehicle detection: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence, 28(5), 694–711.

Taylor, G., & Kleeman, L. (2004). A multiple hypothesis walking person tracker with switched dynamic model. In Proceedings of the Australasian Conference on Robotics and Automation (ACRA).

Thornton, S., Hoffelder, M., & Morris, D. (2008). Multi-sensor detection and tracking of humans for safe operations with unmanned ground vehicles. In Proceedings 1st. Workshop on Human Detection from Mobile Robot Platforms, IEEE ICRA. IEEE.

Thornton, S., & Patil, R. (2008). Robust detection of humans using multi-sensor features. In Proceedings of the 26th Army Science Conference.

Urmson, C., Anhalt, J., Bae, H., Bagnell, J., Baker, C., Bittner, R., Brown, T., Clark, M., Darms, M., Demitrish, D., Dolan, J., Duggins, D., Ferguson, D., Galatali, T., Geyer, C., Gittleman, M., Harbaugh, S., Hebert, M., Howard, T., Kolski, S., Likhachev, M., Litkouhi, B., Kelly, A., McNaughton, M., Miller, N., Nickolaou, J., Peterson, K., Pilnick, B., Rajkumar, R., Rybski, P., Sadekar, V., Salesky, B., Seo, Y.-W., Singh, S., Snider, J., Struble, J., Stentz, A., Taylor, M., Whittaker, W., Wolkowicki, Z., Zhang, W., & Ziglar, J. (2008). Autonomous driving in urban environments: Boss and the urban challenge. Journal of Field Robotics Special Issue on the 2007 DARPA Urban Challenge, Part I, 25(1), 425–466.

Wang, C.-C., Lo, T.-C., & Yang, S.-W. (2007). Interacting object tracking in crowded urban areas. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'07), Roma, Italy.

Xavier, J., Pacheco, M., Castro, D., Ruano, A., & Nunes, U. (2005). Fast line, arc/circle and leg detection from laser scan data in a player driver. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA).

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. IEEE Transactions on Neural Networks, 16(3), 645–678.

Yang, S.-W., Wang, C.-C., & Thorpe, C. (2011, August). The annotated laser data set for navigation in urban areas. The International Journal of Robotics Research, 30(9), 1095–1099.

Zhao, H., Chen, Y., Shao, X., Katabira, K., & Shibasaki, R. (2007). Monitoring a populated environment using single-row laser range scanners from a mobile platform. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).

Zhao, H., Shao, X. W., Katabira, K., & Shibasaki, R. (2006). Joint tracking and classification of moving objects at intersection using a single-row laser range scanner. In IEEE Intelligent Transportation Systems Conference, Toronto, Canada (pp. 287–294).

Zhao, H., & Shibasaki, R. (2005). A novel system for tracking pedestrians using multiple single-row laser-range scanners. IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans.

Ziebart, B., Ratliff, N., Gallagher, G., Mertz, C., Peterson, K., Bagnell, J., Hebert, M., Dey, A., & Srinivasa, S. (2009). Planning-based prediction for pedestrians. In International Conference on Intelligent Robots and Systems (IROS).