OTMS: A Novel Radar-Based Mapping System for Automotive Applications

Justin Bode* and Arslan Munir†
* Sierra Nevada Corporation
† Department of Computer Science, Kansas State University
Email: *justin.bode@gmail.com, and †amunir@ksu.edu

***Abstract*—Current automotive radar systems are designed to detect objects within an area around the vehicle and provide alerts or other driving assistance features to the driver. The main goal of these driving assistance systems is to increase safety of not only the driver but also increase safety for pedestrians on and around roadways. In this paper, we propose an on-board terrain mapping system (OTMS), which is an automotive radar-based system that extends the automotive safety goals to include mapping and vehicle-to-everything (V2X) communication. The proposed system utilizes a 94 GHz 3-dimensional (3D) pulsed radar to sense the environment around the vehicle and create a 3D model of the environment that can be sampled into a global database for mapping purposes. Objects are also detected and tracked within the model and can be used to provide assistive driving to the user even in adverse weather conditions. The model generated by the system can then be stored locally on the automobile or transmitted to other vehicles or infrastructure for use in mapping, navigation, and/or safety applications. The results validate that the OTMS is a valid system for providing safety and mapping functions for advanced driver assistance systems (ADAS).**

I. Introduction and Motivation

Automotive technology has made great improvements over the past decade in the fields of sensing and assistive driving. Automotive radars and cameras have been used for many years now to help with driving tasks such as backing up, lane keeping, and collision detection [1]. In recent years, automotive sensors have even been used in self-driving vehicles that require little to no user intervention [2]. The automotive driving assistance technologies are improving at a very rapid rate and much research, both public and private, is currently being conducted in this field.

One of the main focuses of the automotive research is to increase the safety of vehicles driving on roadways. To achieve this goal, the sensors contained on the vehicles are designed for the detection of objects within regions around the vehicle and not necessarily the detection of the entire scene including the terrain. It is mostly assumed that the terrain portion of the scene is going to be relatively flat roadway. The proposed on-board terrain mapping system (OTMS) has been designed to include detection of the entire scene, terrain and above terrain objects, using only a radar sensor. This novel design approach allows for many new features such as mapping and vehicle-to-everything (V2X) communication to be added to the automotive system to help with advanced driver assistance systems (ADAS).

Central to the OTMS design approach is the 3D imaging radar sensor, which offers unique advantages over current automotive radars. The 3D imaging radar sensor is not yet cost-effective because of economy of scale. We have used it to demonstrate the capabilities that the OTMS can provide given a sensor with similar characteristics. If the radar were to be produced in large quantities, it could become cost-effective in the future. The radar operates at 94 Ghz and is a pulsed beam with a 1 degree angular beamwidth. The radar is capable of scanning an area of 30°by 30°, has a range resolution of $\sim$ 0.74 feet with a max range of $\sim$ 1106 feet. These characteristics make the radar similar to other long range automotive radars. The radar scans the entire field of view in 0.5 seconds using a pencil beam to create truly 3D data. The operating frequency of the radar still allows it to penetrate obscurants like fog and rain. Other sensors, such as light detection and ranging sensors (LIDAR) and cameras, cannot penetrate these obscurants, which continue to be the leading causes of accidents on roadways [3]. The characteristics of this radar allow for the design of the OTMS to be accomplished. The OTMS design is unique in that it not only allows for the detection of objects that might pose a threat to the vehicle, but it also allows for the creation of accurate 3D models of the environment being driven in. Our main contributions in this paper are:

- Proposing OTMS that enhances the capabilities of ADAS by providing driver assistance, even in hazardous environmental conditions, such as fog, snow and sand storms.
- Developing OTMS as a cost-effective ADAS enhancer as its design is based on only one radar sensor as opposed to multiple sensors.
- Designing OTMS that not only enhances the safety features of automobiles but is also capable of mapping the external environment using only a radar sensor as opposed to a camera or LIDAR sensor for operation in all weather conditions.
- OTMS development using a 94 GHz long range radar sensor with high angular resolution to improve detection and tracking of objects in the scene at further ranges.
- Developing efficient mapping techniques and data structures for storing the mapped environment by OTMS in a local database that can then be accessed in future to recall information about the roadway (terrain) and/or can be easily sent to other vehicles to improve their situational awareness.

The goal of these contributions is to expand the field of research in automotive on-board sensors and ADAS to provide features beyond that of the local safety of the vehicle. Our contributions will likely help utilize the sensing capabilities already present on the vehicle to their fullest potential. The rest of this paper is organized as follows. Related work is presented in Section 2. The OTMS will be described in detail in Section 3 followed by a description of some key algorithms enabling OTMS in Section 4. Test results will be provided in Section 5. Finally, Section 6 provides concluding remarks.

II. Related Work

Radars, LIDARs, and cameras have been used in automotive safety applications for many years now. A popular method for determining which areas around the vehicle are safe for driving is to create occupancy maps. Many different sensors have been used to build occupancy maps including radar, LIDAR, and sonar [4] [5]. The OTMS is different in that the goal is not to create an occupancy grid, but rather to create a 3D model of the entire scene, both ground and above ground elements. This virtual scene can then be used to create maps of the region, be displayed to the user, or even sampled into an occupancy map as an input for collision warning (CW) or collision avoidance applications (CA).

There exists some previous work that uses radar sensors to create 2D maps with the goal of detecting the road boundaries [6][7]. These techniques rely on the intensity information received from radars to detect the edges of roads. These techniques could still be applied to the intensity information received from this radar and the intensity map created by the OTMS, and the design of the OTMS does not preclude this from being accomplished. However, discussing the details of road edge detection using OTMS is beyond the scope of this paper.

The operation of the self-driving Google car reveals that the automatic driving system is reliant on the creation of high-precision, accurate maps together with current sensor data in order to drive the car automatically [2] . While the OTMS is not proposed to become a substitute for those maps, it does provide a way to create maps in real-time and to share that information with other vehicles.

Tesla Motors, Inc. has recently started producing vehicles with an autopilot mode. A recent fatality while in autopilot mode is a reminder that continued research into this field is necessary [8]. The Tesla autopilot feature currently uses camera and radar sensors to provide the necessary information. Tesla has also announced that it plans to switch to using only radar sensors in the future [9]. Hence, this future research thrust of Tesla autopilot mode is similar to OTMS which is designed to utilize a radar sensor to provide the necessary information for ADAS, such as autopilot.

The fusion of current sensor data together with *a priori* data is another possible field of study that the OTMS design can easily be modified to support. In [10], two methods of fusion, low-level and high-level, are proposed and simulated. Low-level fusion involves fusing data that has had little to no processing performed on it, and high-level fusion involves fusing data that has been processed by one or more functions. This type of fusion is not described in this paper, but it can be easily seen how the design of the OTMS can support a high-level fusion from multiple sensors.

III. OTMS—On-board Terrain Mapping System

The OTMS is a system designed to take raw sensor data as input and process it into data that can be used for ADAS applications. The ADAS applications focused on in this paper are obstacle detection and mapping. An overview of the OTMS is shown in Fig. 1.

The OTMS comprises of six major modules: (i) signal processing, (ii) object detection and mapping, (iii) database, (iv) hazard detection, (v) driver advisories generation, and (vi) V2X communication. The signal processing module is the

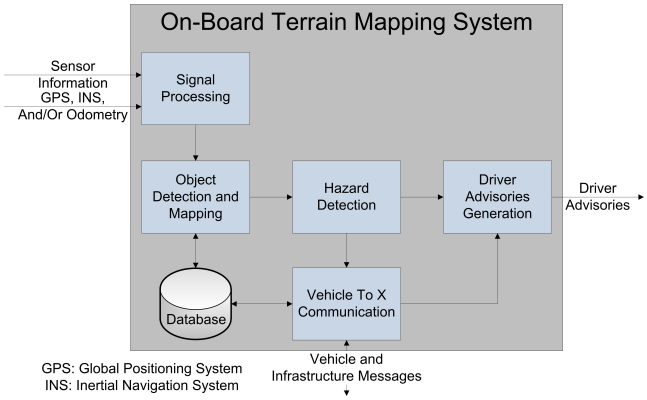

Fig. 1. OTMS top-level design. GPS: Global Positioning System, INS: Inertial Navigation Sensor

interface to the sensors. The module receives the data produced by the sensors and performs low-level signal processing functions, such as noise reduction and motion compensation, on the received sensor data. The output data from the signal processing module is then fed to the object detection and mapping module, which performs radar point classification, object detection, and mapping. The mapping data is then saved in the database module, which is a local database of the 3D model created from the processed sensor data.

The object detection and mapping module also passes the detected objects to the hazard detection module where the objects are checked to determine if they pose a threat to the vehicle. The hazard detection module coordinates with the driver advisories generation module to create driver advisories. The hazard detection module also coordinates with the V2X communication module to send and/or receive driver advisories from other vehicles and/or infrastructure. The V2X communication module also has access to the local database to either send information to other vehicles and/or infrastructure from the database or receive the information that can be added to the database. These modules are not described in detail in this paper, but are reserved for future work.

This paper focuses on the discussion of first two modules: signal processing and object detection and mapping. The detailed discussion of the other four modules is reserved for future work. We discuss the first two modules in the following sections.

A. Signal Processing

An activity diagram of the signal processing module in Enterprise Architect [11] is shown in Fig. 2. This module is responsible for the low-level processing of the radar sensor data, mainly noise reduction and motion compensation. The working of the signal processing module is described below.

As each beam position data of the radar is acquired by the signal processing module, the beam position data is matched with its corresponding navigation data. Each set of data is timestamped using GPS time to ensure that the navigation data matches best with the radar data. This navigation and radar data is then paired together for future processing.

The radar beam position data is then processed further to extract the signal from the returns and reject the radar noise. The probability of false alarm (Pfa) for each range bin in the radar sensor is 10^{-3}. Given that there are 1500 range bins for each beam position, approximately 1-2 false alarms are expected per beam position. The signal extraction function

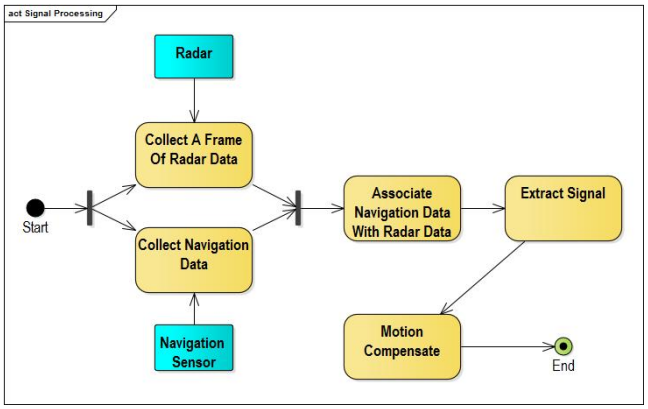

Fig. 2. Activity diagram of the signal processing module

runs a proprietary algorithm to reduce the Pfa down to 10^{-10}. While this reduced Pfa does not guarantee that no false alarms are passed as signal, the probability of false alarm occurring is low enough for ADAS applications, yet the probability of detect (Pda) is still high enough to reliably detect objects.

After the signal has been extracted, each remaining return within the beam position is motion compensated into a level Cartesian coordinate frame using the navigation data paired with the radar beam and calibration information determined beforehand. The origin of the coordinate frame is located at the radar's geographic position at the beginning of the collection cycle. The geographic position is kept with the frame of points so that it can be easily transformed into a geodetic coordinate system such as latitude/longitude/altitude (LLA) during the mapping function. The positive X axis is along the centerline of the vehicle, the positive Y axis is to the right of the vehicle, and the positive Z axis is pointed down aligned with gravity.

After the motion compensation step has been completed for each radar beam position in a frame, the motion compensated points are passed to the object detection module for further processing.

B. Object Detection and Mapping

This section will describe the second module of the OTMS which includes both object detection and mapping.

1) Object Detection: The object detection module of the OTMS is responsible for detection of objects within the collection of radar returns generated by the signal processing module. Each detected object can be tracked and analyzed for safety critical applications. An activity diagram of this module in Enterprise Architect is shown in Fig. 3.

The first step in the object detection phase is to transform the magnitude data received from the radar into intensity data that is used later to validate objects detected in the scene. This is accomplished by scaling the magnitude data with a log function to reduce the dynamic range of the data. A log scaling preserves the dynamic range of low magnitude returns while scaling down the dynamic range of high magnitude returns to provide a better range of data to operate on. The points are then sampled into a 2D image similar to a radar B-scope image. A B-scope image provides a 2-D top-down representation of space, with the vertical axis representing range and the horizontal axis representing azimuth. The B-scope image could then be visualized or used for other purposes such as detection of road boundaries [6] [7].

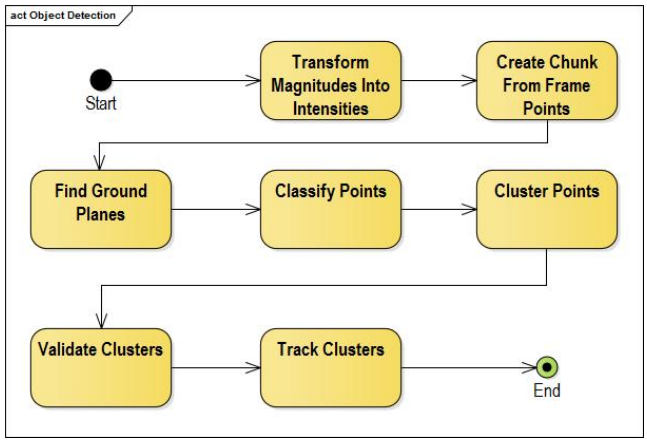

Fig. 3. Activity diagram of object detection module

The process of object detection begins by dividing the points into smaller sections of data called chunks. The chunks are created by projecting all the points into the XY plane and then dividing the plane into small regions. The regions in the X direction (azimuth) are separated by the angle from the centerline and the regions in the Y direction (range) are separated by their range from the sensor. There are five (5) regions in azimuth that span evenly across the field of view and fifteen (15) regions in range that span 25 feet at close ranges and 100-200 feet at the farther ranges.

For the points in each chunk, a plane is created that best fits the data using singular value decomposition (SVD). It is assumed that the majority of the points contained within each chunk are returns from the terrain. Thus, the fitted plane can be used to classify the points within each chunk. A standard deviation value representing the uncertainty of the radar beam width is applied above and below the fitted plane. Points that fall within three standard deviations of the plane are classified as terrain. Three standard deviations was selected because it represents 99.7% of the area of the radar beam. The points more than three standard deviations above the plane are classified as obstacles and the points more than three standard deviations below the plane are classified as below terrain. It should be noted that for radar applications, it is not unusual to see points located beneath a surface due to multi-path effects of the radar beam. These points are usually mirrored returns from strong targets. Multi-path radar effects have been studied for over 50 years and are still being studied today [12].

After point classification, the obstacle points are clustered using DBSCAN [13]. The resulting clusters are validated by comparing them with their surrounding regions in the 2D intensity image generated earlier by this module. Validation occurs by verifying a contrast in intensity of the object itself versus the background of the object. If there is a significant contrast, meaning that the cluster has a higher intensity than its surrounding area, the cluster is validated. Validated clusters are then tracked with time and can be used to determine if the object is a hazard to the vehicle.

2) Mapping: It is not feasible to store all of the points in memory over periods of time longer than 10 seconds. The radar can theoretically generate 5,581,500 points in each 0.5 second frame. Thus, in order to maintain a history of the environment for future use, the data must be sampled into a database capable of storing data over a large physical area

representing the area in the physical world that the vehicle will be traveling in.

A quad-tree is a common data structure used to spatially sort data for quick searching. This construct can also be used to stored data at different resolutions if the levels of the quad-tree are conceived as resolutions of a database. The root of the quad-tree designates the lowest resolution that can be stored and the leaves of the quad-tree signify the highest resolutions that can be stored.

Furthermore, the radar sensor used in the OTMS, like most sensors, has an angular field of view. This means that points near the sensor are higher resolution than the points farther from the sensor in a Cartesian coordinate system. This makes the quad-tree even more suited for sampling sensor data by allowing areas closer to the sensor be sampled into the database at higher resolutions (lower level in the quad-tree) and areas farther from the sensor be sampled at lower resolutions (higher level in the quad-tree).

It is also desirable that the model of the scene, as represented by the quad-tree, be able to be stored outside of RAM on a large scale storage device such as a SSD. This allows areas that are far enough away from the vehicle to be moved to the storage device to free up memory for the areas near the vehicle that are actively being sensed. When those areas that were written to the storage device are revisited, either during the same trip or on a different trip, the data can be recalled back into RAM for further processing. An example of a quad-tree being used in this manner is given in [14].

The points classified as terrain by the object detection module of the OTMS can then be sampled into the quad-tree database at the correct resolution. The data item stored at each level in the quad-tree is a 2D array of locations with each element in the array representing the height of the terrain measured in that area. For example, a 2D array with 32 elements in each direction will span a region in the physical world that is directly related to the level or resolution of the quad-tree. When a radar point classified as terrain is located within that area in the physical world, its height is averaged into the existing height value at that location within the 2D array. The result is a digital elevation map (DEM) of the physical world represented by the quad-tree. By averaging all the returns located within the region of the 2D array element, a better estimate of the true height of the world is obtained.

IV. Algorithms For OTMS

One of the unique algorithms developed for the OTMS is the validation of clusters after the classification and clustering functions. This allows the OTMS to reject above ground clusters that are not true objects in the scene. The underlying principle of the algorithm is that when objects are imaged that are not part of the terrain, their returns are generally larger in magnitude than their surrounding area. By using this principle, above ground clusters can be compared to their immediate surroundings. The algorithm is defined in Algorithm 1.

The algorithm's inputs are the clusters and the intensity image created at the beginning of the object detection module, section III-B. An integral image [15] of the intensity image is created (line 1) to increase performance by allowing for constant time summations of regions in the intensity image. Then, for each cluster detected, a tight-fitting bounding box around the location of the cluster in the intensity image is determined (line 3). All the image pixels contained within this bounding box are then averaged together to obtain the average cluster intensity value (line 4). The bounding box is then padded on all sides (line 5) and the image pixels are averaged again to obtain the average background intensity (line 6). If the difference between the average cluster intensity and the average background intensity is above a threshold (lines 7-8), then the cluster is validated (line 9). If the difference is below a threshold, then the cluster is invalidated and the points are re-classified as terrain (lines 11-12). The threshold is empirically determined for this work but can be determined better by running a training algorithm on a large set of data that has been classified beforehand as either being valid or invalid. The algorithm has some similarity with constant false alarm rate (CFAR) algorithms. The results of the validation step for a frame of data are shown in Section 5.

Input: Clusters, Image, Threshold
Output: Validated Clusters

```
1  Create Integral Image, I, from Image;
2  foreach Cluster in Clusters do
3  |   Create bounding box, B;
4  |   Compute average intensity, I_cluster, inside of B;
5  |   Pad B to obtain B_padded;
6  |   Compute average intensity, I_background, inside of
   |   B_padded;
7  |   Compute diff = I_cluster − I_background;
8  |   if diff > Threshold then
9  |   |   Mark cluster valid
10 |   else
11 |   |   Mark cluster invalid
12 |   |   Re-classify cluster points as terrain
13 |   end
14 end
```

Algorithm 1: Cluster validation algorithm

V. Results

A. Experimental/Test Setup

To test the OTMS design, the 3D imaging radar is mounted on a van and driven to a point overlooking a flat area. Four metal poles of various heights are lined up in a row along the center axis of the radar. These poles are located at distances of 100, 200, 300, and 400 feet from the radar. Other poles are located to the sides of this center line and a vehicle is also in the scene. During the test event, the vehicle drives around the poles on the center line. A snapshot of the test setup from the infrared (IR) camera is shown in Fig. 4. Radar and navigation data is collected during the test event and stored in a binary file.

We implement the signal processing and object detection modules in a multi-threaded C program for windows. Threads are created to perform the following functions:

- Receiving radar data from the binary file.
- Receiving navigation data from the binary file.
- Performing the computations involved in the signal processing module on the raw radar data.
- Executing the computations involved in the object detection and mapping module on the processed radar data.

The C program ran on a windows PC with an Intel Core i5-4690K processor with four (4) cores and running at 3.5 GHz. A separate program also ran that played back the timestamped

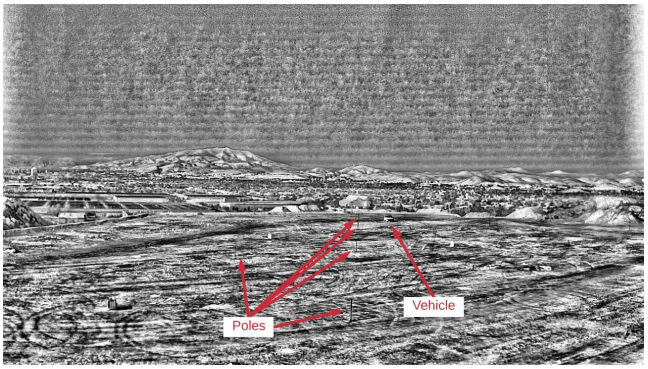

Fig. 4. Snapshot From IR Camera Showing the Test Setup

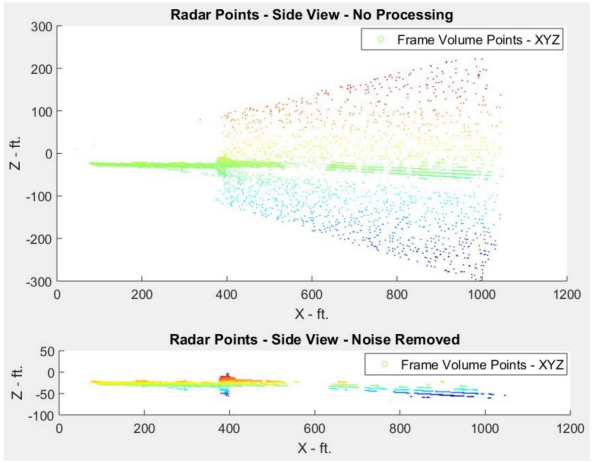

Fig. 5. Test results from the signal processing module on a single frame of radar data. Left: motion compensated raw radar returns including noise. Right: motion compensated raw radar returns after noise removal.

binary file collected during the test event to simulate the program receiving data from the radar in real time. Data is dumped at various stages of the processing pipeline and timing measurements are taken for computations involved in each OTMS module.

B. Signal Processing

Fig. 5 shows the results from the signal processing module on a single frame of radar data. The image on the left contains the motion compensated raw radar returns as they are received from the radar. The signal coming out of the radar contains noise in the entire field of view, except for the near ranges due to a sensitivity/time control (STC) curve applied to the returns. There are also effects from multi-path seen below the terrain that are a result of the corner reflector located on top of the last pole in the scene. This is because the corner reflector represents a very large target to the radar by reflecting almost all of the energy it receives back to the radar while other objects will only reflect a portion of the energy they receive. Thus, when the radar is pointed beneath the corner reflector, energy can bounce off the ground then to the corner reflector and then return back to the radar along the same trajectory. The radar then senses valid returns at the same range as the corner reflector but the returns are located beneath the ground because of the pointing angle. The terrain and the other objects can be seen within the returns and show that the radar is capable of detecting small objects (the poles) and large objects (the corner reflector and vehicle) at ranges exceeding 400 feet.

The image on the right contains the same frame of data that has been processed to remove the noise from the radar returns. The image shows that the noise is removed regardless of its location within the frame. This includes the noise located in the air above the terrain and noise located beneath the ground under the terrain returns. Also, most of the multi-path effects from the corner reflector located under the terrain have been removed. Noise removal is very important to the object detection and mapping functions because of the errors that can be introduced into the data. It should also be noted that the noise removal function did not remove the radar returns from the poles or the vehicle. This shows the robustness of the algorithm in the detection of objects.

C. Object Detection

The same frame of data that is fed to the signal processing module is then sent to the object detection module where it is further processed to classify the radar points, detect objects that are above the terrain and track the detected object. Fig. 6 depicts the test results from the object detection and mapping module. The left side of the figure shows the results after the classification step has been performed. These images show the resulting data after it has been separated into regions, had a ground plane estimated for each region, and each radar point has been classified. The side view shows that the returns from the terrain have been correctly classified while still being able to detect the objects that are above the terrain. Returns determined to be below the terrain are also classified and are essentially thrown away after this step. The top-down view shows the detections on the poles and the vehicle. There are some falsely classified points visible in this view.

Clustering the points classified as above the ground occurs next. A separate image of the individual clusters is not shown, however, each pole was put into its own cluster as well as the vehicle. The clusters are then validated to further reduce the Pfa. The right side of Fig. 6 shows the results of the validation step. It can be seen that all the true objects remain and that some of the false alarms have been invalidated. Note that there are some falsely classified clusters at the near range of the returns. These could be further processed to check for validity.

D. Processing Time Analysis

The processing times for the signal processing and object detection and mapping modules are shown in Table I. The times are obtained on the Windows PC running the C implementation of the modules to process over 10,000 frames of radar data. These execution times are then scaled to estimate the run time on a SABRE SD Freescale infotainment automotive board running at a nominal frequency of 396 MHz. It should be noted that the automotive processor on the SABRE board can be run up to a maximum of 1 GHz if needed at the expense of additional energy consumption.

These results show that the proposed OTMS design and implementation is feasible to run on contemporary automotive technology. The scaled average runtime of 238.37 milliseconds for the object detection computations is approximately half of the required runtime rate of 500 milliseconds which corresponds to the frame rate of the radar. There is also a very low standard deviation on the run times which indicates that the maximum processing time is not very common and that the processor can catch up the processing requirements

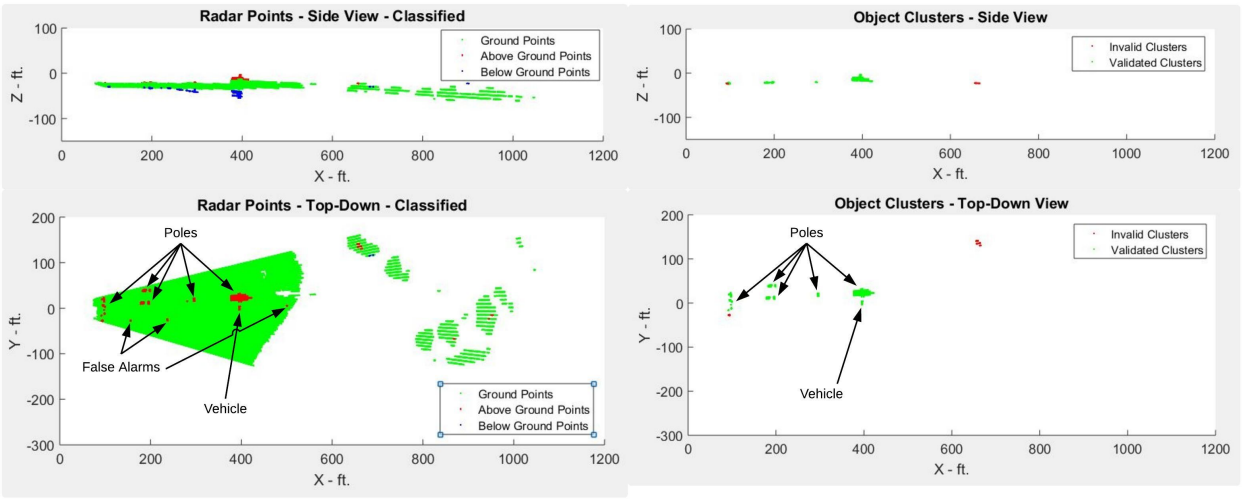

Fig. 6. Test results from the object detection module on a single frame of radar data. Top left: side view of classified radar returns. Bottom left: top-down view of classified radar returns. Top right: side view of detected objects. Bottom right: top-down view of detected objects.

TABLE I
PROCESSOR RUN TIMES, IN MILLISECONDS, OF OTMS MODULES

Module	Function	Intel Core i5 @ 3.5 GHz			Scaled to 0.396 GHz			
		Avg	Min	Max	Avg	Min	Max	Std Dev
Signal Processing	Associate Nav Data	0.16	0	61	1.41	0	536.14	2
	Extract Signal	0.000566	0	4.45	5	0	39.33	0.00189
	Motion Compensate	0.000546	0	6.13	4.83	0	54.18	0.00228
	Total	**0.1635**	**0**	**61**	**1.45**	**0**	**536.14**	**2**
Object Detection	Transform Magnitudes	0.772	0.662	7.26	6.82	5.85	64.17	0.193
	Create Chunks	2.92	2.67	10.99	25.81	23.60	97.13	0.377
	Find Ground Plane	1.70	1.50	12.19	15.03	13.26	107.74	0.344
	Classify Points	0.362	0.295	7.43	3.20	2.61	65.67	0.153
	Cluster Points	11.7	6.07	30.87	103.41	53.65	272.84	2.57
	Validate Clusters	3.14	2.88	10.32	27.75	25.45	91.21	0.446
	Track Clusters	0.005	0	0.098	0.044	0	0.87	0.0027
	Total	**26.97**	**19.51**	**69.34**	**238.37**	**172.44**	**612.85**	**3.38**

even if the processing of some frames takes more than the average computation time.

VI. CONCLUSIONS

This paper proposes OTMS and describes the functionality of signal processing and object detection and mapping modules within OTMS. We also demonstrate a proof of concept for the OTMS in this paper. The system can be used as a baseline for developing similar systems or could be developed further into a system capable of being installed in automotive systems. Benefits of this design include (i) the ability to operate in hazardous weather conditions, (ii) the ability to detect objects using a single radar sensor, and (iii) the ability to map the environment into a scalable database capable of being stored on the vehicle. Testing results show that the OTMS can meet real-time requirements of ADAS applications. The proposed OTMS not only enhances ADAS but can also be leveraged for other tasks, such as, mapping and navigation.

REFERENCES

[1] H. L. Bloecher, J. Dickmann, and M. Andres, "Automotive active safety and comfort functions using radar," in *IEEE Intl. Conf. on Ultra-Wideband*, Sept 2009, pp. 490–494.

[2] M. Birdsall, "Google and ITE: The road ahead for self-driving cars," *Institute of Transp. Engineers J.*, vol. 84, no. 5, pp. 36–39, May 2014.

[3] W. S. Ashley, S. Strader, D. C. Dziubla, and A. Haberlie, "Driving blind: Weather-related vision hazards and fatal motor vehicle crashes," *Bulletin of American Meteorological Society*, vol. 96, no. 5, pp. 755–778, 2015.

[4] Y.-K. Choi and S.-J. Lee, "Development of advanced sonar sensor model using data reliability and map evaluation method for grid map building," *Journal of Mechanical Science and Technology*, vol. 29, no. 2, pp. 485–491, 2015.

[5] F. Homm, N. Kaempchen, J. Ota, and D. Burschka, "Efficient occupancy grid computation on the gpu with lidar and radar for road boundary detection," in *IEEE Intelligent Vehicles Symposium*, June 2010, pp. 1006–1013.

[6] C. Lundquist, L. Hammarstrand, and F. Gustafsson, "Road intensity based mapping using radar measurements with a probability hypothesis density filter," *IEEE Trans. on Signal Processing*, vol. 59, no. 4, pp. 1397–1408, 2011.

[7] S. H. Tsang, P. S. Hall, E. G. Hoare, and N. J. Clarke, "Advance path measurement for automotive radar applications," *IEEE Trans. on Intelligent Transportation Systems*, vol. 7, no. 3, pp. 273–281, 2006.

[8] M. Harris, "Tesla autopilot crash exposes industry divide," http://spectrum.ieee.org/cars-that-think/transportation/self-driving/what-next-for-teslas-autopilot, accessed: 2016-10-04.

[9] D. Z. Morris, "Tesla shifting autopilot from camera to radar-based sensing," http://fortune.com/2016/09/11/tesla-autopilot-shift/, accessed: 2016-10-10.

[10] T. Herpel, C. Lauer, R. German, and J. Salzberger, "Multi-sensor data fusion in automotive applications," in *3rd Intl. Conf. on Sensing Technology (ICST 2008)*, Nov 2008, pp. 206–211.

[11] S. Systems, "Enterprise architect," Oct 2016. [Online]. Available: http://www.sparxsystems.com/products/ea

[12] K. Haspert and M. Tuley, "Comparison of predicted and measured multipath impulse responses," *IEEE Trans. on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 1696–1709, 2011.

[13] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise." in *Intl. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, 1996, pp. 226–231.

[14] W. E. Brandstetter, "Multi-resolution deformation in out-of-core terrain rendering," p. 58, 2007.

[15] F. C. Crow, "Summed-area tables for texture mapping," in *Annual Conf. on Computer Graphics and Interactive Techniques*, 1984, pp. 207–212.