Challenges in Perception and Decision Making for Intelligent Automotive Vehicles: A Case Study

Bunyo Okumura, Michael R. James, Yusuke Kanzawa, Matthew Derry, Katsuhiro Sakai, Tomoki Nishi, and Danil Prokhorov

***Abstract*—This paper overviews challenges in perception and decision making for intelligent, or highly automated, automotive vehicles. We illustrate our development of a complete perception and decision making system which addresses various challenges and propose an action planning method for highly automated vehicles which can merge into a roundabout. We use learning from demonstration to construct a classifier for high-level decision making, and develop a novel set of formulations that is suited to this challenging situation: multiple agents in a highly dynamic environment with interdependencies between agents, partial observability, and a limited amount of training data. Having limited amount of labeled training data is highly constraining, but a very real issue in real-world applications. We believe that our formulations are also well suited to other automated driving scenarios.**

***Index Terms*—Autonomous driving, classifier, finite state machine, high-definition lidar, high-fidelity map, learning from demonstration, robot, roundabout, state representation.**

I. Introduction

ADVANCES in sensing technology and ever increasing computational resources are enabling rapid improvement in driving automation and self-driving vehicle technology. Such intelligent vehicles are being challenged to operate in increasingly complex environments such as urban streets [1], [2]. What might have been seen as a futuristic experience at the time of E. Dickmanns and his 1994 Mercedes autonomous driving [3] appears much more ordinary today, and will likely be commonplace tomorrow. Former DARPA Robotics Challenge Manager G. Pratt notes that "the base technologies on which robots depend–particularly computing, data storage, and communications–have been improving at exponential rates, and [that] recent advances like Cloud Robotics and Deep Learning could leverage these base technologies in a virtuous cycle of explosive growth of all kinds of robots including robotic vehicles" [4].

While computational capacity is becoming less of an issue, the remaining research challenges are in developing perception and decision making algorithms [2], [5] with sufficient performance and reliability in the wide range of real-world environments encountered by an automated vehicle. Recently, advances from the machine learning (ML) community have taken hold in the Robotics community: a trend which has continued into automated driving research. ML methods leverage the availability of large datasets and large amounts of available computation, to provide solutions that outperform traditional approaches while providing a path forward toward developing algorithms for perception and decision making in complex environments.

Research and development of intelligent automotive vehicles is underway at many corporate and educational institutions. We list just a few of the most prominent players, without reflecting a precedence, a priority, or an exclusivity: 1) Google with their fleet of autonomous cars roaming the streets of US in multiple states [1], 2) Daimler with their Mercedes S500 Distronic Plus system [6] and [2], 3) BMW with their highly automated driving on highways [7], 4) Volvo with their 100-car automated driving test fleet deployed in Gothenburg, Sweden [8], 5) Mobileye with their driver assistance solutions deployed in many production vehicles across different OEMs [9], 6) VisLab with their recent experiments in autonomous cars crossing many countries (Intercontinental Challenge, as in [10]), as well as their urban test drives [11], 7) Stanford University autonomous driving team with their recent research advances in [12], [13], 8) CMU-GM collaborative research lab [14] and their research platform [15]. Various public-private partnership and international projects have also been undertaken, e.g., CityMobil [16], DIPLECS [17], AdaptIVe [18]. While there are many differences in these projects, a common thread of research is focused on perception with multiple sensors, underlining the importance of environment understanding for highly automated vehicles.

A. Background: Sensors and Sensor Processing

The typical sensor suite in an automated vehicle includes perceptual sensors such as cameras, low- and high-definition lidar, and radar; positioning and orientation sensors such as GPS and IMU; and vehicle-measurement sensors such as wheel and steering wheel encoders, and throttle position and other control feedback sensors. While it may be argued that all of these play some role in perception (and more indirectly in decision making), here we focus on those sensors that are most directly linked to the perception problem.[1] Radar is a robust and invaluable information source for perceptual tasks, however the

Manuscript received November 01, 2015; revised February 04, 2016; accepted March 13, 2016. Date of publication April 17, 2016; date of current version July 18, 2016. *(Corresponding author: Danil Prokhorov.)*

B. Okumura is with the Toyota Motor Corporation, Toyota, Aichi 471-8571, Japan (e-mail: bunyo_okumura_aa@mail.toyota.co.jp).

M. R. James, Y. Kanzawa, M. Derry, K. Sakai, T. Nishi, and D. Prokhorov are with the Toyota Research Institute of North America, Ann Arbor, MI 48105, USA (e-mail: michael.james@tema.toyota.com; yusuke.kanzawa@tema.toyota.com; matthew.derry@tema.toyota.com; katsuhiro.sakai@tema.toyota.com; tomoki.nishi@tema.toyota.com; dvprokhorov@gmail.com).

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

Digital Object Identifier 10.1109/TIV.2016.2551545

[1]We do not discuss infrastructure/cloud support in this paper, although the assumption of always available, up-to-date, high-fidelity maps is widespread.

spatial resolution of radar is typically poor compared to camera and lidar. Thus, much recent perception research revolves around cameras and lidar.

With respect to research on camera-based perception, in addition to the work done by the aforementioned groups, we wish to reference a few other promising research efforts that are extending the frontier. These are directed at creating a tunable visual attention system [19], recognizing various objects such as traffic signs [20], traffic lights [21], pedestrians [22], vehicles and lanes [23], and even whole intersection layouts [24]. Of course, the topic of self-driving cars is rapidly growing in popularity and generating many publications annually, so it is impossible to provide a comprehensive overview of all work in this area.

While camera systems provide high-resolution 2-D images, high-definition lidars typically give lower-resolution, but contain highly accurate distance/depth data. Thus, they have been quite popular for automated vehicle perception systems that provide the three-dimensional (3-D) data necessary for high-quality decision making algorithms. Some well known but now dated publications are worthwhile to get a glimpse into some inner workings of the lidar-equipped smart cars at the DARPA Urban Challenge 2007 (see, e.g., [25]) and others (see, e.g., [26], [13]).

Some teams from the 2007 Urban Challenge have continued to publish research on lidar perception. The Stanford team, exemplified by research on lidar calibration [27] and sophisticated tracking algorithms [28], is a good example. Their lidar calibration performs intrinsic and extrinsic calibration in arbitrary unstructured scenes through large-scale optimization. The lidar tracking algorithm addresses the problem of aligning point clouds of dynamic obstacles obtained from a wide variety of viewpoints (making traditional algorithms such as ICP difficult to apply). It frames the problem of pose alignment in a hierarchical probabilistic framework over pose transforms, working from rough to finer grained transform estimates. This anytime algorithm demonstrated state-of-the-art pose alignment (and therefore tracking and object reconstruction) performance.

The KIT team [29] has continued to explore lidar perception algorithms, starting from low-level 3-D point segmentation [30], a critical step in generating high-quality lidar perception results. Further work [31] frames the localization problem jointly with obstacle detection and tracking, directly addressing the interdependence between ego-motion estimation and obstacle tracking. The method works by creating sets of hypotheses, one for each dynamic or static object. Objects determined to be static are used in the localization estimation, while dynamic objects are tracked.

Moreover, the success of Google's self-driving car program has captured people's imagination, and appears to strongly rely on high-definition lidar as the primary sensor; see, e.g., [32], [33]. However, there are relatively few publicly available details about the underlying algorithms, especially those which couple perception with decision making.

B. Full System Development in Complex Scenarios

In many of the above papers, it is a specific part of the vehicle perception problem which is addressed, rather than both perception and decision making. For highly automated vehicles which are expected to make automatic decisions it is important to provide a solution for the perception and decision making system as a whole, e.g., as in the recent paper about the historic test of the Mercedes Benz S-Class prototype vehicle "Bertha" [2] or in the paper overviewing the results of BMW's highly automated driving experience on German highways [7].

Similarly, our case study presents an in-depth examination of the perception and decision making system for a challenging scenario: negotiating a roundabout. We believe that the current work will be useful to those who study practical perception and decision making systems. Specifically, we present an approach for developing a system addressing combined challenges in perception and decision making for intelligent automotive vehicles.

Roundabouts are a challenging scenario for road driving, due to tight time constraints on vehicle behavior for yield and merge maneuvers, high-quality sensing requirements for estimating the state of other vehicles, and multi-factor decision making based on these state estimates. It is especially difficult to predict the behavior of other vehicles in a roundabout due to the existence of multiple discrete decision points (exits) combined with continuous vehicle dynamics. Thus, we believe that roundabouts are especially well-suited for studying the full system development problem.

C. Related Roundabout Research

There are only a few examples of prior work specifically about roundabout planning. In [34] and [35] a planning method is introduced for autonomous driving maneuvers for roundabouts. Additionally, in [36] a finite state machine (FSM) is used in motion planning for environments such as left and right turns. In this work, a neural network is used to map observations to actions. The roundabout is then handled as a combination of turns. In both cases, these methods focus on path and speed planning for the roundabout, but neglect the interactions with and among other vehicles.

M. Muffert *et al.* [37] proposed both an image-based vehicle tracking method on a roundabout and an algorithm that provides merge decisions by computing time-to-contact and applying a threshold. This is enhanced with a method for predicting if a vehicle is leaving the roundabout based on estimated yaw rate. The method is demonstrated to be capable of human-like decisions for a given roundabout scenario, but little information is provided concerning failure rates, causes of failures, generalizability, etc. In [2], optimization-based planning is used for behavior generation. Yield or merge behaviors are generated as a result of constraints from both traffic rules and surrounding vehicles. The maneuver at the roundabout is achieved as a combination of a merge behavior and a subsequent right turn. It is mentioned that the robot will "generally wait for clear gaps between preceding vehicles before entering a roundabout," however a comprehensive analysis of the roundabout performance is missing.

II. Case Study Overview

Our approach addresses the main challenges of negotiating roundabouts by highly automated vehicles, viz., using sensor suites to detect, track, and make predictions about surround-

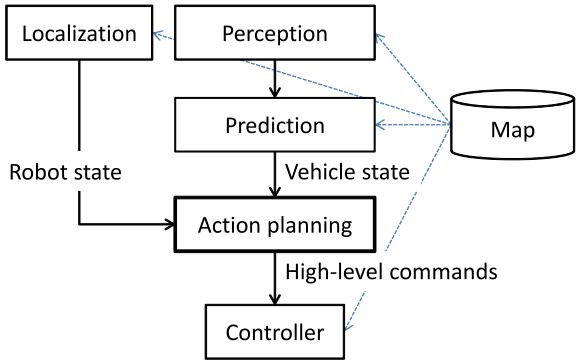

Fig. 1. Generalized structure of autonomous driving system. Localization provides information about the robot state such as position in our map and velocity. The perception module provides detection and tracking information about surrounding vehicles based on sensor inputs. The prediction module computes predictions about the motion of other vehicles, and the action planning module computes high-level commands. Finally, the controller generates low-level motor controls to actuate the robot.

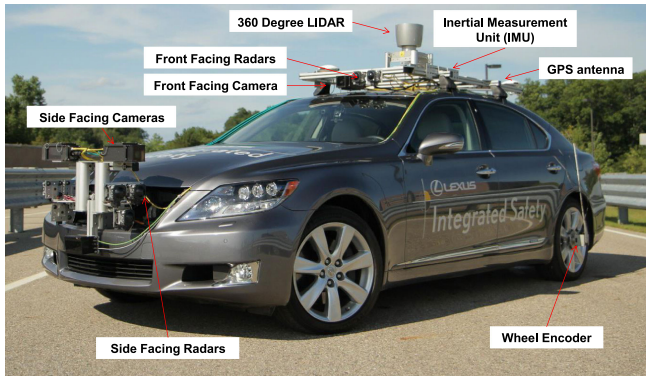

Fig. 2. Our test vehicle showing the hardware and sensor configuration. Details are in the text.

ing vehicles, then using that information for safe and effective decision making. These same challenges occur to some degree in every autonomous driving situation, and indeed are often considerably more challenging in the roundabout situation due to complex patterns of interaction and multiple decision points for each vehicle. Therefore, we believe that lessons learned during this case study can be useful in general.

We give an overview of the sensor suite, the core algorithms, and the overall system in Section III. We introduce a high-level action planning method in Section IV. In our system, *action planning* is the part of the highly automated driving system that takes perception results, and outputs high-level commands. These inform a low-level controller that generates motor controls. Fig. 1 shows our system structure.

Our action planning architecture consists of two levels: 1) an instantaneous *policy* based on a learned classifier, mapping from the current perception and robot state to an *action* (Section V); and 2) a FSM that ties together the sequence of actions along with environment information to generate the high-level command. The bulk of our work is on learning the classifier. We take a learning from demonstration approach [38], [39], due to numerous successes generating policies for safe and natural behavior.

One characteristic of the roundabout problem is that it is difficult to obtain a large amount of high-quality training data, making the learning problem more challenging. To address this, we have re-formulated the learning problem by decomposing the task (see Section V-A). We develop a general formulation of this decomposition that we believe is applicable to a wide range of automated driving tasks. This formulation not only decomposes the task, but introduces and motivates the use of auxiliary variables, in the form of predictions, to account for information lost both in the decomposition process and in hidden state.

Section VI presents experiments and analysis on a range of tests using real-world data gathered by our instrumented test vehicle. The results demonstrate that our decomposition incorporating predictions results in higher performance classification, which we verify with full-system testing.

The key elements that this case study demonstrates are:

1) An action planning pipeline and the details of making it work in conjunction with perception and control modules for real-world driving.
2) Formulation of a (ML) policy decomposition method, suited to complex multi-object domains where labeled training data is difficult to obtain.
3) A prediction method for vehicles to augment perception algorithms.
4) An extensive evaluation of our policy generation methods using real-world experiments.

III. Sensor Processing and Perception

As discussed in Section I, sensor selection has a large impact on the potential performance of perceptual algorithms, and therefore on the overall capabilities of the vehicle, hereafter called the *robot* to distinguish its autonomous capability. Surrounding vehicles are simply called *vehicles*.

Our robot shown in Fig. 2 is equipped with: an Applanix POS LV 220 (fused GPS + IMU positioning), 360 degree 64 beam high-definition lidar (Velodyne HDL64e), millimeter-wave radars, and multiple cameras. The robot also utilizes a detailed digital map containing a high-quality 3-D road network with positions for each lane and associated traffic rules, e.g., speed limits, the priority of each road at intersections and roundabouts, and stop line positions. The map is used to predict the behavior of surrounding vehicles as well as to plan its own behavior. Photometric and shape based features for localization, e.g., lane markers, curbs, are also embedded in the map. While driving autonomously, the robot estimates its precise 6DOF pose on the map by correcting output from the Applanix-based positioning system using a feature-based localization algorithm, which extracts features from the high-definition lidar and aligns their geometry to the map.

The lidar is also used for detecting and tracking surrounding vehicles. The lidar data gives a precise 3-D point cloud that can be used to classify surrounding objects, and estimate the position and heading of surrounding vehicles, as detailed in [40]. The algorithm operates by first performing ground/obstacle discrimination of each lidar point, followed by a clustering algorithm

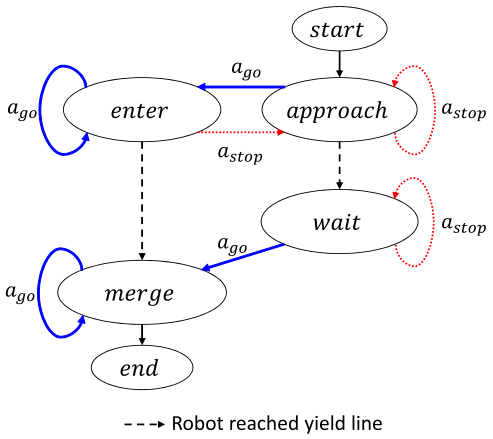

Fig. 3. Finite state machine for the roundabout situation. This FSM generates high-level commands (corresponding to the nodes) from instantaneous actions and robot position (corresponding to labels on transitions).

over obstacle points. Subsequently, point cloud clusters are used to update tracking estimates, implemented via particle filters, as well as features (of both point clouds and tracking results) which are used in the object classification process.

In addition to the perception system already mentioned, this paper introduces another perceptual cue in the form of predictions about other vehicles' decisions. This cue is explicitly learned from training data, as detailed in Section V, and its use is closely tied to the formulation of our decision making procedure as detailed in Equation (7), and explained in the surrounding text. This formulation serves to illustrate the tight bond, and interplay, between perceptual and decision making algorithms. In practical systems, it is extremely difficult to construct decision making algorithms without consideration of the perceptual algorithm performance. And it can be difficult to characterize the performance of a perceptual algorithm, and so we employ ML to create the mapping from perception output to decision.

IV. Action Planning for Roundabout

We have two main components in our action planning framework:

1) An *instantaneous policy*, mapping the current perception to an *action*. We develop ML classification techniques in Section V.
2) A FSM (Fig. 3) that ties together robot position and instantaneous actions to output high-level *commands*. The goal is to provide stable robot behavior.

The FSM is driven by actions given by classifier $\Pi = F() : \boldsymbol{z} \rightarrow \boldsymbol{A}$ that provides reasonable action $\boldsymbol{A} = \{a_{go}, a_{stop}\}$ for current environment $\boldsymbol{z}$. The action a_{go} indicates that the robot can enter roundabout safely and a_{stop} indicates that the robot can not.

A. Finite State Machine

Fig. 3 shows the structure of the FSM with four nodes: {*approach*, *enter*, *wait*, *merge*}. Transitions are dictated by inputs from the instantaneous policy and from the position of the robot relative to the yield line.

TABLE I
Mapping between FSM Nodes and High-Level Commands

Approach	The robot moves forward to stop at yield line.
enter	The robot moves forward to enter roundabout.
wait	The robot stops at yield line.
merge	The robot moves forward to pass through roundabout.

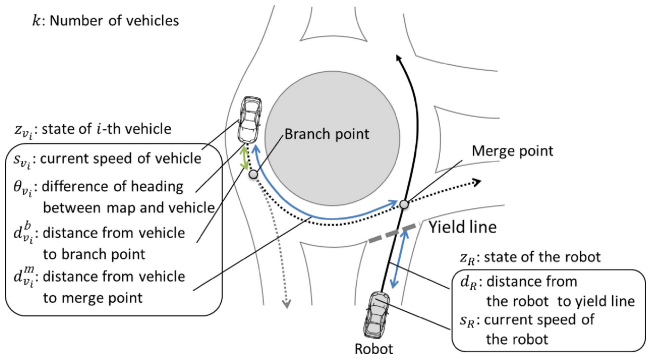

Fig. 4. An example environment illustrating notation for the roundabout situation. Details are in the text.

Each node corresponds to a high-level command, summarized in Table I. The controller will handle constraints such as:

1) Speed limits (from traffic rules, from curvature, etc.),
2) Speed control due to a leading vehicle, or vehicles that merge in front of the robot.

B. Notation

The example environment in Fig. 4 illustrates most of our notation. There is one vehicle approaching a branch point while traversing the roundabout, and the robot is approaching the entrance yield line.

Using high-quality localization and maps, the robot position and the position of detected vehicles with respect to the map are computed. The measurable state of the robot is $\boldsymbol{z}_R = (d_R, s_R)$, where d_R is the distance from the robot to yield line, and s_R is the current speed of the robot.

The set of surrounding vehicles is $\boldsymbol{V} = (v_1, v_2, ..., v_k)$, where k is the number of vehicles. For each vehicle we can measure four associated variables: s_{v_i}, θ_{v_i}, $d^m_{v_i}$, and $d^b_{v_i}$. The speed s_{v_i} of the vehicle is computed by the perception module, θ_{v_i} is the difference in heading between the vehicle and the road, $d^m_{v_i}$ is the distance from the vehicle to the merge point, and $d^b_{v_i}$ is the distance from the vehicle to the nearest branch point. The merge point is the intersection between path of the robot and the path of the vehicle. The state of an individual vehicle is $\boldsymbol{z}_{v_i}$ and the joint state is $\boldsymbol{z}_{\boldsymbol{V}}$. These will be defined as needed for each variation of the classification problem presented in Section V.

Additionally, once the prediction classifier is introduced, the vehicle state $\boldsymbol{z}_{v_i}$ will use the superscript p to indicate the state input to the prediction classifier and superscript a to indicate the state input to the action policy classifier.

V. Policy Mapping by Classification

The core of our approach is the classifier learned using a learning from demonstration methodology [38]. Using Argall's terminology, this system would be *learning by demonstration*, meaning that states and actions are the same on the robot as in the examples; and *teleoperation*, meaning the examples are performed on the actual robot with the same sensors.

It is challenging to gather a sufficient quantity of training data for a problem with such a wide variety of configurations. Additionally, there are challenges arising from partial observability due to limitations of the perception system, and with the quality of the state estimation. Next, we develop our formulation of the problem to deal with these challenges. The following Equations (1)–(4) are the general foundation for the four variations of our approach that are evaluated in this work. For clarity, we omit the robot's state in this section, and add it back in the following sections.

A. General Formulation for Multi-Object Classification

The classification problem can be written as:

$$a = F(\boldsymbol{z}_{full}) = F(\boldsymbol{z}_1, ..., \boldsymbol{z}_k) \tag{1}$$

where $\boldsymbol{z}_i$ is the state associated with object i (in our case this would be each vehicle). Ideally, a learning algorithm could be applied to learn this function directly, covering all possible states. However, the number of possible state vectors is exponential in the number of vehicles: $\#s = \#vs^{\#v}$ (where $\#vs$—number of states for each individual vehicle, $\#v$—number of vehicles). For many real-world applications, it is extremely difficult to obtain labeled training samples for such a large state space, and training with impoverished samples yields poor results as detailed below (see Section VI).

For certain cases, the classification function can be written as:

$$F(\boldsymbol{z}_1, ..., \boldsymbol{z}_k) = \Phi(f(\boldsymbol{z}_1), ...f(\boldsymbol{z}_k)) \tag{2}$$

with a *composition function* Φ, the individual classifier f, and the assumption of independence (in an information theoretic sense) between objects in the state vector $\boldsymbol{z}_i \perp\!\!\!\perp \boldsymbol{z}_j \forall i, j, i \neq j$. The composition function must respect certain properties in order to admit this factorization. For instance, with our go/stop problem, any one vehicle classifier saying stop should cause our vehicle to stop, so with $a_{go} = 1$ and $a_{stop} = 0$, we have

$$\Phi_{rb} = \prod_{i=1}^{k} f(\boldsymbol{z}_i). \tag{3}$$

The construction of other composition functions that allow factorization is beyond the scope of this paper. Given this, the number of states to be learned is equal to the number of states for any one vehicle, and the individual classifier f is defined over a much simpler state space.

There are two drawbacks to this setup:

1) Assumed independence between objects $\boldsymbol{z}_i \perp\!\!\!\perp \boldsymbol{z}_j$. This does not hold in general, and for problems like ours (and many similar), it holds only approximately, or for only a subset of the objects.
2) Assumption of no hidden information. Similar to an MDP, it is assumed that the state is fully observable (this assumption is often violated).

Even so, we would like to make use of the functional decomposition above. To accomplish this, we introduce auxiliary variables in the form of predictions of each object. This is motivated by two strands of work: the use of auxiliary variables to encode joint information in probabilistic graphical models (e.g., Junction Tree Algorithm) [41]; and the so-called "partial observability" problem from reinforcement learning [42]. It has been shown that predictions can be used as an alternate [43] or as a supplementary [44] state representation for problems with hidden state. Here, we use predictions to augment our state representation, attempting to reduce the amount of hidden information.[2]

With regard to independence between objects consider the case with $\boldsymbol{z}_\alpha \not\perp\!\!\!\perp \boldsymbol{z}_\beta$, but all other objects are independent. Then, conceptually,

$$F(\boldsymbol{z}_1, ...\boldsymbol{z}_k) = \Phi(f(\boldsymbol{z}_1), ...f(\boldsymbol{z}_\alpha, \boldsymbol{z}_\beta), ...f(\boldsymbol{z}_k))$$

and so it is only the term $f(\boldsymbol{z}_\alpha, \boldsymbol{z}_\beta)$ that must be approximated. The auxiliary variables p_α, p_β represent predictions for objects α and β, and each is meant to capture the joint and hidden information about the interaction between those objects. Then,

$$\begin{aligned}\Phi(f(\boldsymbol{z}_\alpha, \boldsymbol{z}_\beta)) = \; & \Phi(f(\boldsymbol{z}_\alpha, \boldsymbol{z}_\beta, p_\alpha, p_\beta)) \\ \approx \; & \Phi(f(\boldsymbol{z}_\alpha, p_\alpha), f(\boldsymbol{z}_\beta, p_\beta)).\end{aligned}$$

To use a single classifier, we require a prediction variable for every object. Therefore, the final form is

$$F(\boldsymbol{z}_{full}) \approx \Phi(f(\boldsymbol{z}_1, p_1), ..., f(\boldsymbol{z}_k, p_k)). \tag{4}$$

B. Multi-Vehicle Classification for Roundabout

We apply this formulation to the roundabout problem and use four variations of the classification problem developed between Equations (1)–(4). Two variations are related to the features considered for the action policy classifier, one variation introduces simple predictions for vehicles exiting the roundabout, and the final variation takes lessons learned from the first three and uses a different state representation in conjunction with ML-based predictions of vehicles exiting the roundabout. The four approaches are:

Joint	Joint representation of vehicles (Eq. (1)).
Indep.1	Assume vehicles are independent (Eq. (2)).
Indep.1.SP	Independent vehicles with prediction based on observed statistics (Eq. (4)).
Indep.2.CP	Independent vehicles with a prediction classifier (Eq. (4)) and alternative features for the action policy classifier.

The next sections present the details of constructing each state representation and classification functions. In all subsequent

[2]Note that there are many differences between this work and PSRs, most notably we do not use or define a state update equation.

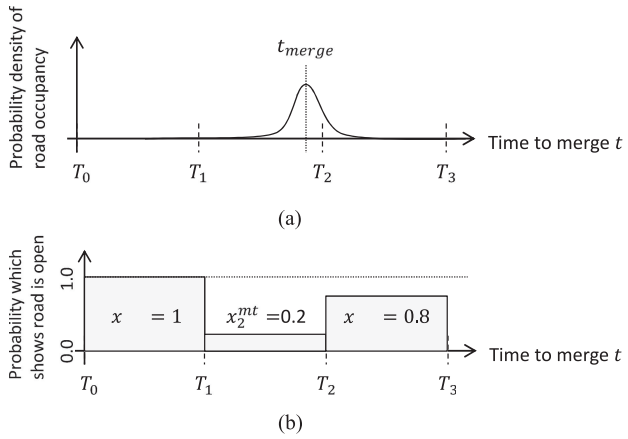

Fig. 5. Example of computing the discrete feature vector $\boldsymbol{x^{mt}}$ according to Algorithm 1 for a single vehicle and the time axis. (a) An example probability density showing vehicle occupancy along the time axis. (b) The binned feature vector for time, using the probability density above.

sections, the underlying classifier is a support vector machine (SVM) using the radial basis function kernel implemented using LibSVM [45]. Additionally, the SVM score was calculated as the distance of the sample from the decision boundary defined by the support vectors. The assumption is that a larger distance from the decision boundary indicates stronger class membership.

C. Classifier and State Representation for ***Joint***

For the joint state representation (Equation (1)), we use

$$a = F_{joint}(\boldsymbol{z}_R, \boldsymbol{z}_V)$$

where $\boldsymbol{z}_R$ is robot's state $\boldsymbol{z}_R = [d_R, s_R]$, and $\boldsymbol{z}_V = \boldsymbol{z}_{(v_1,...,v_k)}$ is a fixed-size feature vector constructed from a joint representation combining all vehicles. A fixed-size is required for the same classifier to be used for any number of vehicles.

From Section IV-B, each vehicle v_i has speed and distance to the merge point, along with a standard deviation of estimation error. We assume that estimation error is independent for each vehicle from which a probability density over:

1) distance to reach the merge point, and
2) time to reach the merge point

are computed. This is illustrated in Fig. 5(a).

We assume that merge time (and distance) is represented by a Gaussian distribution with $t_{v_i}^{merge}$ mean and σ standard deviation. Let $g(t, t_{v_i}^{merge})$ be the cumulative distribution function of occupancy at t for a ith vehicle whose merge time is $t_{v_i}^{merge}$:

$$g(t, t_{v_i}^{merge}) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{t_{v_i}^{merge} - t}{\sqrt{2\sigma^2}}\right)\right), \qquad (5)$$

where $\mathrm{erf}()$ is the Gauss error function. From this (implicitly) the probability densities are computed.

The joint representations, one for distance and one for time, each consists of binned vector x representing the combination of these probability densities over all vehicles, as illustrated in Fig. 5(b). In x, the information for all vehicles is represented simultaneously. Each feature in $\boldsymbol{x^{mt}}$ corresponds to the probability that the road is open for certain time period, computed using Algorithm 1. The same approach is used to compute the distance vector $\boldsymbol{x^{md}}$. The resulting state is $\boldsymbol{z}_V = [\boldsymbol{x^{mt}}, \boldsymbol{x^{md}}]$, showing the probability distribution along merging time and merging distance, respectively.

Algorithm 1: Create feature vector for merge timing.

Input: Vehicles $\boldsymbol{V} = (v_1, v_2, ..., v_k)$
Output: feature vector for merge time $\boldsymbol{x^{mt}}$

```
x^mt ⇐ 1 /* Initialize feature vector */
for i = 1 to k do
    t_{v_i}^{merge} ⇐ d_{v_i} / s_{v_i} /* Compute merge time */
    for j = 1 to M^{mt} do
        /* Compute occupancy probability q */
        q ⇐ g(T_j, t_{v_i}^{merge}) − g(T_{j−1}, t_{v_i}^{merge})
        x_j^{mt} ⇐ x_j^{mt}(1 − q)
    end for
end for
```

D. Classifier and State Representation for ***Indep.1***

While the joint representation (Equation (1)) can take into account the interaction between the robot and vehicle as well as interactions among vehicles, it is difficult to learn $F_{joint}()$ with limited training data. This result can be seen both theoretically as above, as well as being confirmed empirically in our experiments (see Section VI).

We argue that it is a reasonable assumption that the robot can enter the roundabout when every vehicle considered individually allows for entry in to the roundabout. From Equations (2) and (3), and the computation of the state vector from Section V-C we have:

$$\begin{aligned} F(\boldsymbol{z}_R, \boldsymbol{z}_V) &= \Phi(f(\boldsymbol{z}_1), ...f(\boldsymbol{z}_k)) \\ &= \prod_{i=1}^{k} f(\boldsymbol{z}_i) \\ &= \prod_{i=1}^{k} f_{ind}(\boldsymbol{z}_R, \boldsymbol{z}_{v_i}) \end{aligned} \qquad (6)$$

where f_{ind} is the sole classifier learned in this scheme. It computes the action for each vehicle independently of the rest, and therefore operates on a more compact state, allowing better results for learning with limited data. This representation includes the robot's state vector and the vehicle state $\boldsymbol{z}_{v_i} = [d_{v_i}^m]$ which in this case is already of constant size, so can be used directly.

E. Classifier and State Representation for ***Indep.1.SP***

The next variation incorporates simple predictions, based on frequentist inference, into the state vector in order to account for interactions between vehicles and hidden state. As in Section V-D, the input state of each vehicle for the action policy classifier is $\boldsymbol{z}_{v_i}^a = [d_{v_i}^m]$, while we are adding the new input state for the prediction $\boldsymbol{z}_{v_i}^p = [d_{v_i}^m, \theta_{v_i}]$, with $\boldsymbol{z}_{v_i} = [\boldsymbol{z}_{v_i}^a, \boldsymbol{z}_{v_i}^p]$

Combining Equations (4) and (6) gives:

$$F(\boldsymbol{z}_R, \boldsymbol{z}_V) = \prod_{i=1}^{k} f_{pred}(\boldsymbol{z}_R, \boldsymbol{z}_{v_i}, p_{v_i}). \tag{7}$$

In a roundabout, it is possible that a vehicle will exit the roundabout before it reaches the merge point as shown in Fig. 4. To handle such scenes, we predict the probability that a vehicle will stay on the roundabout, based on training data. Let this probability be

$$P(d, \theta) = \frac{K^{(d,\theta)}_{(stay)}}{K^{(d,\theta)}_{(all)}}, \tag{8}$$

where $K^{(d,\theta)}_{(\cdot)}$ shows the number of vehicles whose d^m_v and θ_v are the same as d and θ. $K^{(\cdot)}_{(stay)}$ is the number of vehicles that remain on the roundabout beyond the exit and $K^{(\cdot)}_{(all)}$ is the total number of vehicles. Then $f_{pred}()$ is defined as:

$$f_{pred}(\boldsymbol{z}_R, \boldsymbol{z}_{v_i}, p_{v_i}) = \begin{cases} a_{go} & P(d^m_{v_i}, \theta_{v_i}) < p_{th} \\ f_{ind.p}(\boldsymbol{z}_R, \boldsymbol{z}^a_{v_i}) & \text{otherwise} \end{cases}$$

where p_{th} is a probability threshold below which the vehicle is predicted to exit the roundabout. All vehicles that are predicted to exit the roundabout will be ignored, and the classifier $f_{ind.p}$ is trained on the same features as f_{ind}, but using a differently constructed training set that incorporates prediction, as detailed in Section VI-A.

F. Classifier and State Representation for ***Indep.2.CP***

The final variation on our proposed approach improves on the previous variations in two ways. First, in order to improve the exit predictions for the vehicles, a more sophisticated ML classifier is trained to make the prediction C, where $C \in \{stay, exit\}$. The observation state of each vehicle $z^p_{v_i}$ is changed such that the notion of distance $d^b_{v_i}$ indicates the distance to the nearest branch point, or exit, instead of the distance to the merge point with the robot, leading to $z^p_{v_i} = [d^b_{v_i}, \theta_{v_i}, s_{v_i}]$. This combination of features is provided as input to an SVM classifier f_p such that

$$C(\boldsymbol{z}^p_{v_i}) = f_p(d^b_{v_i}, \theta_{v_i}, s_{v_i}) \tag{9}$$

where the output C_{v_i} represents the prediction of whether the ith vehicle will be exiting the roundabout. Using the prediction from Equation (9), $f_{pred}()$ is redefined as:

$$f_{pred}(\boldsymbol{z}_R, \boldsymbol{z}_{v_i}, p_{v_i}) = \begin{cases} a_{go} & C(\boldsymbol{z}^p_{v_i}) = exit \\ f_{ind.p}(\boldsymbol{z}_R, \boldsymbol{z}^a_{v_i}) & C(\boldsymbol{z}^p_{v_i}) = stay. \end{cases}$$

The second improvement in this variation of our approach is that input state to the action policy classifier is extended to include the speed s_{v_i} of the other vehicles, such that $\boldsymbol{z}^a_{v_i} = [d^m_{v_i}, s_{v_i}]$.

VI. Experimental Results

In this section we detail 1) the construction of our data set; 2) experimental results for our learned classifier on the data set; and 3) experimental results on real-world tests using the learned classifier. The generation of high-quality data sets is critical to the success of the algorithm, as discussed next.

A. Data Set Generation and Labeling

We have implemented a hybrid methodology for generating data which we call *semi-automated* driving, in which the human and autonomous vehicle are operating simultaneously, in complementary roles. This data collection method addresses certain correspondence issues that arise from the mismatch between the human perception and action spaces and the autonomous vehicle perception and action spaces [38]. In semi-automated driving, a human operator acts as the action policy classifier in real time, giving high-level commands (a_{go} and a_{stop}) to the car, which are processed by the FSM. The advantage is that, after some amount of practice, the human operator will provide reasonable actions to compensate for system delay such as the latency of the controller and hardware. This approach is necessary in order to allow the decision classifier to incorporate the controller delays into the learned decision model. In a few rare instances (5% of the time), the driver made a mistake leading to a transition to manual driving while in interesting traffic configurations that seemed valuable for training. In these sequences, labeling was done as manual post-processing. We determined that rare scenarios were still useful for our process, and as such the reduced accuracy (in terms of timing of the transition from a_{stop} label to a_{go} label) of the manual labeling was offset by the novelty of the training example.

We took 105 sequences of executing a roundabout maneuver utilizing five different entrances of three separate roundabouts. Because each sequence contains many individual frames, the total size of the data set is 4327 frames. Due to sensor range limitations for tracking vehicles on the far side of the roundabout, as well as the need for decision-making only when approaching the yield line, our training and test sequences begin only once we are within 15 m of our entrance to the roundabout. Even at this distance, there are a_{stop} examples even though there is no vehicle perceived on the roundabout. Such cases are caused when a vehicle is not yet detected by the perception system, but is perceived by the human operator.

The labels as given are for an entire (joint) scene, but our ***Indep.1***, ***Indep.1.SP***, and ***Indep.2.CP*** data sets require a label for each vehicle within a scene. Consider a situation with three vehicles in a roundabout, it is not generally the case that all three of the vehicles are simultaneously causing the robot to stop, even though the given label is stop. We use the following scheme to generate *samples*: the actual data used to train the classifier.

*1) Samples for **Joint**:* We use training data as labeled to construct joint features. Each frame provides one sample for training.

*2) Samples for **Indep.1**:* For each frame with a_{go} label, generate one sample from each vehicle. For each frame with a_{stop} label, generate a sample from the vehicle with smallest $\boldsymbol{d}_i$. The assumption is that stop decision is dependent on the closest vehicle.

TABLE II
NUMBER OF SAMPLES FOR LEARNING A CLASSIFIER

Method	Number of a_{go}	Number of a_{stop}
Joint	3240	1087
Indep.1	5774	1051
Indep.1.SP	4191	1024
Indep.2.CP	2901	1055

TABLE III
CLASSIFICATION RESULT FOR DISTANCE UP TO 15 METERS

Method	Overall success	Failure of a_{go}	Failure of a_{stop}	# Decision changes
Joint	76.89%	12.13%	10.98%	246
Indep.1	82.30%	15.16%	2.54%	**125**
Indep.1.SP	87.27%	9.98%	2.75%	140
Indep.2.CP	**92.06%**	**5.59%**	**2.40%**	132

*3) Samples for **Indep.1.SP**:* First remove every vehicle whose $P(v_i)$ is less than threshold, because an exiting vehicle should not affect decision making. Then apply the same scheme as in ***Indep.1***.

*4) Samples for **Indep.2.CP**:* A two-stage learning approach is used to more intelligently assign credit for an a_{stop} label when there are multiple vehicles on the roundabout. In the first stage, all frames in which only a single vehicle exists are extracted and used to train an intermediate SVM for the single-vehicle case in which there is no ambiguity about the source of an a_{stop} label. In the second stage, samples are created as in ***Indep.1.SP*** and in ***Indep.1***. The primary difference is that instead of using the smallest $\boldsymbol{d}_i$ as the criteria for a_{stop} label assignment, the single-vehicle SVM trained in stage one is used to provide a score to each vehicle. The vehicle with the strongest a_{stop} according to the SVM score is assigned the a_{stop} label, and all other vehicles from that frame are ignored for training purposes.

Table II shows the number of samples in training data.

B. Classification Results

We evaluate the four learned classifiers ***Joint***, ***Indep.1***, ***Indep.1.SP***, and ***Indep.2.CP*** on the data set. Classification methods are evaluated by five-fold cross validation.

The feature vector ***Joint*** is constructed by partitioning merge times from 0 to 9 s every 1 s (nine dimensions for $\boldsymbol{x^{mt}}$) and partitioning merge distance from 0 to 60 m every 10 m (six dimensions for $\boldsymbol{x^{md}}$). In total, the length of feature for ***Joint*** is 17, including robot state. For both ***Indep.1*** and ***Indep.1.SP***, we found that the classifiers performed better using just the robot state and the vehicle position (ignoring estimated velocity). Therefore, feature state length is 3. On the other hand, for ***Indep.2.CP*** the estimated vehicle velocity improved the performance of the classifier. For ***Indep.1.SP***, the prediction $P(d^m_{v_i}, \theta_{v_i})$ (Equation (8)) is estimated from the frequency of observations in the training data using discrete bins on distance and angle. Distance $d^m_{v_i}$ is partitioned into 15.0 m bins from 0.0 to 60.0 m, and θ_{v_i} every 0.2 rad from -0.5 to 0.5. With the ***Indep.1.SP*** classifier, the prediction threshold is $p_{th} = 0.1$.

The results of frame by frame classification are in Table III. This offline analysis compares the different approaches quantitatively to understand the effect of different formulations and predictions on the quality of the classification. It is unrealistic to expect a ML algorithm to achieve 100% accuracy, thus the full system must be able to handle transient errors from the classification. A qualitative analysis of the effect of imperfect classification on the failure rate of the full autonomous vehicle system is presented in Section VI-C.

The right column of Table III shows the total number of decision changes. In this evaluation set, there are 52 decision changes out of 105 sequences. This is because a decision change is not counted when human operator commands a_{go} prior to 15 m before the yield line, then maintains a_{go} through the sequence. A large number of decision changes indicates less stability in decision making which can lead to hesitations in the robot and ultimately incorrect behavior. Overall, ***Joint*** has poorer performance both in terms of classification accuracy and in number of decision changes. This is due to both higher feature dimension and a small number of training samples.

By introducing even simple predictions, ***Indep.1.SP*** showed a significant boost in performance over ***Joint*** and ***Indep.1***. With the lessons learned from ***Joint***, ***Indep.1***, and ***Indep.1.SP*** and incorporated into ***Indep.2.CP***, it showed the best performance overall, with significantly better results on a_{go} and slightly better for a_{stop}. This indicates that ***Indep.2.CP*** was more responsive without sacrificing safe decisions. With the substantial improvement in performance for both ***Indep.1.SP*** and ***Indep.2.CP***, it is clear that predictions are of benefit when constructing the classifier.

Fig. 6 shows the performance of classifiers at different distances. Most errors occur when the robot is beyond 10 m from the entrance of roundabout. Therefore, we also evaluate on subset of the full data set. Table IV shows results when the distance to yield line is less than 10 m. We have labeled data for 3135 frames and there are 22 ground truth decision changes. All classifiers are improved in this case, with ***Indep.2.CP*** the clear winner.

Fig. 7 shows a real-world example that illustrates classifier behavior over time (frame a) and shows the perceived roundabout situation at three key time points (frames b–d) for a busy roundabout. Overall, ***Joint*** outputs fluctuate significantly. Both ***Indep.1*** and ***Indep.1.SP*** provides better result than ***Joint***, however, the reasons for failure are different. In Fig. 7(b), ***Indep.1.SP*** failed because $P(\boldsymbol{z}_{v_1})$ is smaller than p_{th} meaning the prediction is that the vehicle will exit, when in fact **v1** will actually stay on the roundabout. Of note, is the quick recovery from the wrong decision once better evidence is obtained about the intended destination of **v1**. In Fig. 7(c), ***Indep.1.SP*** outputs a_{go} correctly since $P(\boldsymbol{z}_{v_2}) < p_{th}$, meaning the exit is correctly predicted. On the other hand in both ***Joint*** and ***Indep.1***, classifiers output a_{stop} because of **v2** even though **v2** will exit from roundabout. In Fig. 7(d), ***Indep.1*** outputs a_{go} because **v2** completely

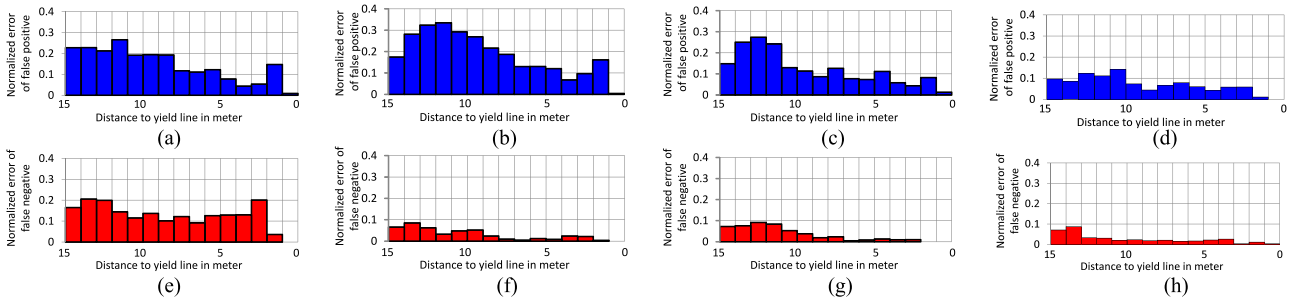

Fig. 6. Errors along distance. Top row shows failure to predict a_{go} at different distances, and bottom row shows failure to predict a_{stop}. (a) Failure of a_{go} for ***Joint***. (b) Failure of a_{go} for ***Indep.1***. (c) Failure of a_{go} for ***Indep.1.SP***. (d) Failure of a_{go} for ***Indep.2.CP***. (e) Failure of a_{stop} for ***Joint***. (f) Failure of a_{stop} for ***Indep.1***. (g) Failure of a_{go} for ***Indep.2.CP***. (h) Failure of a_{go} for ***Indep.2.CP***.

TABLE IV
CLASSIFICATION RESULT FOR DISTANCE UP TO 10 METERS

Method	Overall success	Failure of a_{go}	Failure of a_{stop}	# Decision changes
Joint	83.00%	8.20%	8.80%	139
Indep.1	88.26%	10.49%	1.24%	72
Indep.1.SP	93.14%	5.93%	**0.93%**	61
Indep.2.CP	**95.21%**	**3.58%**	1.20%	**55**

exits the roundabout. It should be noted that decision to go was delayed 1.5 s by ***Indep.1*** as compared to labeled data. The delay is 0.8 s by ***Indep.1.SP***. Compared to the above, ***Indep.2.CP*** was also able to make the correct decision at the same time as ***Indep.1.SP***, but due to better predictions about the vehicle's destination ***Indep.2.CP*** is able to avoid the decision fluctuation that occurs at $t = -2.4$ with ***Indep.1.SP***.

Causes of decision failures: In terms of causes for failures, an analysis of these experiments found the following causes:

1) Classifier makes error because of limited perception range.
2) Classifier outputs a_{go} even though the robot is following another vehicle. In this case, human operator did not label as a_{go}, but the classifier may output a_{go} because roundabout is safe to go for the front vehicle.
3) Slightly different transition time between human label and classifier, see Fig. 7(a).
4) Error in prediction can cause bad decisions (for ***Indep.1.SP*** and ***Indep.2.CP***).
5) Unexpected behavior from other vehicles on the roundabout (e.g., a vehicle already on the roundabout incorrectly stopped to yield to the robot waiting to enter the roundabout).
6) Classifier made error because other vehicle entering the roundabout had low velocity (see Section VI-C2 for discussion).

One particular failure that had a significant effect on the final classification results happened during a single scenario in which a vehicle unexpectedly stopped on the roundabout to yield to the robot. This is an incorrect behavior by the vehicle and in response, the safety driver of the robot providing the reference decision was cautious and also remained stopped, so in this case the labels in the classification data set were a_{stop}, even though the vehicle's behavior would indicate a_{go}, based on velocity. The ***Joint***, ***Indep.1***, and ***Indep.1.SP*** classifiers all stopped, but the ***Indep.2.CP*** classifier decided to go because it considered the other vehicle's velocity. One could argue that this was not actually a failure since it promoted the safe flow of traffic in the face of unpredictable behavior. Unfortunately, the decision disagreed with the human-provided label for several frames, causing a questionable decrease in overall accuracy. In Table V and VI, the classification results are presented as if this scenario were decided correctly. In this case, the positive effect of the more sophisticated prediction model and expanded input feature vector to the ***Indep.2.CP*** action policy classifier becomes even more obvious. This improved result is particularly apparent for failures in classifying a_{stop}, which are the the most important decisions to get correct to avoid collisions.

This scenario actually highlights a risk that comes with using humans to provide examples, particularly in challenging decision making scenarios such as a roundabout. Care must be taken to verify that the provided examples are consistent examples that will lead to a good decision policy. Alternatively, a more robust approach would be to leave the inconsistent examples in the data set and ensure that the system can safely handle such scenarios.

C. Full-System Experiment With Dense Traffic

In order to confirm the effectiveness of our approach, we performed two experiments in which we completed a series of full system tests, running the whole processing pipeline from perception to controller. In the first experiment, we used ***Indep.1.SP*** as the action policy classifier within the action planning module and in the second experiment we used ***Indep.2.CP*** as the action policy classifier.

*1) Real-World Experiment With **Indep.1.SP**:* Experiment 1 had 61 trials to enter the roundabout autonomously for two different entrances of a roundabout at approximately 5 P.M. on a weekday. This time of day typically presents challenging traffic scenarios due to increased traffic volume. We have evaluated both the commands from the action planner, as well as the overall system performance. We define a failure of classification when the classifier did not output a sequence of acceptable decisions through a trial, and a full-system failure when the robot does

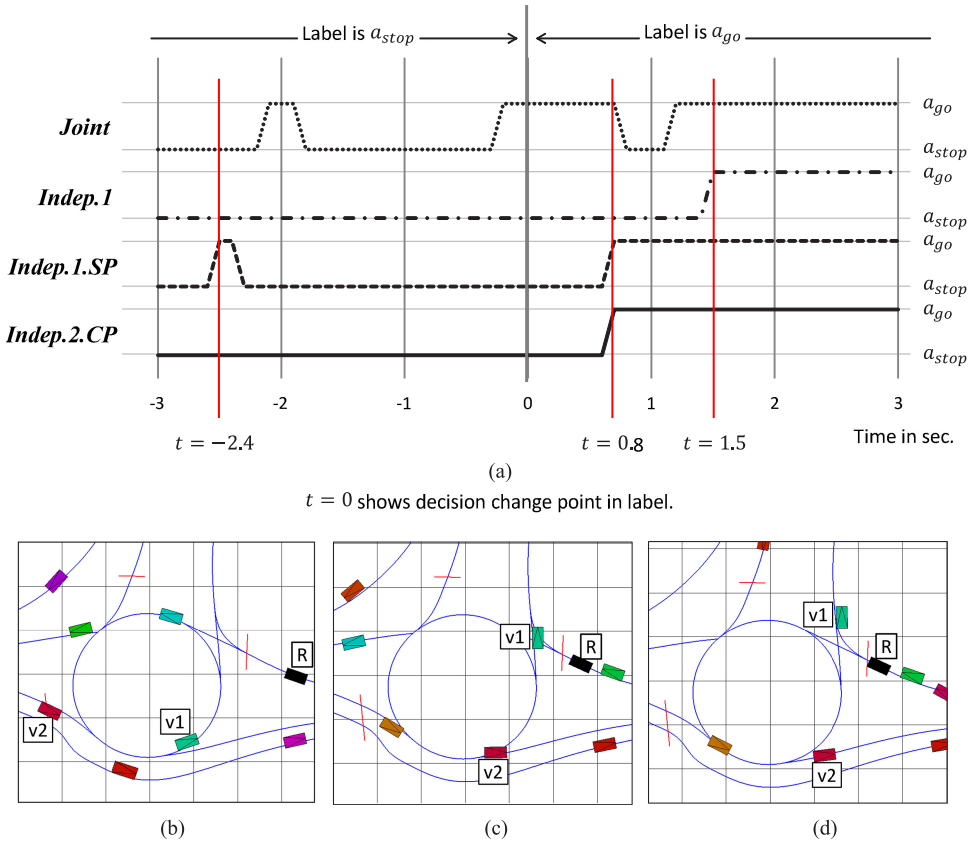

Fig. 7. Example of classification along time. (a) Shows outputs from classifiers along time. $t = 0$ shows that label is changed from a_{stop} to a_{go} by the human driver. (b)–(d) Show corresponding scenes at $t = -2.4$, $t = 0.8$ and $t = 1.5$. **R** shows the robot, **v1** and **v2** show vehicles on roundabout, respectively.

TABLE V
CLASSIFICATION RESULT WITHOUT YIELD INCIDENT—15 METERS

Method	Overall success	Failure of a_{go}	Failure of a_{stop}	# Decision changes
Joint	76.89%	12.13%	10.98%	246
Indep.1	82.30%	15.16%	2.54%	**125**
Indep.1.SP	87.27%	9.98%	2.75%	140
Indep.2.CP	**92.86%**	**5.59%**	**1.54%**	129

TABLE VI
CLASSIFICATION RESULT WITHOUT YIELD INCIDENT—10 METERS

Method	Overall success	Failure of a_{go}	Failure of a_{stop}	# Decision changes
Joint	83.00%	8.20%	8.80%	139
Indep.1	88.26%	10.49%	1.24%	72
Indep.1.SP	93.14%	5.93%	0.93%	61
Indep.2.CP	**96.32%**	**3.58%**	**0.10%**	**52**

not successfully enter the roundabout without a takeover by the safety driver.

Fig. 9 (top) shows the overall result from real-world Experiment 1. The robot made reasonable decisions for 58 trials (95.1% success). Of these, seven trials were easy since there were no other vehicles in the roundabout. There were some vehicles in the roundabout in 51 trials, and the classifier outputted a sequence of actions that should have resulted in reasonable behavior, however there were other system issues as discussed below.

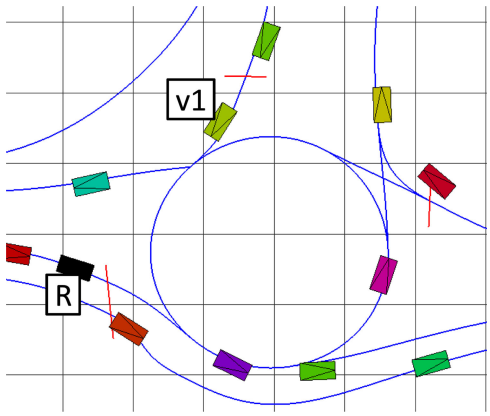

Fig. 8. Example of dense traffic encountered during the full-system road tests.

On the other hand, the ***Indep.1.SP*** classifier failed to provide reasonable decisions in three trials. Two failures were caused by incorrect predictions of other vehicle behavior. In one example, a prediction for a vehicle becomes lower than threshold, leading the action policy classifier to ignore the vehicle, even though the

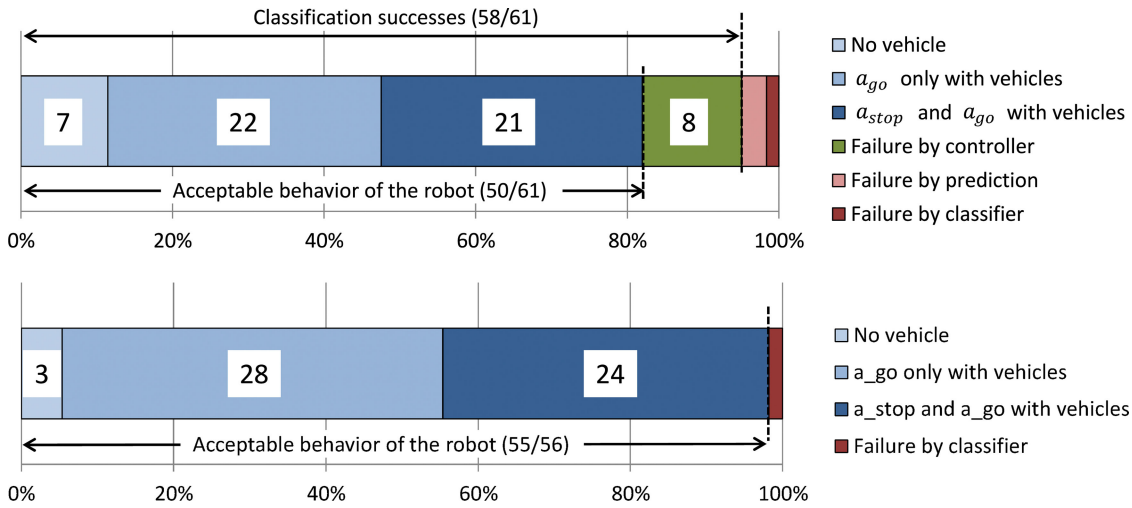

Fig. 9. Result of road testing. (top) Classifier using ***Indep.1.SP*** successfully made reasonable decisions for 58 out of 61 trials (95.1%). The robot successfully entered the roundabout for 50 out of 61 trials (82.0%). Detailed discussion is in Section VI-C1. (bottom) Result of experiment with ***Indep.2.CP***. The classifier successfully made reasonable decisions and successfully entered the roundabout for 55 out of 56 trials (98.2%). Detailed discussion is in Section VI-C2.

vehicle will not exit the roundabout. One failure was caused by fluctuating decisions as the input feature vector stayed near to the decision boundary, causing the vehicle to miss the window for a safe merge.

From the aspect of overall autonomous driving behavior, the robot successfully entered the roundabout in 50 trials. Fig. 8 shows an example of dense traffic at the roundabout. The robot successfully yields to vehicle **v1**. The high-level decision was *approach* according to current speed (2.15 m/s) and distance to yield line (4.95 m). Failures caused by the controller means that the high-level command was not executed correctly. Analysis showed that the system experienced highly variable amounts of latency in executing high-level commands, and the system failures were caused by significantly longer latencies than average. This typically occurred when the robot started moving from complete stop. This was the cause of all eight controller issues.

*2) Real-World Experiment With **Indep.2.CP**:* Experiment 2 was composed of 56 trials entering the same roundabout as in Experiment 1, also at 5 P.M. on a weekday. A similar evaluation to Experiment 1 was completed for Experiment 2, but in between Experiments 1 and 2, the variability in the latency of the controller was corrected, meaning that the latency and latency variance in executing the high-level commands were both reduced considerably. Consequently, all failures in Experiment 2 can be attributed to decision making.

Fig. 9 (bottom) shows the overall results from Experiment 2. The robot made reasonable decisions for 55 out of 56 trials (98.2% success). Of these 56 trials, three were trivial as no other vehicles were present in the roundabout. For 53 of the trials, there were vehicles present in the roundabout 52 of which resulted in a reasonable sequence of decisions from the action policy classifier.

The lone failure of the action policy classifier came from an incorrect decision due to the fact that a vehicle entering the roundabout from its entrance proceeding to the robot entrance slowed to a stop at the entrance yield line and then accelerated rapidly. While our classifier considers the vehicle's velocity, it does not predict the enter or stop decisions of *other* vehicles if they are entering the roundabout, thus the model as formulated cannot account for this particular scenario. The reason for failure is that the vehicle's velocity at the time was near zero (observable by the robot, leading to a decision to go), but with an acceleration imminent (and of course unknown to the robot, leading to a potential collision). This is one example that highlights the complexity of decision making in roundabouts. Despite this issue, the overall real-world success rate remained quite high at 98.2%.

VII. Discussion

While we developed and demonstrated our approach on the specific example of safely entering roundabouts, we believe that the approach is general, and can be applied to many situations in autonomous driving. Further, the remaining issues present in the roundabout situation are useful to examine in the context of general automated driving applications.

As discussed in Section VI, reliable lidar perception is a limiting factor of the perception and decision making system as a whole. The classification problem is especially difficult at large distances due to the sparseness of the point cloud, and also when the object is sufficiently close but partially occluded. A similar limitation is also illustrated in [40], [46]. This issue is more general, whether using a lidar or camera: the fundamental issue with sensing is the need for adequate resolution at a suitable range of distances. Operating conditions must also be considered. A lidar point cloud might be negatively affected with noise due to rain or snow, and a camera may produce a wide range of image appearances due to differing illumination conditions. Thus, perception algorithms must be designed to cover all potential variations. It is also important to have sufficient viewing angles for unobstructed sensing. Relying on front-facing cameras

Fig. 10. An intersection somewhere in India. This is an example of how complex the perception problem might be in real-life traffic situations.

Fig. 11. This is an example of a blind corner in Japan, with a bicyclist suddenly appearing (in the yellow circle) and the planned direction of the vehicle shown by the arrow.

exclusively for doing all of the perception tasks while attempting to negotiate the roundabout safely would not be sufficient because observations of other vehicles must sometimes be made at angles exceeding 90 degrees from the robot's longitudinal axis.

While adequate perception is important, it is only the first step toward successfully negotiating challenging driving situations such as roundabouts. The interface, or information passed between perception and decision making is crucial to designing a robust system.[3] The decision making system must have an extremely robust design on two levels. First, it must be robust to the wide range of real-world situations that may be encountered. Second, it must be robust to the limitations and error modes of perception. In other words, it should be expected that perception is not perfect, and those imperfections should be identified, modeled where possible, and accounted for. In the case study, this is done through the use of ML, by having the system learn against ground-truth decisions while running the (imperfect) perception system. Thus, decision making should, with enough data, learn those situations that will be imperfectly sensed. Even with more advanced sensing on future vehicles, some driving situations may still be devilishly hard for any computer to perceive and act in. For example, in Fig. 10 many kinds of objects appear to be moving in all directions while automotive vehicles are trying to negotiate the intersection safely. Another example is shown in Fig. 11 in which an obstacle suddenly appears from the area not easily observable by on-board sensors. Fortunately, each of these situations admits a simple action plan: reducing the speed to move reasonably slowly toward the goal so that the car could safely stop if a collision probability becomes very high. This action plan accounts for potentially significant uncertainty of perception as well as limitations in our ability to predict accurately behavior of other agents in the driving environment (prediction accuracy limitations of our classifiers were also illustrated in Section VI).

In some driving situations such as shown in Fig. 10, a construction zone, and others in which many different kinds of agents such as vehicles, humans, and even animals might be interacting in non-trivial ways, it is possible that increased computation in the form of cloud computing could provide better results. The on-board sensors would broadcast partially processed data to the cloud for detailed scene analysis. The results of an in-cloud analysis are then sent back to the vehicle, as demonstrated in [47]. Such cloud robotics, in the form of perception support, has yet to receive significant attention in the area of automotive vehicles but some have been proposing to integrate vehicle control with cloud computing already. In [48] it is described how the cloud support could improve a number of vehicle control and diagnostic functions. It is proposed to outsource computation-intensive tasks to the cloud if such tasks are not time critical. For example, route optimization could be carried out in the cloud in terms of fuel economy and other constraints. The ride quality of the vehicle could also be improved by sharing information about the road surface quality from the vehicles which have already passed through the same road segment.

[3]Our case study and system were designed to address this, and we explored the importance of the interface between decision making and actuating the vehicle in Section VI-C.

VIII. Conclusion

There are difficult challenges on the road toward ever more increasing vehicle automation. By studying highly automated driving for a challenging roundabout situation we have illustrated a practical way to address important challenges in realizing the perception and decision making system. We have developed a ML approach for high-level decision making that enabled us to deal with partial observability, perception limitations, and agent (vehicle) interactions successfully.

The use of ML methods also demonstrates the importance of data-driven development. Though we employed SVM classifiers successfully, other ML methods are also applicable. We have applied learning from demonstration for policy development by utilizing semi-automated driving in which the human and the robot were operating in complementary roles. This allowed us to accelerate the system development in the case of negotiating the roundabout, but it would naturally apply to other cases as examples of safe human decisions are abundant and easy to incorporate in the development process. It also allows further refinement of the decision making system upon obtaining more data, and also upon changes to the perception system. Due to the low cost of re-training, we can improve the system

by incorporating new data from novel and problematic driving situations. Thus, we believe that our approach allows systems to be developed for a wide range of situations, and is a useful, generalizable architecture for automated driving applications.

REFERENCES

[1] Google self-driving car project [Online]. Available: https://www.google.com/selfdrivingcar/

[2] J. Ziegler *et al.*, "Making bertha drive: An autonomous journey on a historic route," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 2, pp. 8–20, Summer 2014.

[3] E. Dickmanns, *Dynamic Vision for Perception and Control of Motion*. New York, NY, USA: Springer, 2007.

[4] G. A. Pratt, "Is a cambrian explosion coming for robotics?" *J. Econ. Perspectives (also posted online by IEEE Spectrum)*, vol. 29, no. 3, pp. 51–60, 2015.

[5] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner, "Three decades of driver assistance systems: Review and future perspectives," *IEEE Intell. Transp. Syst. Mag.*, vol. 6, no. 4, pp. 6–22, Winter 2014.

[6] Mercedes s500 distronic plus system . (2015). [Online]. Available: http://500sec.com/distronicdistronic-plus/

[7] M. Aeberhard, S. Rauch, M. Bahram, G. Tanzmeister, J. Thomas, Y. Pilat, F. Homm, W. Huber, and N. Kaempchen, "Experience, results and lessons learned from automated driving on Germanys highways," *IEEE Intell. Trans. Syst. Mag.*, vol. 7, no. 2, pp. 42–57, Spring 2015.

[8] Volvo drive me project. (2014). [Online]. Available: https://www.media.volvocars.com/global/en-gb/media/pressreleases/145619

[9] Mobileye driver assistance solutions. (2012). [Online]. Available: http://www.mobileye.com/

[10] M. Bertozzi, A. Broggi, A. Coati, and R. I. Fedriga, "A 13,000 km intercontinental trip with driverless vehicles: The VIAC experiment," *IEEE Intell. Transp. Syst. Mag.*, vol. 5, no. 1, pp. 28–41, Spring 2013.

[11] A. Broggi, P. Cerri, S. Debattisti, M. C. Laghi, P. Medici, M. Panciroli, and A. Prioletti, "PROUD-public road urban driverless test: Architecture and results," in *Proc. IEEE Intell. Vehicles Symp.*, Dearbon, MI, USA, 2014, pp. 684–654.

[12] Stanford autonomous driving [Online]. Available: http://driving.stanford.edu/papers.html

[13] J. Levinson *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, 2011, pp. 163–168.

[14] Gm-cmu collab res lab [Online]. Available: http://gm.web.cmu.edu/

[15] J. Wei, J. M. Snider, J. Kim, J. M. Dolan, R. Rajkumar, and B. Litkouhi, "Towards a viable autonomous driving research platform." in *Proc. IEEE Intell. Vehicles Symp.*, 2013, pp. 763–770.

[16] Citymobil project [Online]. Available: http://www.citymobil-project.eu/

[17] Dynamic interactive perception-action learning in cognitive systems (diplecs) project. (2010). [Online]. Available: http://www.diplecs.eu/

[18] Automated driving applications and technologies for intelligent vehicles (adaptive) project [Online]. Available: http://www.adaptive-ip.eu/

[19] T. Michalke, J. Fritsch, and C. Goerick, "A biologically-inspired vision architecture for resource-constrained intelligent vehicles," *Comput. Vision Image Understanding*, vol. 114, no. 5, pp. 548–563, 2010.

[20] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "A committee of neural networks for traffic sign classification," in *Proc. Int. Joint Conf. Neural Netw.*, 2011, pp. 1918–1921.

[21] V. John, K. Yoneda, B. Qi, Z. Liu, and S. Mita, "Traffic light recognition in varying illumination using deep learning and saliency map," in *Proc. 17th Int. IEEE Conf. Intell. Transp. Syst.*, 2014, pp. 2286–2291.

[22] A. Angelova, A. Krizhevsky, and V. Vanhoucke, "Pedestrian detection with a large-field-of-view deep network," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 704–711.

[23] S. Sivaraman and M. Trivedi, "Integrated lane and vehicle detection, localization, and tracking: A synergistic approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 906–917, Jun. 2013.

[24] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3D traffic scene understanding from movable platforms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 1012–1025, May 2014.

[25] C. Urmson, J. Anhalt, D. Bagnell, C. Baker, R. Bittner, M. Clark, J. Dolan, D. Duggins, T. Galatali, C. Geyer *et al.*, "Autonomous driving in urban environments: Boss and the urban challenge," *J. Field Robot.*, vol. 25, no. 8, pp. 425–466, 2008.

[26] P. Abbeel, D. Dolgov, A. Ng, and S. Thrun, "Apprenticeship learning for motion planning with application to parking lot navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 1083–1090.

[27] J. Levinson and S. Thrun, "Unsupervised calibration for multi-beam lasers," in *Experimental Robotics*. New York, NY, USA: Springer, 2014, pp. 179–193.

[28] D. Held, J. Levinson, S. Thrun, and S. Savarese, "Robust real-time tracking combining 3D shape, color, and motion," *Int. J. Robot. Res.*, 35 (1–3), pp. 30–49, 2016.

[29] S. Kammel, J. Ziegler, B. Pitzer, M. Werling, T. Gindele, D. Jagzent, J. Schröder, M. Thuy, M. Goebl, F. von Hundelshausen *et al.*, "Team AnnieWAY's autonomous system for the 2007 DARPA Urban Challenge," *J. Field Robot.*, vol. 25, no. 9, pp. 615–639, 2008.

[30] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion," in *Proc. IEEE Intell. Vehicles Symp.*, 2009, pp. 215–220.

[31] F. Moosmann and C. Stiller, "Joint self-localization and tracking of generic objects in 3D range data," in *Proc. IEEE Int. Conf. Robot. Autom.,* 2013, pp. 1146–1152.

[32] A. Chatham, M. Montemerlo, C. Urmson, H. Dalhkamp, and J. Zhu, "Building Elevation Maps From Laser Data," U.S. Patent 8 825 391, Sep. 2, 2014.

[33] D. Dolgov, J. Zhu, and N. Fairfield, "Determination of object heading based on point cloud," U.S. Patent 9 014 903, Apr. 21, 2015.

[34] J. P. Rastelli, V. Milanés, T. De Pedro, L. Vlacic, "Autonomous driving manoeuvres in urban road traffic environment: A study on roundabouts," in *Proc. 18th World Congr. Int. Fed. Autom. Control*, 2011, pp. 1–5.

[35] M. A. Ali, M. Mailah, and T. H. Hing, "Path planning of mobile robot for autonomous navigation of road roundabout intersection," *Int. J. Mech.*, vol. 6, no. 4, pp. 203–211, 2012.

[36] D. O. Sales, L. C. Fernandes, F. S. Osório, and D. F. Wolf, "FSM-based visual navigation for autonomous vehicles," presented at the Workshop Visual Control Mobile Robots-IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Vilamoura, Algarve, Portugall, 2012.

[37] M. Muffert, D. Pfeiffer, and U. Franke, "A stereo-vision based object tracking approach at roundabouts," *IEEE Intell. Transp. Syst. Mag.*, vol. 5, no. 2, pp. 22–32, Summer 2013.

[38] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, no. 5, pp. 469–483, 2009.

[39] D. Silver, J. A. Bagnell, and A. Stentz, "Learning from demonstration for autonomous navigation in complex unstructured terrain," *Int. J. Robot. Res.*, vol. 29, no. 12, pp. 1565–1592, 2010.

[40] M. Delp, N. Nagasaka, N. Kamata, and M. James, "Classifying and passing 3D obstacles for autonomous driving," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, 2015, pp. 1240–1247.

[41] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[42] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, no. 1, pp. 99–134, 1998.

[43] S. Singh, M. R. James, and M. R. Rudary, "Predictive state representations: A new theory for modeling dynamical systems," in *Proc. 20th Conf. Uncertainty Artif. Intell.*, 2004, pp. 512–519.

[44] M. R. James, B. Wolfe, and S. P. Singh, "Combining memory and landmarks with predictive state representations," in *Proc. 19th Int. Joint Conf. Artif. Intell.*, 2005, pp. 734–739.

[45] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, pp. 27:1–27:27, 2011, [Online]. Available: http://www.csie.ntu.edu.tw/ cjlin/libsvm

[46] D. V. Prokhorov, "Overview of CI research in automotive ITS," in *Proc. IEEE Symp. Comput. Intell. Vehicles Transp. Syst.*, Paris, France, Apr. 11–15, 2011, pp. 1–7.

[47] S. Kumar, S. Gollakota, and D. Katabi, "A cloud-assisted design for autonomous driving," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, 2012, pp. 41–46.

[48] D. Filev, J. Lu, and D. Hrovat, "Future mobility: Integrating vehicle control with cloud computing," *ASME Dynamic Syst. and Control Mag.*, vol. 1, pp. 18–24, Mar. 2013.

Authors' photographs and biographies not available at the time of publication.