2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)

WiDrive: Adaptive WiFi-based Recognition of Driver Activity for Real-time and Safe Takeover

Yunhao Bai†, Zejiang Wang*, Kuangyu Zheng†, Xiaorui Wang†, and Junmin Wang*
†Department of Electrical and Computer Engineering, The Ohio State University, USA
*Department of Mechanical Engineering, University of Texas at Austin, USA

***Abstract*—Autonomous vehicles often need human driver to take over in some complicated conditions. Such a sudden takeover could jeopardize the vehicle's safety and stability if not handled properly. Hence, if the driver's takeover intention can be recognized as early as possible, the vehicle can have sufficient time to make important takeover preparation. The existing in-car monitoring systems are mostly based on camera, which have several key limitations, such as brightness condition and motion obscurity. On the other hand, WiFi-based wireless sensing has recently shown a great promise in human activity recognition, but mainly for large-scale movements performed in the room environment.**

In this paper, we propose WiDrive, a real-time in-car driver activity recognition system based on Channel State Information (CSI) changes of WiFi signals. WiDrive consists of three major components: A novel algorithm to extract small-scale in-car human activity features, a real-time recognition system based on Hidden Markov Model (HMM), and an online adaptation algorithm to adapt for different drivers and vehicles. We implement WiDrive with commercial WiFi devices and evaluate it in real cars. Our results show that WiDrive has an average recognition accuracy of 91.3% and improves the takeover safety.

I. Introduction

Autonomous vehicles have received increasing research attention in recent years. Despite the ideal objective of being 100% autonomous, many such vehicles often need to switch between the self-driving and human-driven modes by performing the "takeover" action. For example, the human driver may desire to take over the driving wheel in some complicated conditions (e.g., when another car ignores the red light or unexpectedly stops ahead due to severe weather or accidents). Likewise, when the human driver is unable to safely drive the vehicle (e.g., due to the influence of alcohol, drug, or fatigue), future autonomous vehicles may force a takeover back to the self-driving mode. In either case, the human driver could suddenly make a movement at any time that either is a takeover attempt itself (former case) or can trigger one if necessary (latter case). Such a sudden takeover, if not handled properly, can generate potential risks, because the autonomous vehicle may lose its stability and result in undesired accidents. From the mechanical perspective, it is highly desirable to recognize a takeover intention in real time, such that the vehicle can have sufficient time to gradually adjust its steering setting for vehicle control preparation [1][2].

There exist some approaches for driver activity recognition. One commonly used system is camera [3][4], as it is cost-efficient and easy to be installed. With advanced image processing techniques, camera can provide detailed information about the driver activities. However, traditional camera has several key limitations, such as Line-of-Sight (LoS) and brightness condition. Though the emergence of the infra-red camera and multi-camera systems has partially overcome the limitations, the camera system still has problems such as poor behavior in direct sunlight, obscurity caused by motions, and edge erosion [5][6]. Moreover, a recent study has shown the defect of state-of-the-art image processing algorithm known as "the elephant in the room" (see Section II) [7]. Besides camera, another possible solution is radar systems. They do not have these limitations and can recognize activities using radio wave reflection [8][9]. However, current radar systems on vehicles are usually used outside to detect large objects (e.g., other nearby vehicles). Fine-grained, short-range radar systems are still in the experimental stage and not yet available.

On the other hand, WiFi sensing has recently shown a great promise in human activity recognition [10][11][12]. Compared with radar systems, WiFi devices are easier to install and have a significantly lower cost. In addition, they do not have the limitations of camera. The WiFi-based solution is also non-intrusive, i.e., the device does not need to be attached to the human body. Recent studies show that WiFi-based human activity recognition system can achieve an accuracy over 90% for in-room activities [11][12], and its feasibility has been proven in a wider range of application scenarios, e.g., smoking detection [13] and people identification [14]. However, most WiFi activity recognition systems are designed only for large-scale activities in the room environment only.

With the development of WiFi-based V2V (Vehicle to Vehicle) network, WiFi is expected to soon replace Bluetooth for in-car interaction [15]. However, existing WiFi-based in-room activity recognition techniques cannot be directly applied to driver activity recognition, mainly due to three challenges. First of all, the range and duration of these activities can be much smaller than those of an in-room activity due to space limitation: The frequency shift caused by driver activities is much smaller, making it hard to capture in WiFi signals; Moreover, these activities usually last short ($<$1s). Thus, the recognition scheme should be sensitive in both time and frequency domains. Second, existing studies on activity recognition can recognize one activity only after it is completely finished. However, it could be too late to do such recognition for certain emergency driving scenarios. The activity should be recognized as early as possible, such that the vehicle can have sufficient time for the takeover preparation. Third, the

2575-8411/19/$31.00 ©2019 IEEE
DOI 10.1109/ICDCS.2019.00094

classification model should be adaptive and robust for different vehicles and drivers. For in-room activities, the classification model is often trained offline and fixed [11][16]. As a result, an offline trained model achieves low accuracy when the driver is not included in the training set or applied to a different environment (e.g., a new car). Thus, an online adaptation algorithm is required. Although recent studies [5][17][18] have started to recognize driver activities with WiFi signals, they only try to differentiate larger-scale movements (arm vs. head), focus mainly on non-emergency activities (no need for real-time recognition), and do not have online model adaptation.

In this paper, we propose WiDrive, a WiFi-based driver activity recognition system. It analyzes the moving speed of the driver's body parts, and extracts the direction information of small-scale activities with WiFi receivers. WiDrive is a real-time recognition system that can not only recognize activities in a short time, but also finish the recognition when the activity is still being performed. In addition, to be more robust, the classification parameters of WiDrive are designed to be updated adaptively for a new environment or a new user.

The main contributions of this work are as follows:

- We propose WiDrive, an in-car activity recognition system using commodity WiFi devices. WiDrive can recognize at least seven takeover-related small-scale in-car activities with an overall accuracy of 91.3%.
- WiDrive can recognize an ongoing activity within 60ms before it is finished. This leads to as much as 60% safety performance improvement compared to other solutions.
- WiDrive is designed to be adaptive and can update its classification model, based on Expectation Maximization (EM) algorithm, to fit different vehicles or drivers.

The rest of this paper is organized as follows. Section II discusses the related work. Section III presents the motivation by analyzing a typical driver takeover scenario. Section IV presents the design of WiDrive. Section V presents the evaluation results of WiDrive. Section VI concludes the paper.

II. Related Work

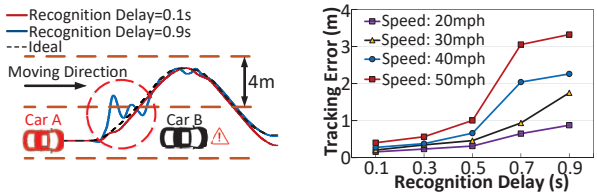
Camera is one commonly used sensor for driver activity recognition. The captured images can be processed to provide information about the traffic [4] and driving behaviors [3]. Infra-red and thermal cameras have overcome the limitations of the traditional camera (e.g., darkness and illumination [19][20]. However, a camera-based solution still relies on the placement of the devices, and has problems like poor behavior in direct sunlight, fragile lens subject to physical damage, and obscurity by motions [6]. Moreover, image processing algorithms commonly have the problem known as "the elephant in the room", i.e., the overlap of items in the image can cause recognition failures [7]. Thus, camera is not the best solution for the takeover detection scenario.

With the wide-spread WiFi networks, using WiFi receivers as a sensor has been a growing topic for human-computer interactions. Some initial studies focus on the Received Signal Strength Indicator (RSSI) for sensing and localization [10][21][22]. WiGest records the fluctuations caused by hand movements to recognize different gestures, but it can only work when the hand is within several centimeters around the receiver and the gestures are limited to hand push-pull combinations. Recent research studies have shown that the Channel State Information (CSI) can be used to recognize more fine-grained activities like finger movements than using RSSI [23][11][24][25][26]. Moreover, CSI can be directly extracted in many commercial Network Interface Card (NIC) like Intel 5300 [27]. For example, CARM proposes a CSI speed model and extracts the movement patterns of human body [11]. WiHear uses the reflection signals from lip movements to detect spoken words. [23]. Xi et al. propose to use CSI for counting the number of people in a crowd [26]. Several studies have leveraged CSI signal to detect small-scale human gestures. WiDraw is proposed to track the motion of people hands using Angle-of-Arrival estimation [25]. QGesture leverages CSI phase to track human hand movements [24]. However, these studies either have to use a lot of transmitters ($>$10) [25], or have to rely on special devices that are not commercially available [23][26]. Hence, they are not suitable for cost-efficient in-car recognition of small-scale driver activities.

Recently, some research has started to address in-car activity recognition with CSI [5][17][18]. Wibot is proposed to classify the driver's activities into head or arm movements [5]. However, the difference between head and arm movements is too coarse-grained for takeover recognition. WiFind aims to infer whether the driver is fatigued using Hilbert-Huang Transform [17], but is also coarse-grained as it can only recognize whether the activity is fatigue-related or not. ViHOT is proposed to track the head rotation for the driver with CSI phase difference [18] However, all the three aforementioned solutions focus only on the non-emergency activities and do not consider the vehicle's dynamics. In contrast, WiDrive aims to recognize the small-scale takeover activities, in real time, using CSI and takes the vehicle's dynamics into consideration to dynamically optimize the recognition performance.

III. Motivation

In this section, we motivate the designs of 1) real-time recognition and 2) online model adaptation for WiDrive. For real-time recognition, we analyze a real-world example as shown in Figure 1(a). Car A is on the autonomous driving mode and following another car (Car B). At some time point, Car B slows down due to some reasons (e.g., a mechanical failure). Noticing this situation, the driver of Car A decides to switch off autonomous driving and pass car B by reaching out for the steering wheel, and starts to manually take the pass maneuver. However, due to the difference between the vehicle's autonomous control law and the driver's assumption, there exist some risks in the first few seconds after the driver gets back the control of the vehicle [1][2]. We use a widely adopted virtual-reality driving simulation environment to simulate our motivation scenario [28]. In our simulation, a sinusoidal pulse is set to simulate the traveling path of Car A and the driver is told to track this path. In order to reduce the impact of driving styles on the experiment, we have recruited



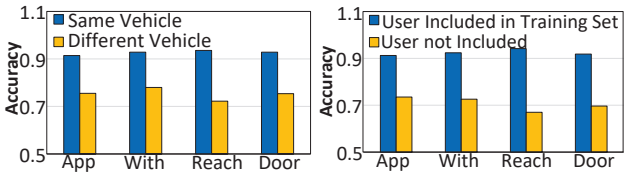
(a) Motivation Setup (b) Impact of Recognition Delay

Fig. 1: Motivation scenario setup and results. (a) The motivation scenario and the typical trajectories of two recognition delays. (b) The average tracking error (meters) related to recognition delay for three volunteers.

three volunteers to conduct the experiment independently. At the beginning of the experiment, the driver is asked to follow a straight line for a few seconds (autonomous driving with hands off the steering wheel), then grab the steering wheel and track the trajectory (manual driving). Note that the steering ratio is normally different for autonomous driving, so the takeover results in a steering change [1]. To better illustrate the importance of early recognition, we define "recognition delay" as the monitoring time to do the recognition for an ongoing in-car activity. For instance, if the recognition delay is 0.5s, it indicates that the driver's takeover activity is recognized 0.5 seconds after the driver starts to approach the steering wheel. Once the driver's takeover intention is recognized, the simulator adjusts the steering ratio gradually. Generally, an earlier recognition results in an earlier change of the steering ratio, so it can achieve a better stability and safety result.

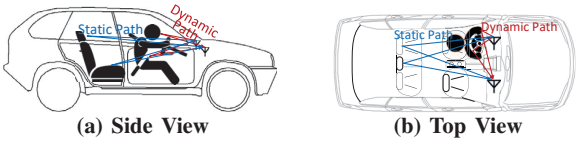
Figure 1(a) shows the typical traveling trajectories for 0.1s recognition delay (the red curve), 0.9s recognition delay (the blue curve), and ideal case (the black dash line). We can clearly see the oscillation made by the vehicle in the blue curve, which is caused by the steering setting change after the manual driving begins. This oscillation is dangerous because it not only causes vehicle instability, but also can interfere with other nearby vehicles. On the other hand, the trajectory of 0.1s recognition delay is almost as the same as the ideal trajectory, indicating that an early recognition and preparation for the takeover activity could improve the driver's performance of controlling the vehicle. Figure 1(b) shows the average tracking errors with recognition delay for the three volunteers. The tracking error stays small given a 0.3s recognition delay. As the recognition delay becomes larger, the tracking error increases rapidly with a normal speed (40mph), indicating that the vehicle does not have sufficient time to adapt its steering setting to the manual driving mode, so the driver cannot track the path well. Furthermore, the tracking error can be as large as 3.3m for the 0.9s recognition delay. Given that a typical lane's width is only 4m, this can causes a serious accident. This result shows that the takeover activity must be recognized in real time in order to leave sufficient time for the vehicle to do the takeover preparations. Otherwise, the driver may not be able to keep the vehicle in control, which can cause accidents.

To motivate the necessity of online model adaptation, we test the robustness of the existing WiFi recognition schemes using offline training scheme in the vehicle [29]. We test four different activities: Approach the driving wheel (*App*), withdraw from the driving wheel (*With*), reach for items



(a) Test with a Different Vehicle (b) Test with a Different Driver

Fig. 2: Robustness test for activity recognition system.



(a) Side View (b) Top View

Fig. 3: Multipath reflection of WiFi signals in the vehicle.

(*Reach*) and grab the door (*Door*). Two sets of experiments are conducted: First, we test these activities in a new vehicle that is different from the one used in model training; Then we test with a new user whose activities are not included in the training set. Figure 2 shows that the accuracy drops from over 90% to nearly 70% in a different vehicle, and drops even more for a new driver. With this accuracy for only four activities, the off-line trained model cannot be directly used in a different car or for a different driver. This result shows that an adaptation algorithm is needed to improve the recognition accuracy when the system is used in a new vehicle or with a new driver.

IV. Design of In-car Activity Recognition System

In this section, we first introduce the design overview of WiDrive, then introduce the details of each component.

A. Channel State Information and Activity Recognition

Modern WiFi devices usually contain multiple transmitting and receiving (Tx-Rx) antenna pairs. In order to perform rate adaptation or power optimization, the receiver has to continuously monitor the situation of the wireless channel and reports information back to the transmitter. Information about the wireless channel is called Channel State Information (CSI). For WiFi devices, CSI consists of channel frequency response of each subcarrier between each Tx-Rx pair. Taking multipath issues into consideration, the total channel system response should be the summation of signals (both amplitude and phase) coming from LoS paths and all the other reflection paths. By analyzing CSI, we could know how the reflection paths change: The changes on a certain reflection path lead to the change in the phase of the WiFi signal on the corresponding path [11]. If the changes are caused by human activities, these changes in CSI can be used to recognize different activities. As shown in Figure 3, the blue lines and red lines represent the static propagation paths and dynamic paths, respectively. For dynamic signals that are reflected by the moving human body, the speeds of body parts are recorded as Doppler shift in the corresponding CSI values.

B. Design Overview

We propose WiDrive, a system to recognize in-car activities based on CSI of WiFi signals. Figure 4 shows the framework of WiDrive. Generally, there are three major components in WiDrive: 1) Small-Scale Activity Feature Extraction, 2) Real-Time Recognition, and 3) Online Model Adaptation. Specifically, when the driver performs an activity (e.g., stretching

and grabbing the wheel), WiDrive can capture the sudden change of wireless signals, and record the following CSI values. Afterward, the energy distribution of recorded CSI values during the activity is extracted by the time-frequency analysis as feature vectors. These feature vectors serve as the input to the real-time recognition component, even if the whole activity is not finished yet. After the whole activity finishes and when the activity type can be confirmed by other devices, such as additional cameras, the model adaptation component adjusts the parameters of the real-time recognition component to improve the recognition accuracy.

The feature of an activity can be seen as the combination of the moving speeds of different human body parts (limbs, head, and torso) at different time points [11][14]. For example, activities involving only arms are much faster than those of the torso but with a less reflection area; Approaching the driving wheel causes a larger Doppler shift compared with swiping hands. Thus, the first component aims to extract the features of those activities using conjugate multiplication proposed in WiDance [29]. Then the signal is denoised by a low-pass filter and Principle Component Analysis (PCA) is applied to reduce the dimension of CSI signals. We use time-frequency analysis to extract activity features through Fourier transform.

While existing work focuses mainly on in-room large-scale human movement recognition (e.g., running, walking), a particular challenge for in-car activity recognition is that a driver can move only the upper body in a much smaller scale (than in-room activities) due to the limited space in a car cabin. Small-scale activities involve only limb motions and usually last short. Though several designs have been proposed to track small-scale activities like gestures and finger motions, either they can only work within a limited distance (<30cm) [30][25] (while the driver's hand can be anywhere near the driver's seat when the vehicle operates autonomously), or the recognition delay cannot meet the stringent time requirement of takeover (>0.7s) [24]. We propose to recognize small-scale activity through time-frequency analysis by finding the optimal time window and overlap ratio to fulfill the conflicting minimum frequency resolution requirement and visibility requirement of the time-varying signal. Then the activity is formulated as a sequence of feature vectors that are sent to the real-time recognition component for the activity recognition.

Another key novelty of WiDrive is its real-time recognition, which aims to recognize the activity quickly, even before the driver finishes it. However, there is a trade-off between the recognition accuracy (a longer trace results in a higher accuracy) and the vehicle control performance (earlier recognition allows better control preparation). We model this trade-off as a multi-objective optimization problem and find the best time point to finish the activity monitoring and start the recognition. Then, we fuse the feature vectors from two WiFi receivers and use Hidden Markov Models with Gaussian Mixture emissions Model (HMM-GMM) based classifier to recognize the activity.

The third component is online model adaptation. After an activity is recognized, we may be able to evaluate our recognition accuracy by comparing our result with the ground truth, which sometimes can be provided by other sensors, e.g., a camera. Note that the ground truth cannot be always obtained due to the limitations (e.g., direct sunlight), and WiDrive uses this component only when it can get the correct result from other sensors. If the accuracy is not sufficient, WiDrive adjusts the model using our adaptation algorithm based on the EM algorithm. The online model adaptation ensures a high accuracy, even when WiDrive is applied to different vehicles or different drivers.

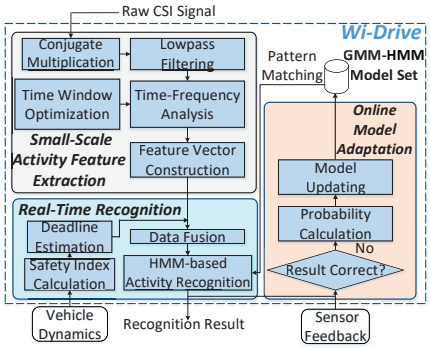


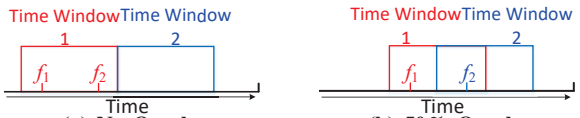
Fig. 4: Framework of WiDrive.

C. Small-scale Activity Recognition

A key challenge for in-car activity recognition is its small scale due to the space constraint in a car cabin. To explain the challenge in detail, we first introduce the existing WiFi-based recognition approaches and their major limitations.

CARM [11] and WiDance [29] propose to use CSI for in-room activity recognition with a good accuracy. However, these approaches cannot be used directly for in-car activities, because in-car activities do not have large torso motion due to the space limitation. The typical activities like approaching the driving wheel and shifting the gear only involve arm motions and their duration is usually small (less than 1s). As a result, the amplitude of wireless signals reflected from human body becomes smaller and so the WiFi signal change is harder to capture. On the other hand, WiDraw [25] can track the small motion of human hands and claims that its error is within 10%, but requires a large number of stationary WiFi transmitters (usually > 10). This assumption can be hard to meet due to limited space in the car cabin. QGesture [24] proposes to use the CSI phase to track the hand's traveling distance, but the phase change is sensitive to other turbulence like random body motions and surrounding changes.

We leverage time-frequency analysis to recognize small-scale activities with Short Time Fourier Transform (STFT), because STFT can preserve negative Doppler shifts while wavelet transform cannot. This is crucial in recognizing moving direction. However, small-scale activities have one issue using STFT: Under a certain sampling rate, a short time window used by STFT leads to few data points accumulated in each window length, so the frequency resolution is low. On the other hand, a large time window cannot capture the quick changing signal in the time domain, because it cannot reflect how exactly the signal changes in one single time window. Small-scale activities require a high frequency resolution, because the Doppler shift caused by the motion is limited,



(a) No Overlap **(b) 50% Overlap**

Fig. 5: An example of utilizing overlapping. We only know both frequency components f_1 and f_2 occur in time window 1 in (a). We can find f_1 is in the early half of window 1 while f_2 is in the overlapped part in (b).

i.e., it needs a long time window to distinguish how speed is distributed in the spectrum, while the short duration of in-car activities requires a short time window that can capture quick body speed changes during the activity.

To resolve the conflict, we utilize overlapping of two consecutive time windows to increase the visibility for time-varying signals while preserving frequency resolution. The redundant data introduced by overlapping allow us to have the visibility to see when the body speed changes in a very short time (less than a time window size). Figure 5 shows an example of how overlapping works. Frequency components f_1 and f_2 are in the same window shown in Figure 5(a). Thus, we cannot know when f_1 and f_2 appear within time window 1, because the whole time window will be transformed into frequency domain and we only know f_1 and f_2 are in time window 1. If we use an overlap ratio of 50% (shown in Figure 5(b)), we can know f_1 is only in time window 1, but f_2 appears in both windows. Thus, we can find out that f_1 is in the first half of time window 1 and f_2 is in the second half of time window 1 by comparing the frequencies in time windows 1 and 2, which doubles the visibility for time-varying signals while preserving frequency resolution. The more data points overlapped, the better visibility can be provided for time-varying signals.

However, overlapping introduces redundant data to be processed and increases recognition time overhead, which is an issue for a real-time system like WiDrive. To resolve the conflict between time-frequency resolution and the time overhead, we formulate an optimization problem with regard to time window length ΔT and overlap ratio $r_o (0 \leq r_o \leq 1)$ to minimize the time overhead of the system. The time overhead can be expressed as NL^2, where N is the number of feature vectors of an activity and L is the length of the feature vector. Specifically, $N = T_{total}/[\Delta T(1-r_o)]$ given that T_{total} is the duration of the activity; $L = \frac{F_{max}}{\Delta F} + \log_2(F_{cut} - F_{max})$, F_{cut} is the cutoff frequency for the low-pass filter, and F_{max} is the threshold frequency. Below F_{max} we equally divide the band of $[0, F_{max}]$ into several sub-bands whose bandwidth is exactly frequency resolution ΔF, while the bandwidth decreases exponentially for frequency band $[F_{max}, F_{cut}]$; Though N and L should be integers, we relax these constraints as a way to approximate the optimal solution to the problem. Mathematically, the problem is formulated as:

$$\begin{aligned} \min \quad & NL^2 \\ s.t. \quad & \frac{1}{F_{th}} \leq \Delta T \leq \frac{T_{th}}{1-r_o} \\ & 1 = \Delta F \Delta T \end{aligned} \tag{1}$$

where F_{th} and T_{th} are the lowest requirements for frequency resolution and maximum allowed time interval to distinguish speed change, respectively. Both of them are given according to the range and duration of the activity. In our implementation, F_{th} is chosen to be 4Hz because otherwise the Doppler shift will be overwhelmed for a small time window, and T_{th} is chosen to be 0.08s to capture the changes in one activity. The objective function is to minimize the computation overhead. This is a convex optimization problem and we can solve it using the Lagrange multiplier method by transforming the original problem into an unconstrained one by assigning auxiliary variable λ_i to each constraint shown in Equation (1) [31].

D. Real-Time Recognition

Within the allowed time length for driver takeover, three operations must be finished: 1) driver monitoring to collect a data trace for activity recognition, 2) recognition computation, and 3) vehicle control if the driver is recognized as intending to take over. Hence, a key challenge of real-time driver activity recognition is to decide how much time each operation can take while ensuring the total time is shorter than the allowed time length. As discussed before, for data collection, the longer it takes, the more accurate the recognition result will be, but leaves less time for computation and control.

1) Modeling accuracy-timeliness trade-off: In order to improve the real-time recognition performance, we hope to recognize an activity before it is finished. However, the decision of how long we should monitor an ongoing activity for the recognition is not trivial: For one activity, longer monitoring time of the activity yields better accuracy but leaves less preparation time for the vehicle to react. As shown in Section III, if the vehicle does not have sufficient time to prepare for the takeover action, it may lose its stability or increase the accident risk. Conversely, if the activity is recognized with insufficient data, the vehicle could receive false alarms and lower the user experience. Therefore, there is a trade-off between the recognition accuracy and the takeover control performance. We formulate it as a multi-objective optimization problem. Our objective vector $\mathbf{F}$ is defined as follows:

$$MAX\{\mathbf{F}(t_d, t_c) = [acc(t_d), safety(v, t_c)]\}, \tag{2}$$

where t_c is the duration for the vehicle to do the takeover preparation; t_d is the duration for WiDrive to collect data and perform real-time recognition; v is the velocity of the vehicle.

The first term $acc(t_d)$ is the recognition accuracy. We infer their relationship by collecting data on real vehicles and establish a statistical model offline based on the measured recognition result with logarithmic curve fitting, because it provides the minimum square error among all fitting options. The relation between accuracy and t_d is defined as:

$$acc(t_d) = \mathrm{In}(\alpha(t_d + b)) + c_0, \tag{3}$$

where α, b and c_0 are the parameters to be estimated in the curve fitting process. The second term $safety(v, t_c)$ is the safety index that reflects the effectiveness of the takeover control performance, and is related to the dynamics of the vehicle. The dynamics of the vehicle can be viewed as a

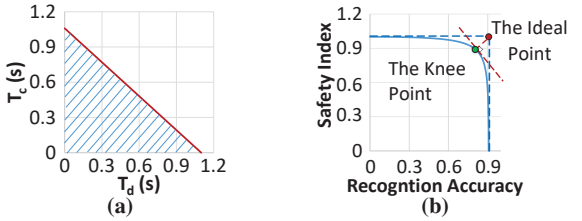


Fig. 6: An illustration of how to solve the multi-objective optimization problem. (a) The feasible zone. (b) The Pareto front (blue curve) and the knee point (green dot).

combination of lateral tracking error l_s and stability β in most cases on the road. Specifically, with a fixed fraction coefficient ($\mu = 0.6$) and a turning scenario, we can approximate l_s and β with t_c and v based on the general steering model [32]:

$$\begin{aligned} l_s(v, t_c) &= \frac{3.732}{1 + e^{-(0.4385 t_c + 0.1234 t_c v - 3.95))}} \\ \beta(v, t_c) &= 0.651 t_c^2 - 0.10 v t_c + 0.83 t_c + 0.38 v \end{aligned} \tag{4}$$

Note that different scenario may have different l_s and β estimations. In a real implementation, l_s and β can be estimated online by leveraging the path finding and tracking system [33][34]. After that, we compute $safety(v, t_c)$ related to l_s and β as follows in order to combine these two factors:

$$safety(v, t_c) = (\lambda e^{c_1 l_s(v,t_c)} + (1 - \lambda) e^{c_2 \beta(v,t_c)}) \tag{5}$$

where λ is the weight for the lateral tracking error term; c_1 and c_2 are chosen according to the road condition, vehicle dynamics and preference of vehicle manufacturers.

The constraint of our model comes from the fact that the total takeover time is limited. Thus, the summation of t_d and t_c should be smaller than the total takeover time T_{total}. Because T_{total} is usually small, we take the computation overhead t_{comp} into consideration and model $t_{comp} = kt_d + b$ as the complexity of our classification algorithm. Mathematically, the constraint of our multi-objective optimization function is formulated as follows:

$$t_c + t_d + t_{comp} \leq T_{total} \tag{6}$$

2) Optimal trade-off point calculation: To solve this multi-objective optimization problem, we propose to find the set of optimal solutions using the Pareto front theory. The Pareto front is a widely used tool to solve multi-objective optimization problem, and it is defined as the set of points where no objective can be improved without sacrificing at least one other objectives. Here we first find the Pareto front of our problem, then solve the problem by converting it to a single objective optimization problem. As shown in Figure 6(a), the feasible zone is shown as the shadow area below the red line. To find the Pareto front, we need to map every $[t_d, t_c]$ pair in the feasible zone to the objective function zone $[acc(t_d), safety(v, t_c)]$. Then the Pareto front can be found when the inequality constraint becomes an equality constraint (i.e., $t_c + t_d + t_{comp} = t_{total}$). On the Pareto front, all the points are equally optimal, because we can not improve $acc(t_d)$ without degrading $safety(v, t_c)$, or vice versa.

On the Pareto front, we choose the best trade-off point as the point whose distance between the ideal point (red dot in Figure 6(b)) and itself is minimum. This point is called the knee point and is widely used as the best trade-off point [35][36]. Figure 6(b) shows how to find the knee point: We draw a normal line to the Pareto front starting at the ideal point. Then the knee point can be found as the intersection of the normal line and the Pareto front (green dot in Figure 6(b)).

E. Online Adaptation of HMM Parameters

The widely-used training algorithm for Hidden Markov Models with Gaussian Mixture emissions Model (HMM-GMM) based classifier is the Baum-Welch algorithm, which can only work offline in a batch mode [37]. However, this offline model is not sufficient for different vehicles and drivers, or when a driver changes his posture pattern for an activity. Moreover, the HMM classifier cannot be re-trained when a new activity trace is collected due to computation overhead and storage limitations. Thus, we propose to have an adaptive design for the in-car activity recognition system: It should learn and update its classifier's model if the driver activity is not recognized correctly. When such activity sample is collected, the classifier changes its set of parameters to fit the new environment or a new driver. For an HMM classifier, the parameter set θ can be defined as $\theta = (\pi, A, w, \mu, \mathbf{\Sigma})$. π is the prior probability vector for the HMM model; A is the transition probabilities for the HMM model; w is the weight matrix for GMM; μ is the mean matrix for GMM; $\mathbf{\Sigma}$ is the covariance matrix for GMM. Generally, the HMM parameters (π and A) reflect the typical posture changing during an activity, and GMM parameters (w,μ,$\mathbf{\Sigma}$) represent how exactly the activity is performed and how velocities are distributed for each posture.

Specifically, we leverage the EM algorithm [38], a well-known algorithm for hidden state parameter estimation, to derive our online adaptation process. In our case, whenever a new sample O_T is obtained from our wireless receiver and the recognition result is not correct given the ground truth, we start a round of adaptation by assigning a weight W to O_T and calculating the lower bound of the likelihood function for activity estimation in the training set, namely $Q(\theta, \theta_i)$ as:

$$\begin{aligned} Q(\theta, \theta_i) = &\sum_{t=1}^{T-1} \sum_{S} p(S|O_t, \theta_i) \log p(O_t, S|\theta) \\ &+ W \sum_{S} p(S|O_T, \theta_i) \log p(O_T, S|\theta)) \end{aligned} \tag{7}$$

where θ_i is the parameter set estimator for HMM model before current round of adaptation. O_t is the t^{th} old sample; S is the hidden state set; T is the number of old samples. Let X_d^t be the random variable for the hidden state, where the d^{th} feature vector in the t^{th} sample belongs to (e.g., X_1^2 is the hidden state that the 1st feature vector in the 2nd activity sample in the training set belongs to), the update equation for the element in π using Lagrange multiplier can be derived as [31]:

$$\pi_i = \frac{\sum_{t=1}^{T-1} p(X_1^t = i|\theta_i)}{N - 1 + W} + \frac{W p(X_1^T = i|\theta_i)}{N - 1 + W} \tag{8}$$

where π_i in set $\theta = (\pi, A, w, \mu, \mathbf{\Sigma})$ represents the prior probability in hidden state S_i. In Equation (8), the first term only uses the old samples, so it can be calculated in the previous round of adaptation and be retrieved in this round of adaptation; The second term includes the new sample, and

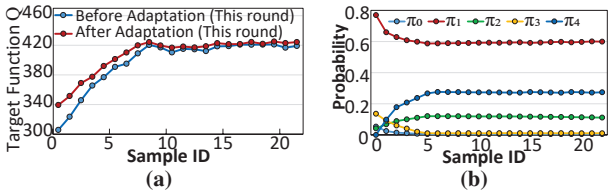


Fig. 7: An illustration of WiDrive's adaptation algorithm. (a) Accumulated probability of current training set. (b) Prior probability π_i for the classifier with five states.

is calculated using the first feature vector and the old estimator. In other words, with Equation (8), the probability π_i is updated with a weight averaging approach.

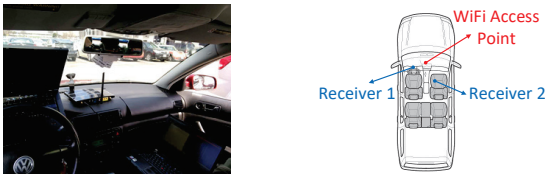
Figure 7(a) shows the target likelihood function $Q(\theta, \theta_i)$ of each round of adaptation when applying with a new driver. The gap between the red curve and blue curve is caused by the change of HMM-GMM parameters after each round of adaptation, which indicates the improvement on $Q(\theta, \theta_i)$. We can see that $Q(\theta, \theta_i)$ increases quickly for the first few samples, indicating that the mismatch between the original HMM-GMM model and the current activity sample is decreasing. After several rounds of adaptation, $Q(\theta, \theta_i)$ becomes almost static, indicating that the parameters have already been adapted to the new scenario. Figure 7(b) shows the change of state probability π_i, which gradually becomes stable after about five rounds of adaptation. Similarly, the formulas for the element a_{ij} in transition matrix A and GMM parameters (w, μ) are updated adaptively in the same way with π_i.

For every observed new sample which is misclassified, a round of adaptation is conducted by WiDrive. This process is preformed continuously and the calculation results are stored for possible next rounds. To avoid storing a large old sample set, we forget the sample when it is too old. The speed of forgetting old samples depends on the weight W: A larger weight indicates that WiDrive counts less on old samples, so it should store fewer old samples.

F. Discussion

1) False Alarm and Rollback: As an activity recognition system, despite a high accuracy, WiDrive still has false positives, e.g., the driver reaches his arms out but not aims for the driving wheel, which might raise the concern that WiDrive could lead to car instability with false alarms, as it may mistakenly disable the autonomous driving system. To that end, once WiDrive recognizes a takeover activity, it can still work with the vehicle control system to keep the vehicle stable while changing the steering setting gradually by using haptic feedback control [2][39]. WiDrive can also set a timer for the takeover activity: If there is no change of the vehicle dynamic for a certain time period upon a recognized takeover activity, WiDrive will consider this recognition as a false alarm and rollback the vehicle settings.

2) Impact of Surrounding Objects: One concern of in-car recognition system is that the environment is mobile: Objects outside the vehicle may impact on the robustness of the activity recognition. In order to test the impact of surrounding objects, we have collected activity traces on the road and calculated their spectrum. Results show that the impact of the outside objects is negligible compared to the driver's activity. WiDrive can still recognize the activity in a moving vehicle with an accuracy above 90%. The reason is that the major changes are caused by the driver's activity as the driver is much closer to the WiFi devices. The Doppler shift caused by surroundings is much weaker than that by the driver's activity.



(a) Devices inside the Car **(b) Top View Illustration**

Fig. 8: Device placement inside the car.

3) Multiple People in the Vehicle: One major concern of WiFi-based activity recognition systems is that it can work for single person only, since the reflection path change would be mixed if there are two or more subjects acting together in the same environment. For in-car scenario, the problem is simpler than that of the indoor scenario, because the position for each person in the vehicle is fixed. Thus, we could minimize other passengers' impact by placing the receiver close to the driver and far to the others. We have also conducted experiments with one passenger on the front seat, and let him perform activities together with the driver. The result shows that WiDrive can still recognize the driver's activity with an accuracy of 88.2%, regardless of what activity the passenger performs. It is our future work to further improve the accuracy of WiDrive in the scenarios with one or more passengers.

V. Evaluation

In this section, we conduct the evaluation of WiDrive. We first introduce the experiment setup. We then test WiDrive with other baselines, component by component (Sections V-B to V-E). At last, we test the overhead of WiDrive (Section V-F).

A. Experiment Setup

1) Testbed Setup: The testbed consists of one TP-Link Archer A7 as the transmitter and two ThinkPad T400 laptops with Intel 5300 wireless NICs (chosen to fit the modified driver [27]) as receivers to establish two Tx-Rx pairs in different directions. For each Tx-Rx pair, the transmitter has one antenna, and the receiver has three antennas. We generate packets using *ping* command at the transmission rate of 800*pkt/s*. CSI values are collected with modified network driver on a packet basis [27]. In order to align the start point of the transmission period, we write a script to synchronize the two laptops to begin CSI collection at nearly the same time. In a real implementation, the synchronization can be done by the central Electronic Control Unit (ECU) in the vehicle, and CSI could be collected using embedded systems instead of laptops. The placement of our devices is shown in Figure 8. The transmitter is placed in the middle of the dash broad. One receiver lays on the driving wheel, and the other is on the passenger seat so that the two receivers are orthogonal to each other.

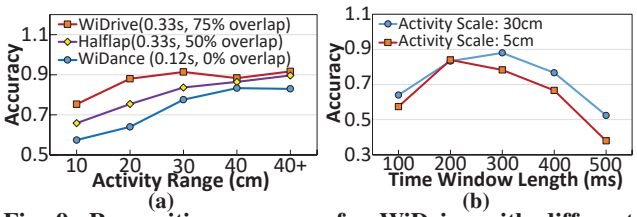


Fig. 9: Recognition accuracy for WiDrive with different time window sizes. (a) Comparison of different activity ranges for WiDrive and two baselines. (b) Detailed recognition accuracy with different time window size.

2) Candidate Activities Collection and Segmentation: For candidate activities, we categorize them as three types:

- **Driver's Intentional Takeover**. This type of activities indicate that the driver is trying to take the control back from autonomous driving mode and the vehicle should do the preparation for the takeover. Specifically, this type of activities include those that will change the vehicle's dynamic settings, e.g., approaching the steering wheel (*app*), switching off the autonomous driving mode (*switch*) and shifting the gear (*gear*).
- **Vehicle's Passive Takeover for Safety**. This type of activities indicate that the driver is not paying sufficient attention to the driving tasks and the vehicle should be noticed, so that the vehicle can enter the autonomous driving mode without any driver's input if necessary. This type of activities include distracted behaviors like eating/ drinking *(eat)*, fetching items (*fetch*), and withdrawing from the driving wheel (*with*).
- **Other activities relevant to safety**. This type of activities do not trigger the takeover action. However, they are still among the driver's most performed tasks [40]. Thus, we still list them as candidate activities for future usage such as operating door buttons (*door*). Note that some activities (e.g., turning the driving wheel during manual driving) are excluded from the candidate activities because they can be detected using internal sensors of the vehicle [5].

For each of these activities, we collect more than 80 traces per driver from eight volunteers (7 males and 1 female) in the real car. In order to automatically detect the start point of in-car activities, we propose to use the percentage of the near-zero frequency energy as an indicator to do the segmentation. For each time window, we calculate the percentage of the near-zero frequency (-5Hz∼5Hz) after Fourier transform and smooth it using median filter. Afterwards, we compare it with a pre-defined threshold. When the percentage is lower than the threshold, a possible activity begins and CSI value will be recorded until the percentage rises again.

B. In-car Activity Recognition Accuracy with Complete Traces

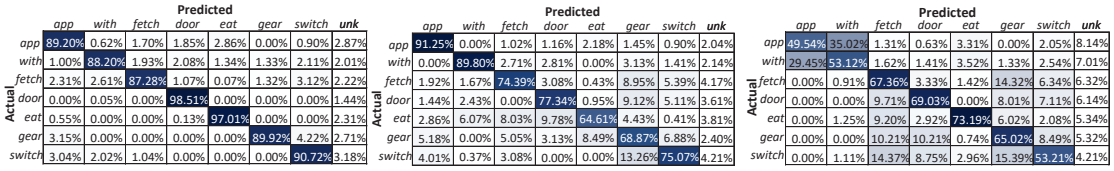
In this section, we disable WiDrive's real-time recognition and online adaptation to focus on evaluating its recognition accuracy of small-scale in-car activities with complete traces collected after the entire activity is finished. We first perform activities with different scale of activities (how far the human body moves) and compare the recognition accuracy of WiDrive ($\Delta T = 0.33$s, 75% overlap ratio) with two baselines: WiDance [29] ($\Delta T = 0.12$s, 0% overlap ratio) and Halflap ($\Delta T = 0.33$s, 50% overlap ratio), which is widely used in spectrum analysis and recognition applications [41].

WiDrive outperforms WiDance by 15% on average and Halflap by 10% for small-scale activities (<20cm). Generally, a larger-scale motion results in a higher accuracy because: 1) the frequency shift is higher because human body moves faster; 2) The reflection area becomes larger when the range of activities is larger. Figure 9(a) shows that WiDance performs the worse, because the Doppler shift caused by the activity will be overwhelmed in the low frequency band and cannot be captured; Halflap is better than WiDance because it considers using a larger window with overlap to increase frequency resolution, but the accuracy is still lower than WiDrive for small-scale activities. As the scale of the activities becomes larger, the difference between the baselines and WiDrive becomes smaller because the Doppler shift caused by the activities is stronger than that of small-scale activities and is easy to capture. Figure 9(b) shows the relation between the time window length and the recognition accuracy for activity scale 30cm and 5cm: The accuracy reaches its maximum when the time window is chosen between 200ms to 400ms, and it drops sharply when the time window is not in this range.

Then we test the recognition accuracy of WiDrive by doing a ten-fold cross validation and compare it with two baselines: CARM [11] and HMM-based WiDance [29]. CARM is a well-known recognition scheme but it cannot distinguish the moving direction of an activity; HMM-based WiDance uses WiDance's signal processing algorithm and uses HMM to recognize activities. Wibot is also an in-car activity recognition scheme that uses peak analysis to do recognition [5]. Compared with time-frequency analysis, the features (e.g. peak width, peak height) used in the peak analysis cannot reflect the time-varying characteristics for the CSI signal during the activity. Peak analysis also is well known to be prone to noises and interference, because there could be false peaks or valleys in the signal [42]. Thus, it is not selected for comparison.

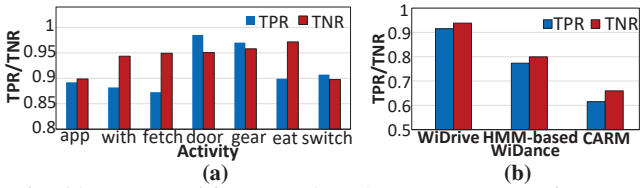
Figure 10 shows confusion matrices for WiDrive and baselines, whose elements on the diagonal show the accuracy for each activity type. WiDrive provides an overall accuracy with 91.3%, which is 14% higher than HMM-based WiDance and 30% higher than CARM. CARM and HMM-based WiDance only achieve an overall accuracy of 62.3% and 78.7%, respectively. CARM cannot identify *app* and *with* well because their difference lies mainly in the different activity direction. Moreover, CARM only leverages one Rx-Tx pair to recognize the activity so it performs the worst among the schemes. HMM-based WiDance can distinguish *app* and *with* with an accuracy over 89%, but the accuracy for other activities are below 80%, because reflection signals for an in-car activity are not as strong as that of an indoor activity. WiDrive manages to overcome this problem by choosing the optimal window length and overlap ratio, and finding the best way to formulate the feature vector and fuse the CSI data.

Besides recognition accuracy, Figure 11(a) shows the True



(a) WiDrive (b) HMM-based WiDance (c) CARM

Fig. 10: The confusion matrices of WiDrive and two baselines.



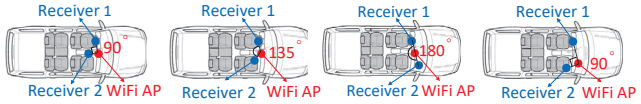
(a) (b)

Fig. 11: True Positive Rate (TPR) and True Negative Rate (TNR) Comparison. (a) Each activity using WiDrive. (b) Average of all activities using different approaches.

Positive Rate (TPR) and True Negative Rate (TNR) of WiDrive for each activity. The overall TPR and TNR of WiDrive is 91.2% and 93.7%, respectively, which indicates that WiDrive can achieve a good trade-off between the missed detection and the false alarm. WiDrive outperforms HMM-based WiDance by 15.4% and 14.9% in terms of TPR and TNR, and outperforms CARM by 30.4% and 27.5% in terms of TPR and TNR as shown in Figure 11(b).

C. Different Placements of WiFi Devices

WiDrive uses Doppler shift to recognize different human activities. Thus, the recognition accuracy could be affected by different placements of the devices, because different placements of the devices could result in different traveling paths of wireless signal reflected from human body. We test four different placement settings shown in Figure 12. Figure 12(a) is the default placement setting of WiDrive. In order to see how the angle for the two receivers affects the recognition accuracy, we change the angle to $135°$ and $180°$ between the two receivers and the router shown in Figure 12(b) and Figure 12(c), respectively. In Figure 12(d), we keep the two receivers to be orthogonal, but each of them has a larger distance to the router on the dashboard. Though there could be an infinite number of device placement setups, we choose these placements as they can minimize the impact of the other passengers. In the real system, WiDrive could be embedded into the car body so that the most possible positions are all included in Figure 12.



(a) (b) (c) (d)

Fig. 12: Top view of different device placement locations.

Figure 13 shows the accuracy of each activity for the four placements. The overall accuracy does not change much for different device placements, indicating that WiDrive is robust and can be set in different locations on the car body. However, if we examine the accuracy for each activity, we can see that compared with placement (a), the accuracy of some activities (*door, gear, eat, and switch*) decrease by 6.30% and 5.92% for

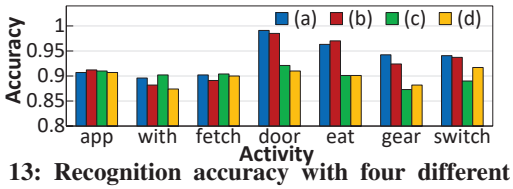
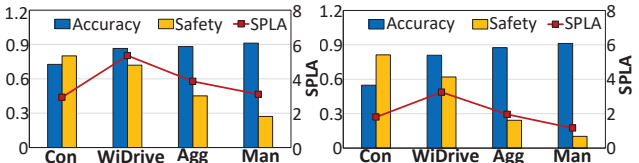


Fig. 13: Recognition accuracy with four different placements (a,b,c,d) shown in Figure 12.



(a) Low Speed Case (40mph) (b) High Speed Case (65mph)

Fig. 14: Comparison of WiDrive with others baselines at low or high driving speed.

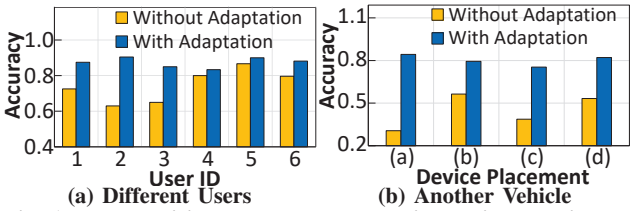
placement (c) and (d), respectively, while the accuracy for the others does not change much. The main reason for the drop is that the distance between the driver and the receiver on the side will be larger compared with placement (a), and these activities (*door, gear, eat and switch*) rely heavily on the Doppler shift on this receiver to be recognized: The reflected signal is weaker for placements (c) and (d), so the Doppler shift caused by the human activity will be harder to capture by the receiver due to the interference. On the other hand, the other activities (*app, with, and fetch*) have more distinctive features for the receiver on the steering wheel and their dependency on the side receiver is much smaller, so the accuracy of these activities does not change as the placement changes.

D. Real-Time Recognition

WiDrive needs to finish activity monitoring, recognition computation and vehicle control action in the total duration of an activity. Here we evaluate the real-time performance of WiDrive comparing with three baselines: 1) Conservative (Con), which is designed to prioritize the control performance and guarantees that the lateral tracking error will not be lager than the half width of a lane (the car shall not travel to other undesired lanes); 2) Aggressive (Agg), where t_d is chosen with the maximum instability index such that the vehicle can exactly maintain the stability; 3) Manual (Man): no takeover preparation is taken by the vehicle. In order to evaluate the impact on vehicle safety, we uses a metric called Safety Per Lost Accuracy (SPLA), which is calculated as:

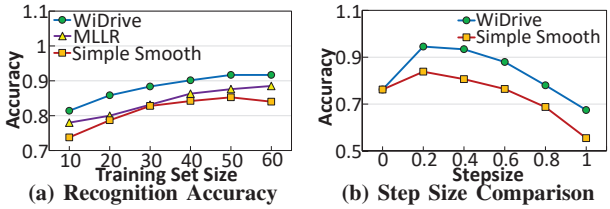
$$SPLA = safety(v, t_c)/[1 - acc(t_d)] \quad (9)$$

SPLA represents the trade-off efficiency in the multi-optimization model. A higher SPLA score indicates a better accuracy-safety trade-off. The test scenario remains the same as in the motivation (Section III). Figure 14 shows that



(a) Different Users (b) Another Vehicle

Fig. 15: Recognition accuracy comparison with or without adaptation among different users and in a different car.



(a) Recognition Accuracy (b) Step Size Comparison

Fig. 16: Evaluation for adaptation algorithm of WiDrive.

WiDrive has the highest SPLA score among all the solutions: 5.35 and 3.35 for the low-speed case and high-speed case, respectively. Conservative has the best performance on the safety index but it can only achieve 54.5% recognition accuracy. Such a low accuracy will generate false alarms and missed detection frequently, so the user experience is lowered. On the other side, the accuracy of Aggressive is only 2.4% higher than that of WiDrive, but its safety index is 28.1% lower. Thus, Aggressive might cause collisions between vehicles on the road because it totally focuses on the vehicle's stability and neglects the tracking error. At last, Manual has the lowest safety score, indicating that without system assistance, it will be too late to finish the takeover control. WiDrive achieves the best trade-off of accuracy and safety by sacrificing only 8.2% safety index for 25.3% accuracy improvement.

E. Online Model Adaptation

Here we first test WiDrive with six new users whose activities are not included in the training set. Then we test WiDrive in a different car other than the one mentioned in Section V.A to compare the recognition accuracy with and without adaptation for each user and in a different vehicle, respectively. Then we compare WiDrive with other well-designed baselines to show the improvements of our adaptation algorithm.

Figure 15(a) shows the effectiveness of using the online adaptation algorithm. For all the users, the accuracy is above 83% with adaptation. For users 2 and 3, the offline-trained HMM model can only achieve 63% and 65% accuracy, because the in-car space and reflection conditions are different from in-room ones: A small variation of the activity can cause a large difference in the reflected CSI signals. Fortunately, the accuracy rises to 90% and 85% for online adaptation given a training set of only 25 samples. Figure 15(b) shows the improvements with the adaptation algorithm. Due to the different environment, the accuracy of offline-trained model can drop from 90.2% to 38.4%, especially when the placement is different from what is shown in Section V.A. This accuracy indicates that the off-line trained model cannot be directly applied into another vehicle. Fortunately, with the adaptation algorithm, the accuracy increases by 35.6% with a training set size of 40 samples. With the adaptation algorithm, WiDrive could be directly applied in a new environment or to a new driver without re-training the whole system.

Then we compare WiDrive with two baselines: 1) Simple Smooth, which uses the exponential smooth function to get the new HMM-GMM parameters. 2) Maximum Likelihood Linear Regression (MLLR), which is a classic adaptation algorithm in speech signal processing [43]. Figure 16(a) shows that WiDrive's accuracy is 91.2% given a training set of 60 samples, whereas MLLR and Simple Smooth achieve an accuracy of 88.3% and 85.6%, respectively. Simple Smooth performs the worst because it does not guarantee the convergence to the new optimal solution. With the training set size increasing from 50 to 60, the accuracy for this scheme even drops. On the other side, WiDrive outperforms MLLR because WiDrive considers adapting HMM parameters, whereas MLLR only changes GMM parameters. MLLR can be used for the speech signals that have well-trained acoustical properties, but does not work well when it comes to activity recognition. WiDrive achieves the highest accuracy and outperforms MLLR by 6% on average. Figure 16(b) shows that the optimal step size for WiDrive is 0.2 to 0.4: A small step size causes slow adaptation given a certain training set, while a large step will downgrades the performance by forgetting the previous data too fast.

TABLE I: System overhead of each step.

Procedure	Processing Time (ms)
Activity Segmentation	0.87
Signal Pre-Processing	12.47
Time-Frequency analysis	24.43
Real-Time Recognition	13.45
Adaptive Learning	14.32

F. System Overhead

Here we test the time overhead of WiDrive with complete activity traces to make sure it can process CSI and recognize an in-car activity in real time. Table I shows that the overhead of WiDrive is modest: The total time overhead of WiDrive is smaller than 60ms, excluding the adaptive learning step. The most time consuming step is the time-frequency analysis, which finishes within 30ms for a sampling rate of 800Hz. This time overhead can be even smaller when a lower sampling rate is adopted. If WiDrive is implemented in DSP or other embedded systems, the time overhead can be further decreased.

VI. Conclusion

In this paper, we have proposed WiDrive, an in-car driver activity recognition system based on CSI changes of WiFi signals. Different from existing in-car driver monitoring systems like camera, our system uses commodity WiFi devices and provides non-intrusive recognition regardless of light conditions or obscurity. WiDrive consists of three major components: an activity feature extraction algorithm to extract information of small-scale activities, an HMM-based real-time recognition system, and an online model adaptation algorithm. WiDrive is evaluated in real car scenarios. The evaluation results show that WiDrive can achieve a recognition accuracy of 91.3% or higher and substantially improve the takeover safety.

REFERENCES

[1] H. E. B. Russell, L. K. Harbott, I. Nisky, S. Pan, A. M. Okamura, and J. C. Gerdes, "Motor learning affects car-to-driver handover in automated vehicles," *Science Robotics*, vol. 6, no. 6, pp. 6–14, 2016.

[2] C. Guo, C. Sentouh, J.-C. Popieul, and J.-B. Haue, "Predictive shared steering control for driver override in automated driving: A simulator study," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 61, no. 1, pp. 326–336, 2018.

[3] A. Jain *et al.*, "Brain4cars: Car that knows before you do via sensory-fusion deep learning architecture," *arXiv preprint arXiv:1601.00740*, 2016.

[4] B. Morris, A. Doshi, and M. Trivedi, "Lane change intent prediction for driver assistance: On-road design and evaluation," in *Proceedings of the 22nd IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[5] M. Raja, V. Ghaderi, and S. Sigg, "WiBot! in-vehicle behaviour and gesture recognition using wireless network edge," in *Proceedings of the 38th International Conference on Distributed Computing Systems (ICDCS)*, 2018.

[6] A. Kadambi, A. Bhandari, and R. Raskar, "3D depth cameras in vision: Benefits and limitations of the hardware," in *Computer Vision and Machine Learning with RGB-D Sensors*, 2014, pp. 3–26.

[7] A. Rosenfeld, R. Zemel, and J. K. Tsotsos, "The Elephant in the Room," *arXiv preprint arXiv:1808.03305*, 2018.

[8] P. Molchanov, S. Gupta, K. Kim, and K. Pulli, "Short-range FMCW monopulse radar for hand-gesture sensing," in *Proceedings of 2015 IEEE Radar Conference (RadarCon)*, 2015.

[9] J. Lien, N. Gillian *et al.*, "Soli: Ubiquitous Gesture Sensing with Millimeter Wave Radar," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 4, p. 142, 2016.

[10] H. Abdelnasser, M. Youssef, and K. A. Harras, "WiGest: A ubiquitous WiFi-based gesture recognition system," in *Proceedings of the 34th IEEE Conference on Computer Communications (INFOCOM)*, 2015.

[11] W. Wang, A. X. Liu, M. Shahzad, K. Ling, and S. Lu, "Understanding and modeling of WiFi signal based human activity recognition," in *Proceedings of the 21st annual international conference on mobile computing and networking (MobiCom)*, 2015.

[12] Y. Wang, J. Liu, Y. Chen, M. Gruteser, J. Yang, and H. Liu, "E-eyes: device-free location-oriented activity identification using fine-grained WiFi signatures," in *Proceedings of the 20th annual international conference on Mobile computing and networking (MobiCom)*, 2014.

[13] X. Zheng, J. Wang, L. Shangguan, Z. Zhou, and Y. Liu, "Smokey: Ubiquitous smoking detection with commercial WiFi infrastructures," in *Proceedings of the 35th IEEE International Conference on Computer Communications (INFOCOM)*, 2016.

[14] Y. Zeng, P. H. Pathak, and P. Mohapatra, "WiWho: WiFi-based person identification in smart spaces," in *Proceedings of the 15th International Conference on Information Processing in Sensor Networks (IPSN)*, 2016.

[15] J. L. Wilson, "Automotive WiFi availability in dynamic urban canyon environments," *Navigation: Journal of The Institute of Navigation*, vol. 63, no. 2, pp. 161–172, 2016.

[16] K. Ali, A. X. Liu *et al.*, "Keystroke recognition using WiFi signals," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2015, pp. 90–102.

[17] W. Jia, H. Peng, N. Ruan, Z. Tang, and W. Zhao, "WiFind: Driver fatigue detection with fine-grained Wi-Fi signal features," *IEEE Transactions on Big Data*, 2018.

[18] X. Xie, K. G. Shin, H. Yousefi, and S. He, "Wireless CSI-based head tracking in the driver seat," in *Proceedings of the 14th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, 2018.

[19] S. Gaglio *et al.*, "Human activity recognition process using 3-D posture data," *IEEE Transactions on Human-Machine Systems*, vol. 45, no. 5, pp. 586–597, 2015.

[20] R. Gade and T. B. Moeslund, "Thermal cameras and applications: a survey," *Machine vision and applications*, vol. 25, no. 1, pp. 245–262, 2014.

[21] Y. Gu, F. Ren, and J. Li, "Paws: Passive human activity recognition based on WiFi ambient signals," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 796–805, 2016.

[22] T. Gu, Z. Wu, X. Tao, H. K. Pung, and J. Lu, "epSICAR: An emerging patterns based approach to sequential, interleaved and concurrent activity recognition," in *Proceedings of the 7th IEEE International Conference on Pervasive Computing and Communications (PerCom)*, 2009.

[23] G. Wang, Y. Zou, Z. Zhou, K. Wu, and L. M. Ni, "We can hear you with Wi-Fi!" *IEEE Transactions on Mobile Computing*, vol. 15, no. 11, pp. 2907–2920, 2016.

[24] N. Yu, W. Wang, A. X. Liu, and L. Kong, "QGesture: Quantifying gesture distance and direction with WiFi signals," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 2, no. 1, p. 51, 2018.

[25] L. Sun, S. Sen, D. Koutsonikolas, and K.-H. Kim, "WiDraw: Enabling hands-free drawing in the air on commodity WiFi devices," in *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom)*, 2015.

[26] W. Xi, J. Zhao, X.-Y. Li, K. Zhao, S. Tang, X. Liu, and Z. Jiang, "Electronic frog eye: Counting crowd using WiFi," in *Proceedings of the 33th IEEE International Conference on Computer Communications (INFOCOM)*, 2014, pp. 361–369.

[27] D. Halperin, W. Hu, A. Sheth, and D. Wetherall, "Tool release: Gathering 802.11 n traces with channel state information," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 1, pp. 53–53, 2011.

[28] "Christie Digital Holostage Mini CAVE." [Online]. Available: https://www.christiedigital.com/SupportDocs/Anonymous/Christie-CAVE-HoloStage-Mini-Brochure.pdf

[29] K. Qian, C. Wu, Z. Zhou, Y. Zheng, Z. Yang, and Y. Liu, "Inferring motion direction using commodity Wi-Fi for interactive exergames," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI)*, 2017.

[30] S. Tan and J. Yang, "Wifinger: leveraging commodity wifi for fine-grained finger gesture recognition," in *Proceedings of the 17th ACM international symposium on mobile ad hoc networking and computing (MobiHoc)*, 2016.

[31] S. Boyd *et al.*, *Convex optimization*. Cambridge university press, 2004.

[32] W. Wang, J. Xi, and J. Wang, "Human-centered feed-forward control of a vehicle steering system based on a driver's steering model," in *Proceeding of 2015 American Control Conference (ACC)*, 2015.

[33] Z. Shuai, H. Zhang, J. Wang, J. Li, and M. Ouyang, "Combined AFS and DYC control of four-wheel-independent-drive electric vehicles over CAN network with time-varying delays," *IEEE Transactions on Vehicular Technology*, vol. 63, no. 2, pp. 591–602, 2014.

[34] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, "Evolutionary algorithm based offline/online path planner for UAV navigation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003.

[35] L. Rachmawati and D. Srinivasan, "Multiobjective evolutionary algorithm with controllable focus on the knees of the Pareto front," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 810–824, 2009.

[36] J. Branke, K. Deb, H. Dierolf, and M. Osswald, "Finding knees in multi-objective optimization," in *Proceedings of the 4th International conference on parallel problem solving from nature (PPSN)*, 2004.

[37] L. R. Welch, "Hidden markov models and the baum-welch algorithm," *IEEE Information Theory Society Newsletter*, vol. 53, no. 4, pp. 10–13, 2003.

[38] A. P. Dempster *et al.*, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 39, no. 1, pp. 1–22, 1977.

[39] C. Lv, H. Wang, D. Cao, Y. Zhao, M. Sullman, D. J. Auger, J. Brighton, R. Matthias, L. Skrypchuk, and A. Mouzakitis, "A novel control framework of haptic take-over system for automated vehicles," in *Proceedings of the 29th IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[40] B. Pfleging and A. Schmidt, "(non-) driving-related activities in the car: Defining driver activities for manual and automated driving," in *Workshop on experiencing autonomous vehicles: Crossing the boundaries between a drive and a ride at CHI*, 2015.

[41] L. Rabiner *et al.*, *Fundamentals of Speech Recognition*. PTR Prentice Hall Englewood Cliffs, 1993, vol. 14.

[42] F. Scholkmann, J. Boss, and M. Wolf, "An efficient algorithm for automatic peak detection in noisy periodic and quasi-periodic signals," *Algorithms*, vol. 5, no. 4, pp. 588–603, 2012.

[43] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.