CarOSense: Car Occupancy Sensing with the Ultra-Wideband Keyless Infrastructure

YONGSEN MA*†, Bosch Research, Sunnyvale, CA, USA
YUNZE ZENG*, Bosch Research, Sunnyvale, CA, USA
VIVEK JAIN, Bosch Research, Sunnyvale, CA, USA

Ultra-Wideband (UWB) is a popular technology to provide high accuracy localization, asset tracking and access control applications. Due to the accurate ranging feature and robustness to relay attacks, car manufacturers are upgrading the keyless entry infrastructure to UWB. As car occupancy monitoring is an essential step to support regulatory requirements and provide customized user experience, we build *CarOSense* to explore the possibility of reusing UWB keyless infrastructure as an orthogonal sensing modality to detect per-seat car occupancy. *CarOSense* uses a novel deep learning model, MaskMIMO, to learn spatial/time features by 2D convolutions and per-seat attentions by a multi-task mask. We collect UWB data from 10 car locations with up to 16 occupancy states in each location. We implement *CarOSense* as a cross-platform demo and evaluate it in 15 different scenarios, including leave-one-out test of unknown car locations and stress test of unseen scenarios. Results show that the average accuracy is 94.6% for leave-one-out test and 87.0% for stress test. *CarOSense* is robust in a large set of untrained scenarios with the model trained on a small set of training data. We also benchmark the computation cost and demonstrate that *CarOSense* is lightweight and can run smoothly in real-time on embedded devices.

CCS Concepts: • **Human-centered computing → Ubiquitous computing**; • **Computer systems organization → Embedded and Cyber-physical systems**.

Additional Key Words and Phrases: Smart Automobiles, Ultra-Wideband (UWB), Wireless Systems, Wireless Sensing

ACM Reference Format:
Yongsen Ma, Yunze Zeng, and Vivek Jain. 2020. CarOSense: Car Occupancy Sensing with the Ultra-Wideband Keyless Infrastructure. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 3, Article 91 (September 2020), 28 pages. https://doi.org/10.1145/3411820

1 INTRODUCTION

In recent years, Ultra-Wideband (UWB) has gained traction as an important wireless technology to support high accuracy localization and tracking applications [13]. It is now available in smartphones [5] and is also used extensively for asset tracking applications in industrial plants [19]. Compared to other solutions based on Wi-Fi or Bluetooth, UWB has much higher bandwidth resulting in much better resolution and spatial awareness. Apart from being more accurate, UWB also provides much better immunity against relay attacks as UWB has better timing resolution (inverse of high bandwidth) and the IEEE 802.15.4-2015 UWB standard explicitly incorporates

*Both authors contributed equally to the paper.
†Yongsen Ma completed this work during his internship at Bosch Research, Sunnyvale, CA, USA.

Authors' addresses: Yongsen Ma, fixed-term.yongsen.ma@us.bosch.com, Bosch Research, Sunnyvale, CA, USA; Yunze Zeng, yunze.zeng@us.bosch.com, Bosch Research, Sunnyvale, CA, USA; Vivek Jain, vivek.jain@us.bosch.com, Bosch Research, Sunnyvale, CA, USA.

2474-9567/2020/9-ART91 $15.00
https://doi.org/10.1145/3411820

timing information [30]. Ranging-based localization is mostly used for access control applications such as for cars, garage doors, asset tracking, etc., which requires operation from distance. Consequently, for the past few years IEEE 802.15.4z Enhanced Impulse Radio Task Group is developing more secure and accurate ranging methods for UWB-based access control [31]. One major application that is looking for disruption is automotive keyless access where current solutions suffer from relay attacks [18, 22, 29, 60]. UWB can be used as a keyless infrastructure in cars for passive entry [6, 33, 43], vehicle authorization [36], and access control [42]. In this paper, we explore using UWB keyless infrastructure for occupancy/presence sensing in vehicles with per-seat granularity.

Car occupancy has always been an active area of interest for automotive makers from both supporting regulatory requirements and providing interesting user experience. At the base level, every car should detect occupancy in front seats and accordingly provide Seat Belt Reminder (SBR). This basic regulation, which was first enacted law in 1970, is changing as the focus is shifting to safety of all passengers and not only front seat ones. As per studies, unrestrained rear seat occupants are nearly eight times as likely to sustain a serious injury in a crash as restrained rear seat occupants and twice likely to die if they are unbelted [1, 8]. In 2017 alone, 1341 fatally injured rear seat occupants were aged 12 years and older with 60% of them not wearing seat belts [41]. In 2018, overall 803 fatalities were reported in vehicle crashes in the USA for unbelted rear seat occupants who were 8 years old or higher [8]. In general, seat belt usage has been reported 10-15% lowers compared to front seats and the number can be worse for different regions, for example, states that do not enforce rear seat belt use in the USA [8]. To address this problem, recently New Car Assessment Program (NCAP) all over the world are providing a given vehicle additional safety ratings for supporting SBR in rear seats [51, 53] as incentives. These incentives are also considered as regulatory requirements in making and hence automakers are developing robust and cost efficient rear seat occupancy detection and seat belt reminder solutions. Apart from basic seat belt reminder function, such a system can also provide additional safety features such as airbag control and enhanced user experience with better climate and audio controls. Car occupancy information is also an important component for effective shared autonomy, such as human sensing, shared perception-control, and deep personalization, for Human-Centered Autonomous Vehicle (HCAV) Systems [23].

There are various sensing modalities that can be used for car occupancy sensing, with each having its own advantages and disadvantages: camera-based solutions have privacy issues [61], weight sensors are not accurate enough to identify between animate/inanimate objects, ultrasound techniques are not pet friendly when frequencies used are less than 80kHz [14, 70], and RF Radar requires high bandwidth and multiple antennas which translate to high cost and power consumption [10, 46]. Moreover, all of them incur additional dedicated hardware and installation cost of wire/cable harnesses, as they are not available as standard offerings in a car. Of all these technologies, weight sensor (essentially a pressure sensor) is a de facto solution for occupancy sensing and classification [28]. Even though weight sensor based solutions are available for years but they are not widely adopted for rear seats primarily due to additional cost, immature rear air bag deployment techniques (not as effective as front airbags) and seat reconfigurability constraints (e.g. wiring wear tear which may happen when seat is removed by the user). Therefore providing a wire-free solution for occupancy sensing can potentially alleviate limitations of current solutions. So, we ask a basic question: can we use potentially existing system in the car to address per-seat occupancy sensing? As mentioned earlier, the keyless system of upcoming vehicles are going to have UWB technology. UWB provides rich channel information in the form of Channel Impulse Response (CIR) at the receiving nodes and is harmless to humans and animals due to extremely low power [58]. Wireless channels are generally sensitive to human movements within the channel and it is this hypothesis that we envision can be used for detecting per-seat occupancy within the vehicle. A typical keyless solution may have up to 8 nodes depending on make/model and the preference of Original Equipment Manufacturers (OEMs), where up to 4 nodes may be external. So to test our hypothesis, we assume up to 8 nodes in the car with different locations as a superset of possible node locations [7, 24, 49].

Although UWB has high spatial resolution, there are still challenges for UWB-based occupancy sensing to be accurate, robust, and practical in real-world scenarios with per-seat granularity. Recently, machine learning algorithms are widely used for wireless sensing applications [48]. Traditional machine learning algorithms, such as k Nearest Neighbor (kNN), Support Vector Machine (SVM), and Gradient Boosting Decision Trees (GBDT), need a lot of human efforts to design and select the right features. More importantly, these algorithms only work when they are trained and tested with UWB data from the same car location and the same car state. Deep learning models, such as Convolutional Neural Networks (CNNs), do not need extensive feature engineering and have higher accuracy than kNN, SVM, and GBDT for unknown car locations. But standard CNN models still do not work for new scenarios that are not seen during training, for example, the driver seat is moved for test data. The reason is that standard CNNs are usually designed for computer vision tasks, while UWB data are different from digital images [40, 58]. Therefore, it is necessary to develop new deep learning models that are specially designed for UWB data. We also would like to point out that automotive indoor environment is small with rich multi-path and very variable with moving seat positions, animate/inanimate objects with different reflection properties at different locations (on seat, car floor, or in trunk), etc. It is impractical to train the model with all possible cases. However, a developed model is expected to work reasonably good for all possible scenarios.

We propose a new deep learning design, *CarOSense*, for per-seat car occupancy sensing with the UWB keyless infrastructure. *CarOSense* has 8 UWB nodes in the car with one node broadcasting UWB packets and other nodes collecting CIR measurements. Raw CIRs are processed and fed to a multi-input multi-output CNN with multi-task mask, or MaskMIMO, for per-seat occupancy classification. MaskMIMO is accurate and robust for unknown car locations and unseen scenarios by learning spatial/time features from 2D convolutions and per-seat occupancy attentions from the multi-task mask. It requires low effort for signal processing, feature engineering, and model training. We implement *CarOSense* as a demo that can run in real-time on different platforms including personal computers and embedded platforms such as Google Coral [26] and Raspberry Pi [21].

We collect UWB CIR data from 10 different car locations with up to 16 occupancy states, and evaluate *CarOSense* under 15 different scenarios from perspectives of accuracy, robustness, and computation cost. Accuracy and robustness are evaluated by leave-one-out test wherein the car location of test data is not seen during training and stress test of different unseen scenarios such as car seats moved and kids in back seats. The average accuracy of *CarOSense* is 94.6% using 8 UWB nodes (77.3% using 4 nodes) for leave-one-out test and 87.0% using 8 nodes (69.4% using 4 nodes) for stress test. Computation cost is evaluated by running the real-time demo on different platforms. Evaluation results demonstrate that *CarOSense* has low computation cost and can run smoothly in real-time on embedded platforms. The time consumption of CIR processing and model inference is 128 milliseconds for Google Coral and 86 milliseconds for Raspberry Pi. In summary, we make the following contributions:

First-of-its-kind: *CarOSense* is the first system that leverages UWB keyless infrastructure in automobiles to demonstrate occupancy/presence detection application with per-seat granularity.

Accurate: *CarOSense* has high accuracy for different scenarios: 94.6% with 8 UWB nodes and 77.3% with 4 UWB nodes for leave-one-out test of unknown car locations.

Robust: *CarOSense* is robust in a large set of untrained scenarios with the model trained on a small set of training data. For stress test of unknown scenarios, the accuracy of *CarOSense* is 87.0% with 8 UWB nodes and 69.4% with 4 UWB nodes.

Cost Efficient: *CarOSense* leverages existing keyless infrastructure and in the best case does not require any additional UWB nodes in the car. Moreover, it does not have location specific requirements and hence can utilize the existing installed UWB sensors. *CarOSense* requires low training effort and has low computation cost thus making it practical to run in real-time on embedded devices with constrained resources.

The rest of the paper is organized as follows. Section 2 gives background of automotive keyless entry infrastructure and UWB basics. Section 3 first presents feasibility study of car occupancy sensing with UWB and

standard machine learning algorithms and then presents *CarOSense* including CIR processing and classification algorithms. Section 4 shows experiment setup, performance scores of different testing scenarios, and real-time computation cost on different platforms. Section 5 presents related works, and Section 6 concludes the paper.

2 BACKGROUND

To better understand the motivation and background of *CarOSense*, we introduce the evolution of automotive keyless systems and UWB basics to enable the sensing use cases in this section.

2.1 Automotive Keyless Entry Infrastructure

The key system of automobiles has evolved from using traditional physical keys to remote keyless control to unlock a car. Using remote keyless control, users can remotely lock/unlock their vehicles by pressing a button on the key fob. In the last decade, major car OEMs have adopted a passive keyless system [35], which allows users to lock/unlock/start vehicles without touching any key fobs. This passive keyless entry system provides users a convenient way to enter and start a car, where a user can put the key fob in the pocket/backpack and opens the car in a seamless and touch-less manner. Typically, such keyless entry system uses a combination of Low Frequency (LF) and Ultra-High Frequency (UHF) channels to measure the proximity of the key fob and check if the key fob is inside or within a certain range (e.g., 2m) of the car. However, such keyless entry system is subject to relay attacks [18, 22, 29, 60]. The attacker puts one or more relay devices between the car and the legitimate key fob. These relay devices create a relay channel to make the car falsely believe the user is in the unlocking zone or inside the car, and allow the attacker to unlock/start the car.

UWB is an alternative and more advanced technology for keyless entry systems. As it can eliminate relay attacks [66, 67], car OEMs have started upgrading existing keyless systems to UWB-based infrastructure [15, 33, 43, 67]. UWB radios carry explicit timing information which is defined in the IEEE 802.15.4-2015 UWB standard [30]. Such timing information is recorded by timestamping the packet received by the UWB radio. If the signal is coming from a relay device, it will take a longer time to be received by the car, as the signal actually travels through a longer distance from legitimate key to the car. By setting up a time delay restriction, UWB based system can perfectly prevent such relay attacks. Meanwhile, UWB radio is coming to smartphones [5]. With UWB-based keyless systems, users no longer need to carry additional key fobs. Instead, they can use their smartphones to lock/unlock/start their vehicles. This will fundamentally change the user experience of car access as users will also be able to share keys digitally [6, 15]. In our work, we leverage such UWB-based keyless infrastructure without adding additional hardware cost to enable per-seat occupancy sensing function in vehicles.

2.2 UWB Basics

UWB is a radio technology with a large bandwidth and low power for short-range wireless transmissions. According to the Federal Communications Commission (FCC) of the USA, UWB refers to radio technologies that have a bandwidth larger than 500MHz or 20% of the arithmetic center frequency. In the USA, the frequency range of UWB is from 3.1 to 10.6GHz, and the Power Spectral Density (PSD) limit for UWB transmitters is -41.3dBm/MHz. Table 1 shows a comparison of UWB with other wireless technologies including Radio-Frequency IDentification (RFID), ZigBee, Bluetooth, and Wi-Fi. Because of low power, high data rate, low interference, and high security features, UWB is also suitable for wireless intra-vehicle communications [9, 59], which can help reduce vehicle weight/cost and simplify electrical wiring/maintenance [64]. Our work is built on top of existing UWB infrastructure on cars to use it as a new sensing modality. Especially, *CarOSense* focuses on per-seat car occupancy sensing without introducing additional dedicated sensors.

One major advantage of UWB is the large bandwidth that provides much better time/spatial resolution than other wireless technologies. The time resolution of wireless sensing is $\tau = 1/B$ where B is the channel bandwidth.

Table 1. Comparison of Wireless Technologies

	Frequency	Bandwidth	Sensing Resolution (Time/Spatial)	Sensing Input (Granularity)	Power	Range	Data Rate	Interference
RFID	14MHz; 900MHz	200-500kHz	2-5μs/0.6-1.5km	RSS (coarse)	Low	10cm; 10m	Low	Low
ZigBee	868/915MHz; 2.4GHz	2MHz	0.5μs/150m	RSS (coarse)	Low	10m; 20m	Low	High
Bluetooth	2.4GHz	1MHz	1μs/300m	RSS (coarse)	Low	20m	Medium	High
Wi-Fi	2.4GHz; 5GHz	20-160MHz	6-50ns/1.8-15m	CSI (fine)	High	200m	High	High
UWB	3.1-10.6GHz	500-1354.9MHz	0.7-2ns/22-60cm	CIR (fine)	Low	10m	High	Low

UWB has bandwidth larger than 500MHz, so its time resolution is better than 2 nanoseconds. This corresponds to spatial resolution of 60 centimeters for electromagnetic waves with speed of 3×10^8 meters/second. A typical Wi-Fi channel with 20MHz bandwidth has spatial resolution of 15 meters [72], so it is hard to provide fine-grained sensing capabilities. A comparison of the time/spatial resolution of UWB and other wireless technologies is shown in Table 1. UWB provides more fine-grained sensing capabilities than other wireless sensing technologies, especially for in-vehicle environments with strong multi-path efforts. Moreover, UWB is more energy efficient and has lower interference than other wireless technologies, since UWB has lower power [2, 34, 58, 65]. Because of high time/spatial resolution, low power, and low interference, UWB is suitable for in-vehicle occupancy sensing.

The transmit signal of UWB, which is a sequence of pre-defined symbols in the IEEE 802.15.4 format [30], travels through multiple paths and arrives at the receiver with different amplitude attenuation and time of flight. The receive signal is compared with the known sequence of transmit symbols to compute the CIR:

$$h(t) = \sum_{i=1}^{N} a_i \delta(t - \tau_i), \tag{1}$$

where a_i and τ_i are the amplitude and time of flight, respectively, of the i-th path, and $\delta(\cdot)$ is the Dirac delta function. The UWB CIR represents the multi-path profile of how the transmit signal travels through the environment around the transmitter and receiver.

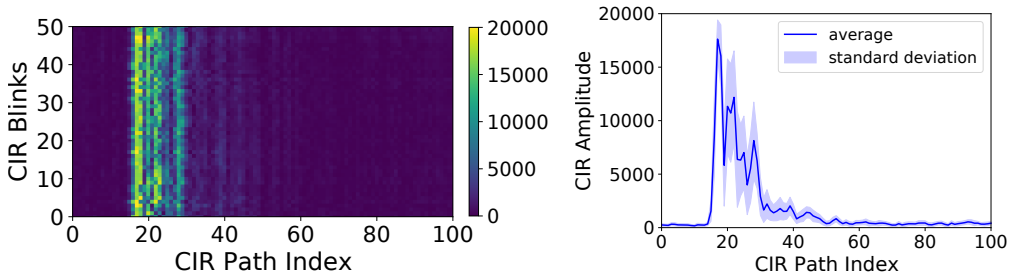
3 DESIGN

In this section, we first present a feasibility study of UWB-based occupancy sensing and demonstrate that standard machine learning algorithms do not work for unseen scenarios. Then we present the design goals of per-seat car occupancy sensing including accuracy, robustness, training efforts, real-time computation cost, versatility, and scalability. Finally, we present the proposed design including CIR processing and classification algorithms.

3.1 Feasibility Study

A time series of CIRs represents multi-path profile variations caused by any changes of the environment. This is the reason of why CIRs can be used for sensing purposes. Fig. 1a shows a heatmap of the CIR amplitude measured from multiple CIR blinks. The CIR heatmap can be represented by the average and standard deviation as shown in Fig. 1b. Different features, such as peaks/valleys, distances between peaks/valleys, number of peaks/valleys, etc., can be calculated and fed to machine learning algorithms for different sensing purposes.

Since CIRs are impacted by multi-path signal reflections from objects and humans in the car, CIR shapes and variations can be used for car occupancy sensing. Fig. 2 shows CIR shapes and variations of multiple UWB receivers for different occupancy scenarios. The transmitter is placed on the center panel, and 7 receivers are placed at different places in the car. The CIR shapes and variations show different multi-path profiles for different occupancy scenarios. For example, Receiver4 at central ceiling and Receiver6 in trunk show different CIR shapes between empty and driver seat occupied cases. Meanwhile, Receiver0 at front-left ceiling and Receiver1 at



(a) Heatmap (b) Average and standard deviation

Fig. 1. Heatmap, average and standard deviation of the amplitude of a time series of CIRs.

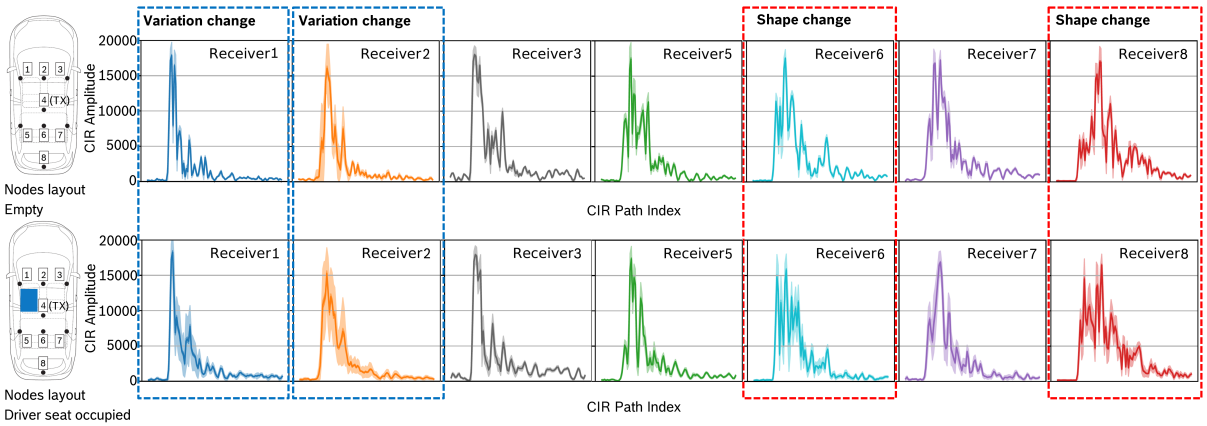


Fig. 2. CIR examples of different occupancy scenarios.

rear-view mirror have high CIR variations due to minor human movements when there is a person in the driver seat, while all UWB nodes have low CIR variations when the car is empty. In this way, CIR shapes and variations can be fed to machine learning algorithms for car occupancy sensing.

Support Vector Machine (SVM) and k Nearest Neighbor (kNN) are two of the most widely used machine learning algorithms for wireless sensing applications [48]. kNN is an instance-based machine learning algorithm. It first computes the distance, e.g., Euclidean and Hamming distance, between the test sample and each of the training samples and then classifies the testing sample based on the majority vote from the k nearest neighbors. SVM tries to maximize the functional margin, i.e., the distance to the nearest training data points of each class, by separating data points with a set of hyper-planes in a high-dimensional space. Another widely used approach is Gradient Boosting Decision Trees (GBDT) which ensembles weak machine learning models, i.e., decision trees, to iteratively improve the weak points of the previous model. It is "one of the best, if not *the* best, algorithm for dealing with nonperceptual data", which makes it one the most commonly used algorithms, along with deep learning, for top winners of Kaggle competitions [11].

However, traditional machine learning algorithms do not work for UWB-based car occupancy sensing in unseen scenarios. Fig. 3 shows the accuracy of kNN, SVM, and GBDT for car occupancy sensing using UWB the setup with 1 transmitter and 7 receivers as shown in Fig. 2. When the car location of test samples is known

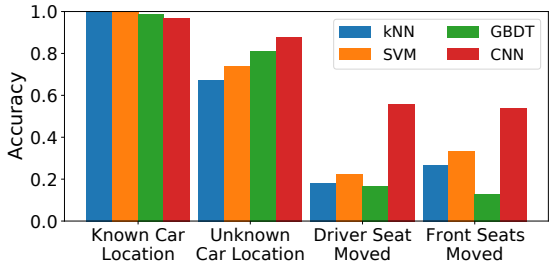


Fig. 3. Accuracy of different algorithms for car occupancy sensing in different scenarios.

during training, the accuracy of kNN, SVM, and GBDT is higher than 97%. For unknown car locations, i.e., the pre-trained model is tested by CIR data from a new location that is not seen during training, the accuracy drops to 67%, 74%, and 81%, respectively, for kNN, SVM, and GBDT. The accuracy further drops below 36% when the driver seat or both front seats are moved for test samples. Another issue for traditional machine learning models is they require a lot of human efforts for signal processing and feature engineering to design and select the right features. Therefore, more accurate and efficient algorithms are needed for UWB-based car occupancy sensing.

Recently, Convolutional Neural Networks (CNNs) are widely used for wireless sensing applications [48]. CNN requires very little or none human efforts for signal processing and feature engineering. More importantly, CNN has higher accuracy than kNN, SVM, and GBDT for unseen scenarios. We modify an image-based CNN model from C4S-C [20], which uses thermal images for in-vehicle occupancy detection, for UWB CIR data and compare its accuracy with kNN, SVM, and GBDT in Fig. 3. The accuracy of CNN is 88%, which is 21%, 14%, and 7% higher than that of kNN, SVN, and GBDT, respectively, when the car location of test samples is not seen during training. However, reusing CNNs from computer vision tasks gives low accuracy for UWB-based car occupancy sensing for unseen scenarios, since standard CNNs are usually designed and trained for computer vision tasks focusing on digital images which are very different from CIR data in many aspects such as spatial resolution and wavelength [40, 58]. As shown in Fig. 3, the accuracy of CNN is around 55% when the driver seat or both front seats are moved. Therefore, new models are needed for per-seat car occupancy sensing to be accurate and robust in different unseen scenarios, e.g., unknown car locations, multiple persons, car state changes, etc.

3.2 Design Goals

We have the following goals for per-seat car occupancy sensing with the UWB keyless infrastructure:

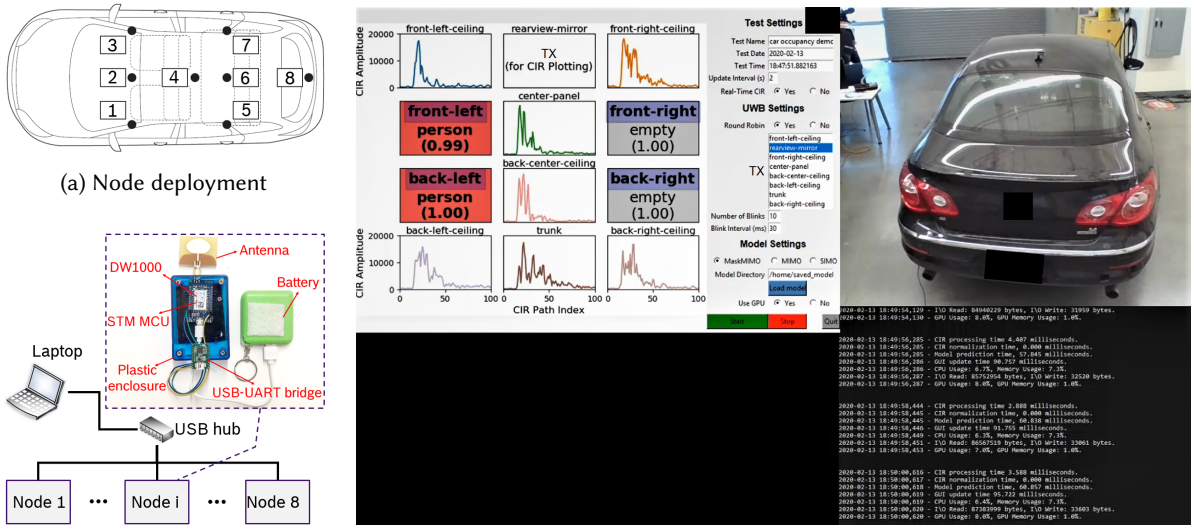
Accurate and Robust: The system is accurate and robust in different unseen scenarios such as unknown car locations, multiple persons, car seats moved, car engine is on, car is driving, car windows are open, etc.

Low Training Effort: The system requires low human effort for data collection, signal processing, feature engineering, and model training for easier adoption and to keep system cost low. It can be trained on a small set of training data and still be robust in a large set of untrained scenarios.

Lightweight Implementation: The system has low computation cost and can run in real-time on embedded devices with constrained resources.

Versatile and Scalable: The system can be trained with pre-trained models and new data without restarting the training from scratch and is scalable in terms of number of UWB nodes, new tasks, and new data.

We propose a new deep learning design, *CarOSense*, for accurate, robust, and practical per-seat car occupancy sensing with UWB, as shown in Fig. 4. There are 8 UWB nodes attached at different places in the car, as shown in Fig. 4a. Each node has a Decawave DW1000 UWB transceiver, an STM micro-controller, an antenna, and a removable battery, as shown in Fig. 4b. Each node is connected to a USB hub through a USB-UART bridge and



(a) Node deployment

(b) Node diagram

(c) Screenshot of real-time demo

Fig. 4. Node deployment, node diagram, and screenshot of real-time demo. *CarOSense* works on different platforms including computers with Windows and Linux, Google Coral development board with Mendel Linux and Raspberry Pi with Raspbian.

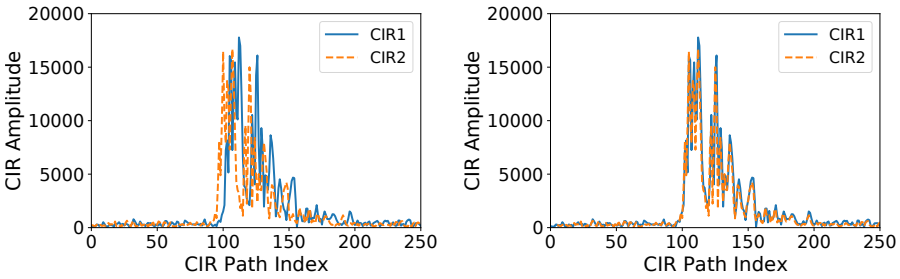
then connected to a laptop by a USB cable. Fig. 4c shows a screenshot of the real-time demo, debug console, and video capturing of the target car. The demo shows real-time CIR plots and occupancy predictions with classification scores of each car seat on the left, and system settings on the right. The proposed classification algorithm is a multi-input multi-output CNN with multi-task masking. It is accurate and robust in different unknown scenarios and has low computation cost in real-time running. *CarOSense* works on different platforms including personal computers and embedded devices such as Google Coral development board and Raspberry Pi.

3.3 The Proposed Design

CarOSense has two components: CIR processing for converting raw UWB signals to normalized CIR tensors, and classification algorithm for predicting per-seat occupancy from normalized CIR tensors.

3.3.1 CIR Processing. For UWB, the transmitter and receiver usually are not time synchronized, so CIRs measured at different times may be randomly shifted with respect to each other. Fig. 5a shows two misaligned CIRs separated with a time interval of 2 seconds. CIR alignment is needed to make a time series of CIRs represent the multi-path profile in the right manner. We use the firth path index, which is usually reported by UWB chips such as Decawave DW1000 [45], for CIR alignment. The first path index is determined by a leading edge detection algorithm that compares the receive power of each path with a threshold calculated from the noise estimation [45]. The CIRs are aligned by the first path index after removing the time shift, as shown in Fig. 5b.

There are 8 UWB nodes attached at different places in the car with 1 node as the transmitter and 7 nodes as receivers. The transmitter is changed every 30 milliseconds in round robin order, i.e., the current transmitter is changed to receiver and the next node is changed to transmitter. Each node collects CIRs and sends decoded CIR measurements to the server through the USB hub. The server conducts CIR alignment using the first path index and transforms aligned CIRs to a time series of truncated CIRs of 101 paths for each node. After every 10



(a) Before alignment (b) After alignment

Fig. 5. Amplitude of two CIRs before and after alignment.

round robin circles, which is in corresponding to roughly 2.4 seconds, the CIR amplitude of all the nodes are concatenated into 4D CIR tensors as the input. The size of 4D CIR tensors is (8, 7, 10, 101) representing 8 UWB nodes, 7 receivers for each round, 10 CIR blinks, and 101 CIR paths. 4D CIR tensors are normalized by

$$x^{train}_{i|1\le i\le n_{train}} = \frac{|cir^{train}_i| - cir^{train}_{mean}}{cir^{train}_{std}};\ x^{valid}_{j|1\le j\le n_{valid}} = \frac{|cir^{valid}_j| - cir^{train}_{mean}}{cir^{train}_{std}};\ x^{test}_{k|1\le k\le n_{test}} = \frac{|cir^{test}_k| - cir^{train}_{mean}}{cir^{train}_{std}},$$

where cir^{train}_{mean} and cir^{train}_{std} are the average and standard deviation of the CIR amplitude of training samples, $|cir^{train}_i|$, $|cir^{valid}_j|$, and $|cir^{test}_k|$ are the amplitude of 4D CIR tensors, and n_{train}, n_{valid} and n_{test} are the number of CIR tensors for training, validation, and testing. cir^{train}_{mean} and cir^{train}_{std} are calculated by only training samples, so no information of validation/testing samples is leaked to the training process. The normalized CIR tensors represent the multi-path profiles inside the car in temporal and spatial domains. For example, when a person is sitting in the driver seat, it introduces multi-path signal reflections for nearby UWB receivers. The multi-path profiles also change over time due to human activities such as gestures and breathing. These information can be used by machine learning models to learn temporal and spatial features for car occupancy sensing.

3.3.2 Classification Algorithm. Wireless signals and digital images have different properties, such as spatial resolution and field of view [40, 58], so we need new CNN models that are specifically designed for UWB data. To this purpose, we propose MaskMIMO, a multi-input multi-output CNN with a multi-task mask, for robust, efficient, scalable, and versatile car occupancy sensing with UWB. The classification algorithm should be designed based on the input data and output target. The output are occupancy predictions of each car seat. A possible model architecture is using a single label for all the combinations of different car occupancy scenarios. For example, "0000" represents an empty car and "1000" means a person in the driver seat and no person in other three seats. In this case, the number of output classes is 16 for all the combinations of "0" or "1" for 4 car seats. This increases the complexity of the model. Also, the large number of output classes could cause a bottleneck in the computation and optimization, which makes it hard to train the model. This issue can be addressed by multi-task learning which learns multiple classification tasks jointly. Multi-task learning reduces the complexity and improves the generalization, scalability, and flexibility of the classification algorithm. First, the output is divided into simpler tasks, i.e., binary classification of empty or occupied for 4 car seats. This reduces the complexity and computation cost of both the model architecture and the optimization algorithm, so the model is easier to train. Second, features learned from each task can improve other tasks, since different tasks are related. By learning multiple related tasks in parallel with a shared representation, multi-task learning is able to improve the overall performance of all tasks. Third, multi-task learning is scalable, and it is easy to add new tasks when new data are available. For example, the model can add "dog" for per-seat occupancy classification and train on the pre-trained model with

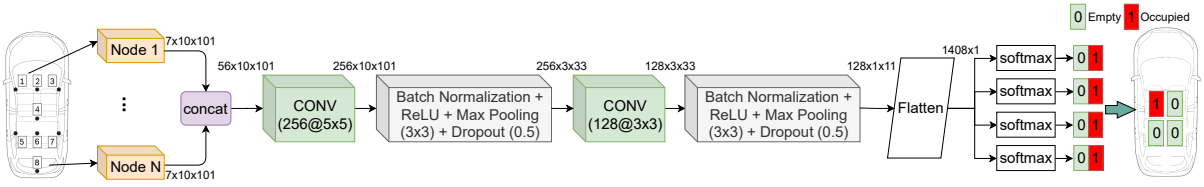


Fig. 6. SIMO model

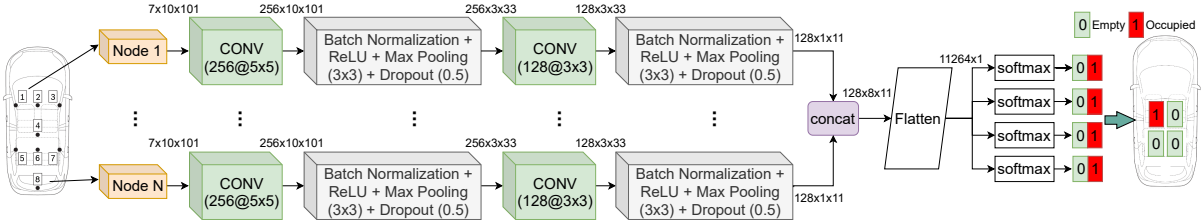


Fig. 7. MIMO model

additional data. Finally, it is flexible to add weights to different tasks for multi-task learning. For example, the model can add a higher weight for back seats if back seats have a higher priority.

The input of the classification algorithm are 4D tensors, which also impacts the design choice of the classification algorithm. 4D tensors can use 4D convolutions for extracting feature maps. But 4D convolutions have high time and space complexity [78] and are not natively supported by deep learning frameworks such as TensorFlow and PyTorch. We can replace 4D convolutions with 3D or 2D convolutions using decomposed models, such as single-input multi-output (SIMO) and multi-input multi-output (MIMO) as shown in Fig. 6 and 7. For the SIMO model in Fig. 6, the 3D CIR tensors of all the nodes are concatenated into a single 3D tensor. 2D convolutions are performed along the time and space domain. The pre-processed 3D CIR tensor of each node is fed to two convolutional layers each followed by batch normalization, Rectified Linear Unit (ReLU), max pooling, and dropout layers. For the MIMO model in Fig. 7, each node uses its own 2D convolutions, and the convolution output of all nodes are concatenated into a single layer for multi-task classification. For MIMO, the neural architecture of each node is the same but the neural weights of different nodes are different. These layers try to learn features in space and time domains for each node independently.

SIMO and MIMO do not capture the spatial features that are correlated for different nodes and different car seats. To address this issue, we add a multi-task mask to the MIMO model to learn multi-task attentions from multiple nodes. The architecture of the MaskMIMO model is shown in Fig. 8. Different UWB nodes have different weights for the performance of different car seats, since they are placed at different places inside the car. The multi-task mask learns per-seat attentions and spatial features to automatically calculate the weights of different nodes and car seats. The output of each node is concatenated as a 3D tensor and then fed to a flatten layer and 4 dense layers to calculate the output of each car seat. The final per-seat occupancy predictions are calculated by the multiplication of the multi-task weights and the multi-task model output.

The multi-task cost function is calculated by the weighted average of the expected loss of all tasks

$$J(\boldsymbol{\theta}) = \sum_{k=1}^{N_t} w_k \mathbb{E}_{(\boldsymbol{x};y_k)\sim\hat{p}_{data}} \left[L\left(f(\boldsymbol{x};\boldsymbol{\theta}), y_k\right)\right] = \frac{1}{N_s} \sum_{k=1}^{N_t} w_k \sum_{i=1}^{N_s} L(f(\boldsymbol{x}^{(i)};\boldsymbol{\theta}), y_k^{(i)}), \tag{2}$$

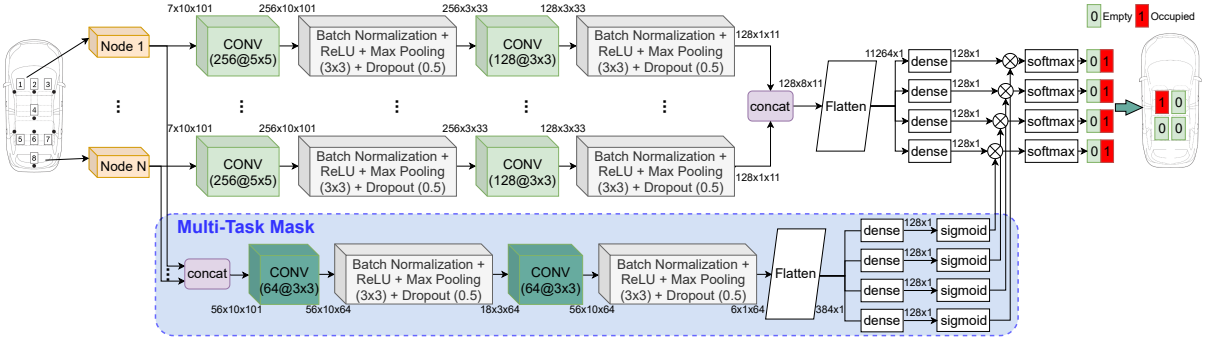


Fig. 8. MaskMIMO model

where $\boldsymbol{\theta}$ represents the model architecture and learning parameters, N_t is the number of tasks, N_s is the number of training samples, w_k is the loss weight of each task, $\hat{p}_{data}$ is the empirical distribution of the training data, $L(\cdot)$ is the loss function, $f(\boldsymbol{x};\boldsymbol{\theta})$ is the model output with input $\boldsymbol{x}$, and y_k is the ground truth label of the k-th task [25]. $L(f(\boldsymbol{x};\boldsymbol{\theta}), y_k)$ is the cross-entropy between the model output $f(\boldsymbol{x};\boldsymbol{\theta})$ and the ground truth label y_k with *softmax* loss. For MaskMIMO, $L(f(\boldsymbol{x};\boldsymbol{\theta}), y_k)$ is calculated by multiplying the output of a MIMO module with the multi-task weights learned from the *sigmoid* output of the multi-task mask. Finally, the loss function is multiplied with the loss weights to give different priorities and contributions for the loss rates of different car seats. In our experiments, the loss weight of each task is $w_k = 1$ for $N_t = 4$ tasks of the occupancy state of 4 car seats.

In summary, we propose a new deep learning design, *CarOSense*, for per-seat car occupancy sensing with UWB. First, MaskMIMO learns both independent and shared features from multi-path profiles of multiple UWB nodes. It also has a multi-task mask to learn spatial features and multi-task attentions from UWB nodes at different locations. MaskMIMO is robust for different unseen scenarios. Second, because *CarOSense* is robust for different scenarios, it can be trained by only 4 car locations and provide robust and high accuracy for different unseen scenarios. Moreover, unlike traditional machine learning approaches such as kNN and SVM that usually need feature engineering/selection, *CarOSense* can learn features automatically and needs little or none human efforts for signal processing. Third, *CarOSense* uses a multi-output CNN model so that it can be re-trained by new data or new tasks without restarting the training from scratch. Finally, the proposed model has low computation cost and can run in real-time on embedded devices with constrained resources.

4 EVALUATION

4.1 Experiment Setup

Fig. 4 shows the experiment setup of node deployment and diagram. There are 8 UWB nodes placed at different places inside the car, i.e., front-left ceiling, rear-view mirror, front-right ceiling, center panel, back-center ceiling, back-left ceiling, back-right ceiling, and trunk, as shown in Fig. 4a. Each node has a DW1000 UWB chip for sending and receiving UWB packets and collecting CIRs, as shown in Fig. 4b. All the nodes are connected to a USB hub for sending CIR measurements to the laptop. At startup, the first node is the transmitter and the other 7 nodes are receivers. Each receiver computes the CIR from the received packet preamble and sends the decoded CIR measurements to the laptop through the USB-UART bridge and USB hub. The transmitter is changed to receiver and the next node is changed to transmitter every 30 milliseconds in round robin order.

The real-time demo is implemented in Python 3 and supports different platforms. A screenshot of the real-time demo is shown in Fig. 4c. The demo supports different settings, such as showing real-time CIR plotting,

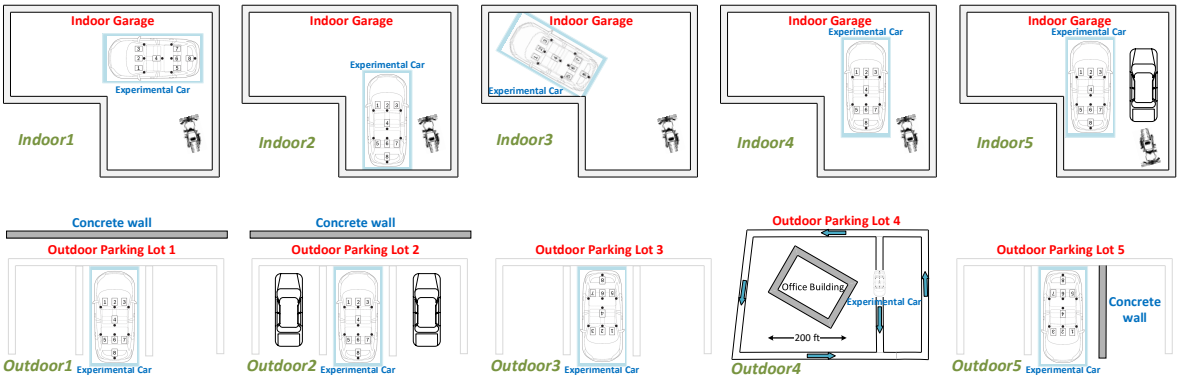


Fig. 9. Car locations. Indoor1, indoor2, indoor3, outdoor1, and outdoor2 are used for leave-one-out tests. Indoor4, outdoor3, and outdoor4 are used for stress tests of special scenarios. Indoor5 and outdoor5 are used for unseen person evaluation.

enabling/disabling round robin transmission, update interval, and pre-trained model. After all the settings are ready, the demo starts the UWB communication module by sending commands to UWB nodes through the USB hub. It first sends request to each node to collect system information of each node. Based on the settings, the demo sends node configuration to each UWB node and starts CIR data collection. Each receiver node estimates CIRs from UWB packets, and sends the decoded CIRs to the server through the USB hub. The server first conducts CIR alignment and then transforms CIRs of all nodes into 4D tensors. Each 4D CIR tensor is normalized and then fed to the pre-trained model to calculate occupancy predictions and classification scores of each car seat which are updated on the Graphical User Interface (GUI). If real-time CIR plotting is enabled, CIR amplitude of each selected receiver is calculated and updated on the GUI. Classification models are trained by a laptop with Nvidia GPU GeForce RTX 2080 Max-Q [52]. *CarOSense* is implemented and evaluated on a laptop and two embedded devices including Google Coral development board [26] and Raspberry Pi [21].

Data collection is conducted in 5 indoor and 5 outdoor locations, as shown in Fig. 9. A summary of different evaluation scenarios is shown in Table 2. In total, there are 9269 instances of 4D CIR tensors each with size of (8, 7, 10, 101). There could be no person or up to four persons simultaneously sitting in the car. There are 9 participants with different bio-metrics (height: [165cm, 182cm]; weight: [125lb, 185lb]). Each participant can sit randomly in any car seat with different motion status such as sitting still, talking with each other, making hand/head gestures, etc. The training and testing data may have different persons sitting in the same car seat and different occupancy combinations for multi-person scenarios. Some persons in the testing data are never seen during training. Each car location of indoor1-3 and outdoor1-2 is corresponding to 1 evaluation scenario, and other car locations contain 10 evaluation scenarios such as kids in the car, back seats fold down, front seats moved, etc. We run 5-fold cross-validation, leave-one-out test of unknown car locations, and stress test of unknown scenarios. We also run evaluations to investigate the impact of different factors including number of training car locations, number of UWB nodes, UWB node locations, round robin transmission, and unseen persons.

CIR data of indoor1, 2, 3, and outdoor1, 2 are used for 5-fold cross-validation and leave-one-out test, and other CIR data are used for stress test, as shown in Fig. 10. For 5-fold cross-validation, CIR data of indoor1, 2, 3, and outdoor1, 2 are randomly partitioned into 5 sets. Each set is used for validation, and the remaining sets are used for model training to find the model architecture, training hypterparameters, and learning parameters. For leave-one-out test, each car location of indoor1, 2, 3 and outdoor1, 2 is for testing and the others are for training and validation. In this case, the car locations of test data are unknown for the trained model. For stress test, a

Table 2. Summary of Evaluation Scenarios for Leave-One-Out Test and Stress Test

Section	Training Locations[1]	Trained Model[2]	Testing Location	# Instances	# Persons	Testing Scenario	Average Accuracy
Section 4.2 5-Fold Cross-Validation	indoor1, 2, 3, outdoor1, 2	$m_k, k = 1, ..., 5$	random	about $\lfloor \frac{3299}{5} \rfloor$	0 to 4	randomly split training data into 5 sets, each set used for validation and the 4 remaining sets for training	99%
Section 4.3 Leave-One-Out Test of Unknown Car Locations	indoor2, 3, outdoor1, 2	M_1	indoor1	508	0 to 1	car parked horizontally	97%
	indoor3, outdoor1, 2	M_2	indoor2	741	0 to 4	car parked vertically	91%
	indoor2, outdoor1, 2	M_3	indoor3	630	0 to 4	car parked diagonally	97%
	indoor2, 3, outdoor2	M_4	outdoor1	519	0 to 4	sometimes persons/cars moving around	94%
	indoor2, 3, outdoor1	M_5	outdoor2	901	0 to 4	sometimes persons/cars moving around; 2 SUVs on the left and right of the car	94%
Section 4.4 Stress Test of Unknown Scenarios	indoor2, 3, outdoor1, 2	M_0	indoor4	718	0 to 1	1 backpack/box in each seat	90%
			indoor4	670	0 to 1	1 baby seat or water bottle in each seat	89%
			indoor4	310	0 to 3	1 or 2 kids (6Yr old, 50in height, 45-55lbs weight) in back seats, w/ or w/o driver	83%
			indoor4	738	0 to 4	front left window is open	84%
			indoor4	761	0 to 4	all windows are open	86%
			indoor4	734	0 to 4	driver seat is pushed back	87%
			indoor4	745	0 to 4	front seats are pushed back	80%
			indoor4	204	0 to 2	back seats are fold down	76%
			outdoor3	736	0 to 4	car engine is on	96%
			outdoor4	354	1 to 4	car is driving in low-speed	92%
Section 4.5	indoor2,3, outdoor1,2	M_0–M_5	All	All	0 to 4	impact of number of training car locations	73%-95%
Section 4.6	indoor2,3, outdoor1,2	M_0–M_5	All	All	0 to 4	impact of number of nodes	63%-95%
Section 4.7	indoor2,3, outdoor1,2	M_0	indoor5, outdoor1,5	All	0 to 4	impact of unseen persons	75%-96%
Section 4.8	indoor2,3, outdoor1,2	M_0	All	All	0 to 4	real-time computation cost: system usage and time cost	N/A

[1] Data randomly splitted into 80%/20% for training/validation. 10 epochs for 5-fold cross-validation, 100 epochs for leave-one-out/stress test.
[2] Different trained models have the same neural architecture, but the learning parameters are different because of different training data.

model M_0 is trained by CIR data of indoor2, 3 and outdoor1, 2 and tested by 10 unseen scenarios for robustness evaluation. In addition to unseen scenarios, stress test has 5 new participants that are not seen during training. For both leave-one-out and stress tests, the training data is randomly partitioned into 80%/20% for training/validation. We compare the proposed models, including SIMO, MIMO, and MaskMIMO, with a baseline CNN model similar as C4S-C [20] in terms of accuracy, precision, recall, and F1 score.

Performance results of per-seat accuracy, average accuracy, precision, recall, and F1-score of k-fold cross-validation, leave-one-out, and stress tests are shown in Section 4.2, 4.3, and 4.4. Section 4.5 shows the impact of number of training car locations, and Section 4.6 presents the impact of number of UWB nodes, UWB node locations, and round robin transmission. Section 4.7 shows evaluation results of the impact of unseen persons.

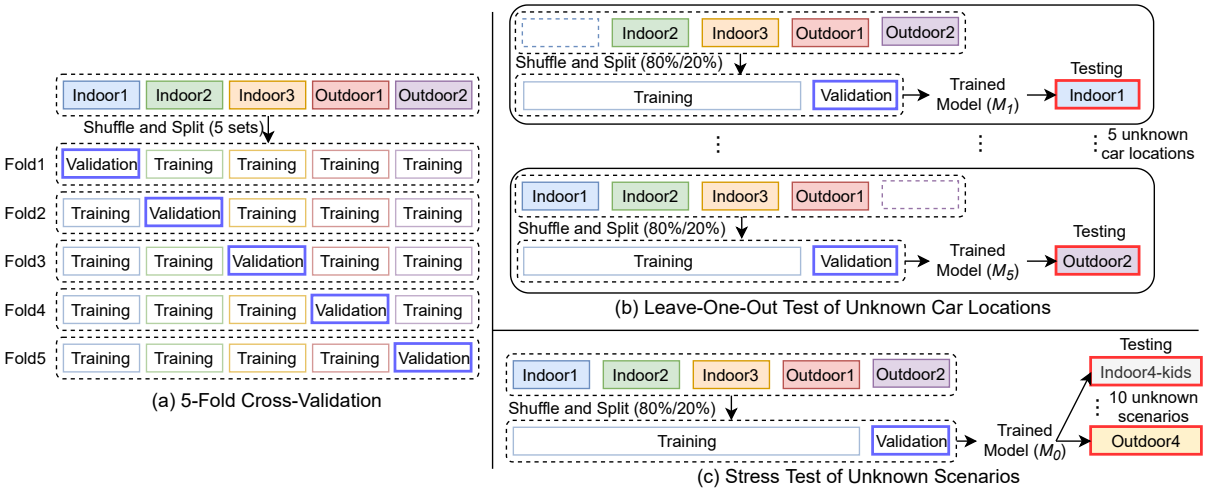
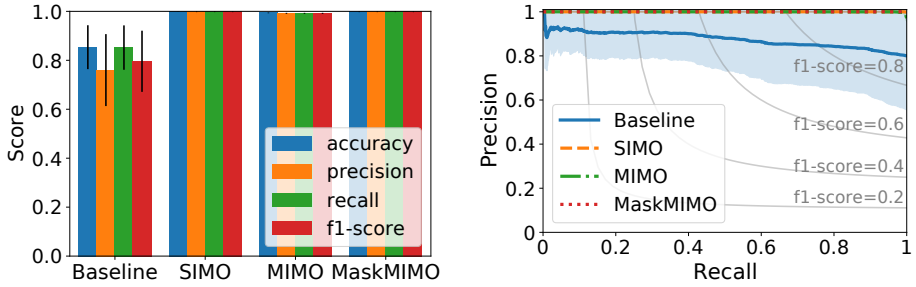


Fig. 10. Evaluation methods: (a) 5-fold cross-validation (Section 4.2), (b) leave-one-out test of unknown car locations (Section 4.3), (c) stress test of unknown scenarios (Section 4.4).

Section 4.8 shows evaluation results of real-time computation cost including system usage and time consumption of *CarOSense* running on different platforms.

4.2 5-Fold Cross-Validation



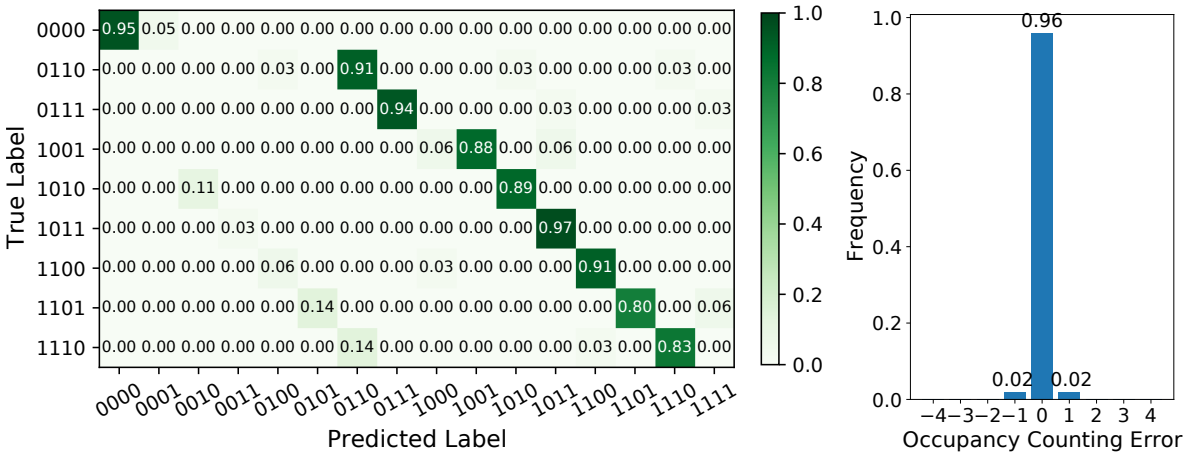
(a) Accuracy, precision, recall, F1 score. (b) Precision-recall curve.

Fig. 11. Accuracy, precision, recall, and F1 score of 5-fold cross-validation.

One of our goals is to get accurate and robust results for a large set of untrained scenarios with the model trained by a small set of training data. So only the data from indoor1, 2, 3, and outdoor1, 2 are used for 5-fold cross-validation, and other data of 10 scenarios are only used later in the stress test. Evaluation results of accuracy, precision, recall, and F1 score of 5-fold cross-validation are shown in Fig. 11. The error bars in Fig. 11 and all the following bar graphs represent standard deviation of performance scores. SIMO, MIMO, and MaskMIMO have higher than 99.9% values for all the 4 performance metrics. The accuracy of the baseline CNN is about 84%. Fig. 11b shows the precision-recall curve for the trade-off between precision and recall for different thresholds. A high area of the regions under the curve represents both high recall and high precision, wherein high precision

means a low false positive rate, and high recall represents a low false negative rate. The shaded area represents the upper and lower bounds of statistical performance results. The baseline CNN has the lowest score and the largest shaded area, which means that it has less stable/robust performance than other models.

4.3 Leave-One-Out Test of Unknown Car Locations



(a) All combinations of 4 car seats for misclassified occupancy states. Other states with 100% accuracy are not shown in the figure.

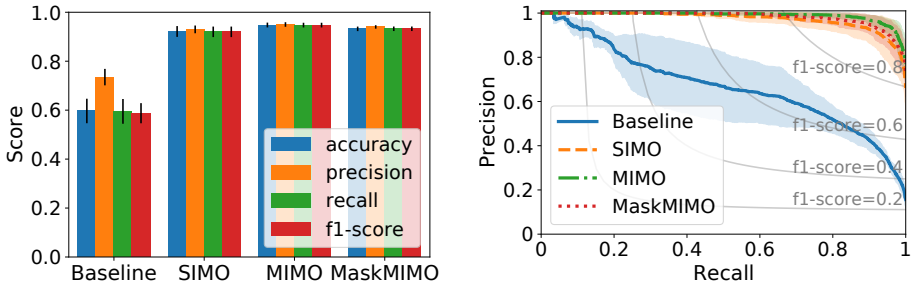
(b) Histogram of occupancy counting error for all 16 states.

Fig. 12. Multi-task confusion matrix and histogram of occupancy counting error for leave-one-test of unknown car location. The model is trained by indoor2, indoor3, and outdoor1 and tested by outdoor2.

Leave-one-out test is used to evaluate whether *CarOSense* is accurate for unknown car locations. Leave-one-out test is conducted by testing pre-trained models on CIR data from unknown car locations that are not seen during training. Fig. 12 shows performance results of a model trained by indoor2, indoor3, and outdoor1 and tested by outdoor2. Note that indoor1 is not included in training since it has no more than one person while other car locations have up to 4 persons. Performance results of when indoor1 is not included in training are almost the same as when indoor1 is included in training. Fig. 12a shows the confusion matrix of misclassified states for multi-task classification combinations, where 0 represents empty and 1 represents occupied by a person for each car seat. For all misclassified states, no more than one car seat has classification errors. For example, 14% of "1110" are misclassified as "0110", which means 14% of "driver seat occupied" are misclassified as "driver seat empty" and all other car seats are correctly recognized. The histogram of occupancy counting errors for all the 16 occupancy states is shown in Fig. 12b. 96% of test samples have no counting error, 2% have counting error of 1 person, and no samples have error larger than 1 person.

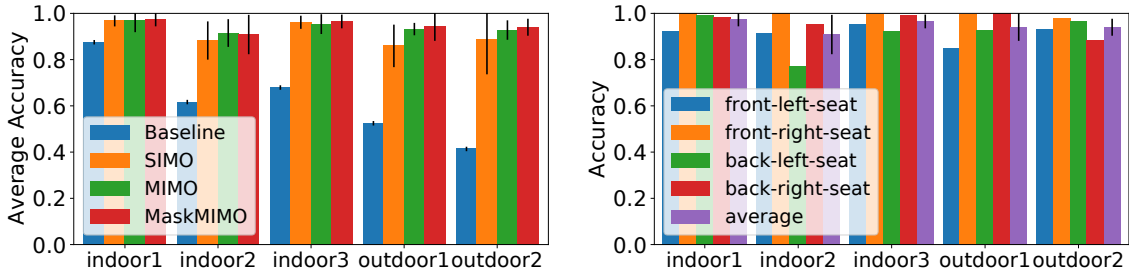
Performance results of different models are shown in Fig. 13. The accuracy, precision, recall, and F1 score are 60%, 73%, 60%, and 59% for baseline CNN and higher than 92% for other models, as shown in Fig. 13a. Precision-recall curves are shown in Fig. 13b with baseline CNN having the lowest area under the curve and the largest shaded area between lower/upper statistical bounds. This means SIMO, MIMO, and MaskMIMO are much more accurate, robust, and stable than baseline CNN.

Fig. 14a shows the average accuracy of SIMO, MIMO, and MaskMIMO for leave-one-out test. The average accuracy of MaskMIMO is 94.6%. The gap between MaskMIMO and SIMO/MIMO and between different car locations is not very big, which indicates that all the three models provide accurate and robust classification



(a) Accuracy, precision, recall, F1 score. (b) Precision-recall curve

Fig. 13. Accuracy, precision, recall, and F1 score of leave-one-out test of unknown car locations.



(a) Average accuracy of different models (b) Per-seat accuracy of MaskMIMO

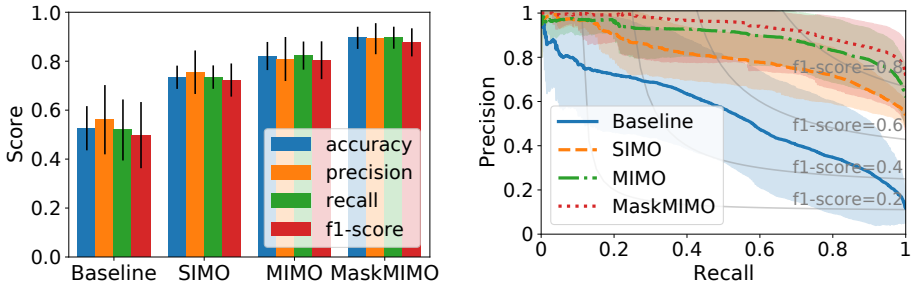
Fig. 14. Accuracy of leave-one-out test of unknown car locations.

results for unknown car locations. The baseline CNN has 90% accuracy only for indoor1 with no more than 1 person in the car, and its accuracy drops below 70% for other car locations of multi-person scenarios. Fig. 14b shows the accuracy results of per-seat classifications of MaskMIMO. Different car seats have similar per-seat accuracy for indoor1, 3, and outdoor1, 2. For indoor2, the accuracy of the back left seat is lower than 80% while all the other three seats have accuracy higher than 90%.

4.4 Stress Test of Unknown Scenarios

Stress test is used to demonstrate that *CarOSense* is robust for unknown scenarios. Stress test has not only unknown car locations but also unseen car states. For stress test, the pre-trained model is trained with CIR data of indoor2, 3, and outdoor1, 2 and tested by a new car location with 10 unseen scenarios, such as front seats pushed back, back seats fold down, and kids (6Yr old, 50in height, 45-55lbs weight) in back seats, that are not seen during training. Fig. 15 shows the accuracy, precision, recall, and F1 score of different models for stress test. Compared with 5-fold cross-validation and leave-one-out test of unknown car locations, all models have lower scores for stress test because of not only unknown car locations but also unseen and changed car states. Scores of baseline CNN drops below 56%, and MaskMIMO still provides relatively high scores of higher than 87%, as shown in Fig. 15a. Stress test is much more challenging than 5-fold cross-validation and leave-one-out test, so all models have wider range of lower/upper statistical bounds for the precision-recall curve, as shown in Fig. 15b.

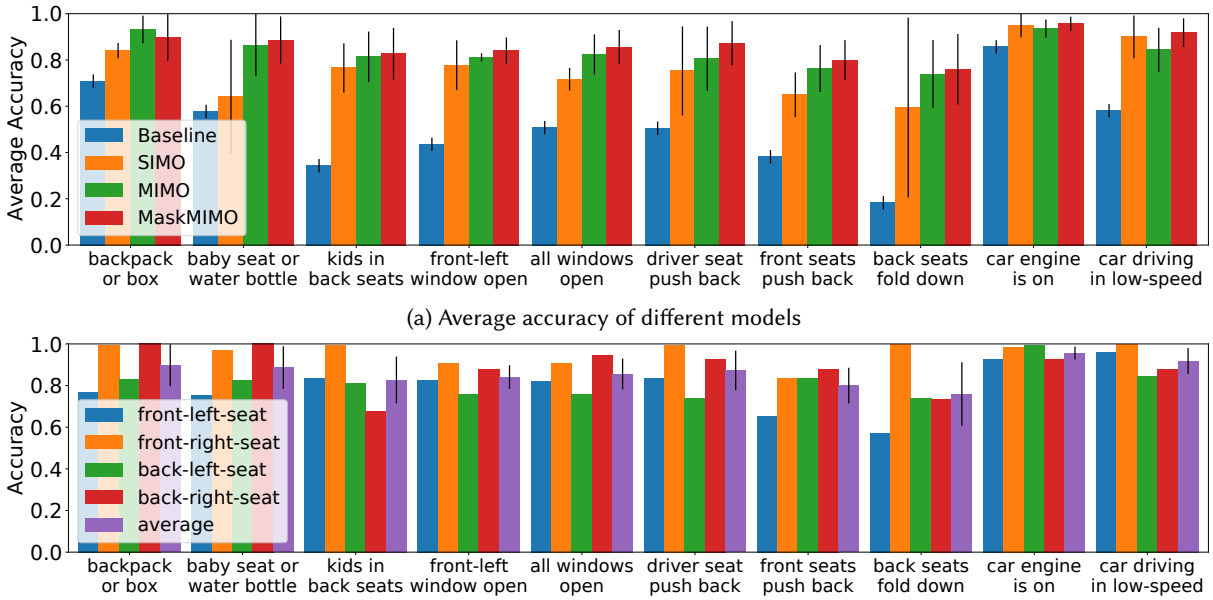
Fig. 16a shows the average accuracy of SIMO, MIMO, and MaskMIMO for stress test. The average accuracy of stress test is 8% lower than that of leave-one-out test. This is because leave-one-out test only has unknown car



(a) Accuracy, precision, recall, F1 score.

(b) Precision-recall curve

Fig. 15. Accuracy, precision, recall, and F1 score of stress test of unknown scenarios.



(a) Average accuracy of different models

(b) Per-seat accuracy of MaskMIMO

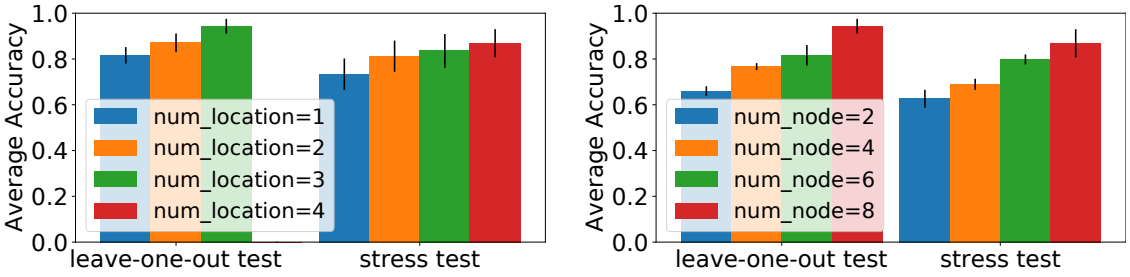
Fig. 16. Accuracy of stress test of unknown scenarios.

locations while stress test has not only unknown car locations but also unknown car states. For most scenarios, MaskMIMO has higher accuracy than SIMO and MIMO. The average accuracy of MaskMIMO for stress test is 87.0%. This means that *CarOSense* is robust for new unseen scenarios. SIMO has the lowest accuracy and is not robust for some unknown scenarios. For example, the average accuracy of SIMO is 64%, 65%, and 59%, respectively, for baby seat or water bottle in each seat, front seats pushed back, and back seats fold down, which is only slightly better than the naive method with random guessing.

Evaluation results of per-seat accuracy of MaskMIMO are shown in Fig. 16b. First, stress test has the lowest accuracy when back seats are fold down. In this case, the multi-path environment is changed dramatically: signal reflections from back seats do not exist anymore, and signals between the node in the trunk and other nodes are

changed from None Line of Sight (NLoS) to Line of Sight (LoS). MaskMIMO has an average accuracy of 76% in this case, so it is robust for dramatic changes of car states. Second, unlike leave-one-out test, per-seat accuracy could be very different for different car seats and for different scenarios. For example, when the driver seat is pushed back, left seats have much lower accuracy than the other two seats. The reason is that multi-path signal reflections of the left side of the car are very different from the training data because of the movement of the driver seat. The accuracy of the back right seat is lower than 70% when there are kids in back seats. The kids are reading books in back seats, so there are almost no body movements. Besides, the kids have much smaller body shape than adults. So signal reflections from back seats are different for training and testing data. Since the model is trained by CIR data of only adults, it gives low accuracy for back seats when there are kids in back seats.

4.5 Impact of Number of Training Car Locations



(a) Impact of number of training car locations

(b) Impact of number of UWB nodes

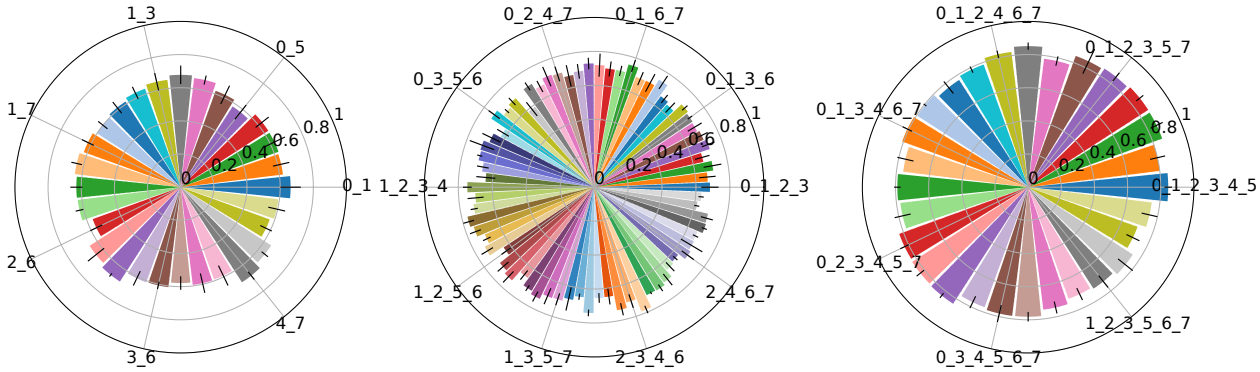
Fig. 17. Impact of number of training car locations and number of UWB nodes.

Fig. 17a shows the impact of number of training car locations on average accuracy for leave-one-out and stress tests. For leave-one-out test, the average accuracy is 95%, 87%, and 82%, respectively, for 3, 2, and 1 training car locations. For stress test, the average accuracy is 87%/84%/81%/73% for 4/3/2/1 car locations used during training. This means *CarOSense* is robust in terms of different car locations and does not require a lot of training data from different car locations. For example, the average accuracy is already higher than 80% when the model is trained using only 2 car locations: 87% for leave-one-out test and 81% for stress test. This reduces the human efforts for data collection and model training.

4.6 Impact of Number of UWB Nodes and Node Locations

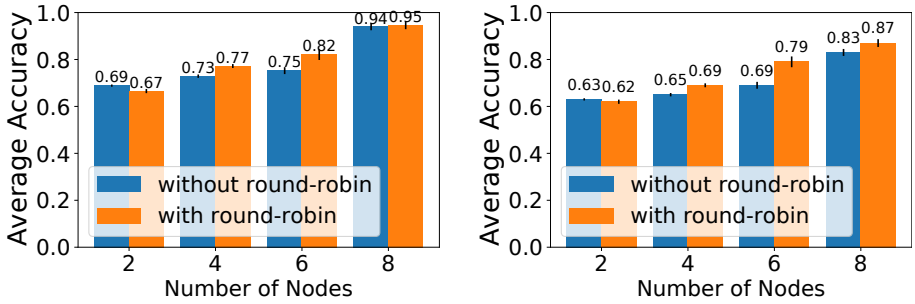
The impact of number of UWB nodes for leave-one-out test and stress test is shown in Fig. 17b. The model is trained and tested with a certain number of UWB nodes. The final accuracy is calculated by the average of all the possible combinations of different node locations. Fig. 17b shows that the number of UWB nodes has a big impact on average accuracy. The average accuracy of leave-one-out/stress test is 95%/87% when there are 8 UWB nodes. It drops to 82%/80% for 6 UWB nodes and further drops to 66%/63% when there are only 2 UWB nodes.

Fig. 18 shows the average accuracy of different node locations when there are 2, 4, and 6 UWB nodes. Labels in Fig. 18 represent which nodes are selected. For example, "1_3" in Fig. 18a means CIR data of node 1 and 3 are used for model training and testing. There is no big difference for the average accuracy of different node locations, which means *CarOSense* is robust for different UWB node locations. Another factor that influences the accuracy is round robin transmission, as shown in Fig. 19. With round robin, the transmitter is changed to receiver and the next node is changed to transmitter periodically. Without round robin, the transmitter and receivers are not changed. Round robin transmission provides more information about the multi-path environment inside the car,



(a) $\binom{8}{2}$ = 28 cases for 2 out of 8 nodes (b) $\binom{8}{4}$ = 70 cases for 4 out of 8 nodes (c) $\binom{8}{6}$ = 28 cases for 6 out of 8 nodes

Fig. 18. Impact of UWB node locations for stress test of unknown scenarios.



(a) Leave-one-out test (b) Stress test

Fig. 19. Impact of UWB transmission with or without round robin.

so it has higher accuracy than when there is no round robin. For example, as shown in Fig. 19b, when there are 6 UWB nodes, the average accuracy with round robin of stress test is 79% which is 10% higher than that without round robin. The CIR tensor size with and without round robin is (6, 5, 10, 101) and (1, 5, 10, 101), respectively. Round robin transmission provides 6 times of multi-path features compared with when there is no round robin, so the model trained with round robin is more robust for stress test of different unknown scenarios.

4.7 Impact of Unseen Persons

Table 3 shows evaluation results of the impact of unseen persons. For single-person scenarios, we collect additional data for unseen person evaluation from three car locations, indoor5, outdoor1, and outdoor5. SIMO, MIMO, and MaskMIMO give robust performance with average accuracy from 93% to 96% for unseen persons and unknown car locations. For multi-person scenarios, MaskMIMO has an average accuracy of 87% for stress test with about 60% unseen persons, unknown car locations, and unseen car states. For unknown car locations with unseen kids, the accuracy MaskMIMO is 75% which is 40% higher than that of the baseline CNN. The kids have much smaller body shape than adults in training data, and MaskMIMO still provides relatively high accuracy. We also evaluate the unseen person scenarios in real-time demo setup. We tested *CarOSense* in real time with 4 volunteers, who are unseen to training data, with different body shapes (height: [170cm, 202cm]; weight: [135lb, 200lb]) and the

Table 3. Impact of Unseen Persons

# Persons	Testing Scenario			Average Accuracy			
	car location	person	car state	Baseline	SIMO	MIMO	MaskMIMO
single-person	unknown	seen	known	0.90	0.92	0.94	0.96
		100% unseen	known	0.83	0.93	0.94	0.94
		100% unseen	unknown	0.80	0.93	0.95	0.94
multi-person	unknown	seen	known	0.68	0.89	0.93	0.94
		60% unseen	known	0.54	0.87	0.93	0.94
		60% unseen	unknown	0.53	0.76	0.83	0.87
		100% unseen kids	unknown	0.35	0.69	0.73	0.75

average accuracy is 90%. Therefore, *CarOSense* is robust for the combined effects of unseen persons, different body shapes, unknown car locations, and unseen car states.

4.8 Real-Time Test of Computation Cost

Table 4. Computation Cost of Real-Time Demo

Platform	Operating System	Inference Device	Inference Framework	System Usage			Time per CIR Tensor (ms)		
				CPU Usage	Memory Usage	GPU Usage	CIR Processing	Model Inference	Total
Laptop (MSI GS75 Stealth 9SG)[1]	Windows 10, 64-bit x86_64	GPU	TensorFlow	6.3%	7.1%	0.1%	3	63	66
		CPU	TensorFlow	6.5%	0.9%	N/A	3	75	78
	Ubuntu 18.04, 64-bit x86_64	GPU	TensorFlow	6.4%	8.6%	0.1%	4	109	113
		CPU	TensorFlow	6.4%	1.2%	N/A	4	107	111
Google Coral Dev Board[2]	Mendel Linux 4.0, 64-bit aarch64	Edge TPU	TensorFlow	25.4%	25.3%	N/A	39	189	228
			TensorFlow Lite	24.6%	6.0%	N/A	39	21	60
Raspberry Pi 4[3]	Raspbian 10, 32-bit armv7l	CPU	TensorFlow	25.2%	5.2%	N/A	24	131	155
			TensorFlow Lite	25.2%	1.2%	N/A	25	18	43

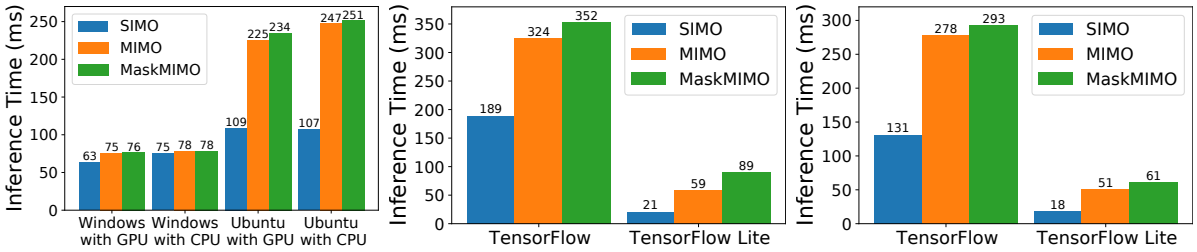
[1] CPU: Intel i9-9880H @ 2.30GHz (8 Cores); RAM: 32GB DDR4 @ 2666MHz; GPU: Nvidia GeForce RTX 2080 Max-Q @ 990MHz with 8GB GDDR6 VRAM @ 1500MHz [52].

[2] CPU: NXP i.MX 8M SoC @ 1.5GHz (Quad-core Cortex-A53) plus Cortex-M4F; RAM: 1GB LPDDR4 SDRAM @ 1600MHz; Edge TPU: Google Edge TPU ML accelerator coprocessor: 4 trillion operations per second (TOPS), 2 TOPS per watt [26].

[3] CPU: Broadcom BCM2711 SoC @ 1.5GHz (Quad-core Cortex-A72); RAM: 4GB LPDDR4 SDRAM @ 3200MHz [21].

Table 4 shows the computation cost, including system usage of CPU, GPU and memory, and time consumption of CIR processing and model prediction, of the real-time demo running on different platforms with different inference devices and different inference frameworks. Model inference is implemented by the same TensorFlow version for Windows and Ubuntu on the laptop. For Google Coral and Raspberry Pi, TensorFlow Lite is used as the inference framework. For comparison of model size, accuracy, and inference time, TensorFlow is also used for Google Coral and Raspberry Pi. CIR processing includes CIR data unpacking/decoding, CIR alignment, and CIR normalization. For Google Coral and Raspberry Pi, memory usage and model inference time of TensorFlow are

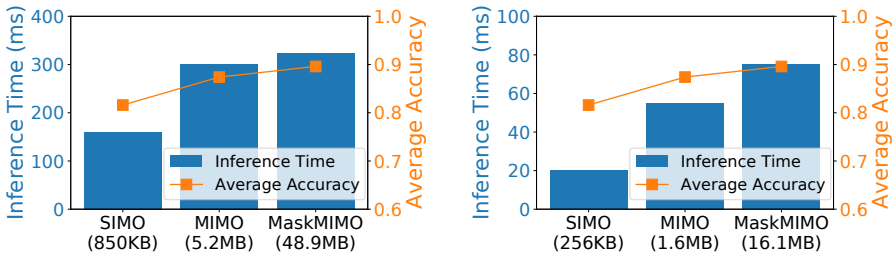
much higher than that of TensorFlow Lite. Using TensorFlow Lite as the inference framework, the total running time is 60 milliseconds and 43 milliseconds, respectively, for Google Coral and Raspberry Pi. So the real-time demo is practical for running on embedded devices with constrained resources.



(a) MSI Laptop w/ TensorFlow (b) Google Coral Dev Board w/ Edge TPU (c) Raspberry Pi 4 w/ CPU

Fig. 20. Inference time per CIR tensor of different models running on different platforms with different TensorFlow versions.

Fig. 20 shows the inference time of SIMO, MIMO, and MaskMIMO on different platforms. For the laptop with TensorFlow as shown in Fig. 20a, there is no big difference between CPU and GPU for both Windows and Ubuntu. The reason is that developed models are lightweight (as per design goal) and do not benefit from having GPU resources. For SIMO, the inference time of Ubuntu is about 50% higher than that of Windows. For MIMO and MaskMIMO, Ubuntu has about 3 times of inference time as Windows. The inference time of different models are similar for Windows. For Ubuntu, MIMO and MaskMIMO have much higher inference time than SIMO. The high inference time for Ubuntu and the big gap between SIMO and MIMO/MaskMIMO on Ubuntu could be because of different compilers and TensorFlow implementations. We leave the investigation and optimization of the inference time of TensorFlow on Ubuntu as future work. Similar to Ubuntu on the laptop, MIMO and MaskMIMO have much higher inference time than SIMO for both Google Coral development board and Raspberry, as shown in Fig. 20b and 20c. Since TensorFlow Lite is optimized for mobile, embedded, and Internet of Things (IoT) devices [27], it has much lower inference time than TensorFlow for different models on embedded devices.



(a) TensorFlow (b) TensorFlow Lite

Fig. 21. Inference time, accuracy, and model size of different models and different TensorFlow versions on embedded devices.

Fig. 21 shows a comparison of model size, accuracy, and inference time of different models using TensorFlow and TensorFlow Lite on embedded devices. TensorFlow Lite significantly reduces inference time without sacrificing classification accuracy compared with TensorFlow. The mode size of TensorFlow is 850KB, 5.2MB, and 48.9MB, respectively, for SIMO, MIMO, and MaskMIMO. TensorFlow Lite uses the same pre-trained model as TensorFlow

and converts it to a smaller model by a converter function [27]. The model size of TensorFlow Lite is 256KB, 1.6MB, and 16.1MB, respectively, for SIMO, MIMO, and MaskMIMO, which is about 1/4 of the size of the same model of TensorFlow. Because of smaller model size and library size, TensorFlow Lite has much lower inference time than TensorFlow. At the same time, TensorFlow Lite provides the same accuracy as TensorFlow. For example, MaskMIMO has inference time of 75 and 323 milliseconds, respectively, for TensorFlow Lite and TensorFlow, and the same average accuracy of 89.6% for both inference frameworks, as shown in Fig. 21b.

5 RELATED WORK

5.1 Car Occupancy Sensing

Table 5. Related Work on Car Occupancy Sensing

Reference	Input	Algorithm	Application	Testing Method	Performance
C4S-C [20]	Thermal Image	Multi-Task CNN	In-Vehicle Occupancy Detection/Counting	1313 samples, k-fold cross-validation	occupancy counting accuracy: 91%
TI-AWR1642 [32]	mm-Wave FMCW Radar	Range-angle heatmap	Rear-Seat Animated Occupancy Detection	lab and car setup	detection error: 0.3%/0.2% (lab/car)
Xu-2020 [73]	Wi-Fi CSI	Markov Chain, DP	In-Vehicle Occupancy Counting	up to 4 subjects, 1 scenario	occupancy counting accuracy: 81%
Luo-2017 [47]	On-Board Motion Sensor	Door Sequencing	In-Vehicle Occupancy Detection/Counting	Matlab simulation	N/A
Zangl-2008 [77]	Capacitive Sensor	Electric Field Signal Spectra	In-Vehicle Occupancy Classification	laboratory experiments	N/A
Sterner-2012 [63]	Electromagnetic Wave	Signal Reflection	In-Vehicle Occupancy Detection	Finite Element Method (FEM)	N/A
Delphi PODS [28]	Weight Sensor	N/A	Front-Seat Occupancy Detection	N/A	N/A
Hyundai-Kia [4, 37]	Ultrasound	N/A	Rear-Seat Occupancy Detection	N/A	N/A
Others [38]	RF, Radar, etc.	N/A	In-Vehicle Occupancy Detection	N/A	N/A
CarOSense (our design)	Ultra-Wideband	Multi-Task CNN	In-Vehicle Occupancy Detection/Counting	8 car locations, 15 scenarios; k-fold cross-validation, leave-one-out/stress test, 9269 samples	accuracy: 99.9%/94.4%/87.0% (cross-validation/ leave-one-out/stress test)

Occupancy information provides important input signals for applications such as adaptive control of heating, ventilation, and air-conditioning (HVAC) systems. Occupancy sensing helps enable and/or enhance applications such as resource allocation, energy saving, safety, security, and human-machine interaction. It can also be used to trigger other applications such as human identification, activity recognition and tracking. Recent car occupancy sensing solutions focus on weight sensors [28], cameras [20], ultrasound [4, 37], and Radars [10, 32, 46]. Examples of available technologies for in-vehicle occupancy sensing can be found at [38]. Table 5 shows a summary of related works on car occupancy sensing using different signals. Luo et al. presents in-vehicle occupancy detection and counting based on sequencing of door opening/closing and weight difference compared with the empty car. However, this method only works when the vehicle is remotely started and cannot distinguish which seat is occupied. Weight sensors are not accurate enough to identify between animate/inanimate objects. Moreover,

weight sensors have not been adopted for rear seats because of additional cost, immature rear airbag deployment techniques, wiring/harness maintenance and reconfiguration constraints. C4S-C [20] uses a thermal imaging camera and a multi-task CNN for in-vehicle occupancy detection and counting. However, thermal camera solution is very cost prohibitive. At the same time, we also applied a baseline CNN similar as C4S-C [20] for UWB-based car occupancy sensing, but it has less than 55% accuracy for unseen scenarios since it is designed for images but not for UWB data. RGB color cameras can be potentially used for in-vehicle occupancy sensing, but camera-based solutions have privacy issues [61] and are sensitive to lighting conditions. Ultrasound is used for detecting movements in rear seats for some vehicles [4, 37], but ultrasound techniques have performance limitations and are not pet friendly when frequencies used are less than 80kHz [14, 70]. UWB signals are not in the hearing range of animals and have extremely low power [58], so UWB sensing is harmless for humans and animals. Texas Instruments uses Radar to detect life forms including adults, children, and pets in vehicles with a field of view of ±60 degrees [32]. Radar requires high bandwidth and multiple antennas which translate to high cost and power consumption [10, 46]. Moreover, all of these solutions incur additional dedicated hardware and installation cost of wires/cable harness, as they are not available as standard offerings in a car.

5.2 In-Vehicle Sensing

Fridman presents seven principles of effective shared autonomy, such as human sensing, shared perception-control, and deep personalization, for Human-Centered Autonomous Vehicle (HCAV) Systems [23]. In-vehicle sensing about the driver, passengers, and vehicle is an important component to follow and implement the principles for HCAV. Table 6 shows a summary of algorithms, applications, testing methods, and performance of in-vehicle sensing using different signals. An overview of the best practices and recent advances of in-vehicle sensing is given in [57]. Kim et al. present driver facial activity recognition with a camera and multi-task CNN [39]. Camera-based in-vehicle sensing has privacy issues and is sensitive to lighting conditions. Sleeper et al. present a survey on the perceptions of vehicle-based sensing and recording from 364 participants [61]. The survey shows that most people are concerned about the comfort and privacy of vision-based sensing and recording. Participants tend to feel more comfortable with privacy preserving sensing technologies. Chuang et al. presents driver head posture recognition using wearable motion sensors and Hidden Markov Model (HMM) [12]. Wearable-based solutions are not contactless and require users wearing sensors at specific locations.

Recently, wireless signals are widely used for sensing purposes and could be a potential solution for privacy preserving and contactless in-vehicle sensing. Smith et al. propose driver gesture recognition with Frequency-Modulated Continuous Wave (FMCW) Radar and Random Forest [62]. Impulse Radio Ultra-Wideband (IR-UWB) Radar is used for vital sign monitoring of drivers [44] and multiple targets in vehicles [76]. Radar can also be used for in-vehicle occupancy detection [32]. Radar-based solutions require dedicated sensing hardware thereby introducing additional installation cost. Xie et al. propose ViHOT for head tracking using Wi-Fi signals in vehicles [71]. WiCAR [68] and WiDriver [17] use Wi-Fi for driver activity recognition. WiBot uses Wi-Fi to detect head and arm movements of drivers [55]. Melgarejo et al. use Wi-Fi RSSI and phase for hand gesture recognition in vehicles [50]. Wi-Fi CSI is used for in-vehicle occupancy counting and respiration monitoring in [73]. RFexpress uses RF RSSI for driver emotion recognition [56]. RF-CAR uses RFIDs with CNN and domain adaptation for driver activity recognition [69]. *CarOSense* uses UWB as the sensing modality, which has higher spatial resolution than Wi-Fi and RFIDs. BreathListener [74] uses Energy Spectrum Density (ESD) of acoustic signals to monitor breathing patterns of drivers. SteerTrack [75] is a device-free approach for tracking the rotation angle of steering wheel using audio signals on smartphones. Acoustic sensing is not pet friendly for frequencies less than 80kHz [14, 70]. UWB sensing does not have this issue because the carrier frequency of UWB signals is not in the hearing range of humans and animals. UWB sensing is also harmless because of its extremely low heating power and photon energy [58]. Moreover, acoustic sensing has very short range ($<$ 1m) and usually

Table 6. Related Work on In-Vehicle Sensing

Reference	Input	Algorithm	Application	Testing Method	Performance
Kim-2019 [39]	RGB Camera	Multi-Task CNN	Driver Facial Activity Recognition	driving simulator, 12 subjects	accuracy: 89% (3 facial behaviors), 93% (3 driver statuses)
Chuang-2015 [12]	Wearable Motion Sensor	Hidden Markov Model	Driver Head Posture Recognition	lab/car (n=129/ 122), 5-fold cross-validation	accuracy: 100%/99.2% (lab/car, 6 postures)
Smith-2018 [62]	FMCW Radar	Random Forest	Driver Gesture Recognition	2 gesture sets, 8 subjects	accuracy: >90% (3 gestures)
Leem-2017 [44]	IR-UWB Radar	Data Fitting	Driver Vital Sign Monitoring	7 motion cases, 5 subjects	error: 0.5/1.3 bpm (breathing/heart rate)
Yang-2018 [76]	IR-UWB Radar	Variational Mode Decomposition	In-Vehicle Multiple Targets Vital Sign Monitoring	driver only, driver and 1 passenger	heart rate accuracy: 93%/ 88% (1/2 targets)
RFexpress [56]	RF RSSI	kNN	Driver Emotion Monitoring	driving simulator, 8 subjects, 10-fold cross-validation	accuracy: 82.9% (3 emotions)
Melgarejo-2014 [50]	Wi-Fi RSSI and Phase	kNN, DTW	In-Vehicle Hand Gesture Recognition	5-fold cross-validation	accuracy: 89% (14 gestures)
ViHOT [71]	Wi-Fi CSI	DTW, Pattern Matching	Driver Head Tracking	various practical scenarios	median error: 4°-10° (3 drivers)
WiCAR [68]	Wi-Fi CSI	CNN, Domain Adaptation	Driver Activity Recognition	64 different domains	accuracy: 94.3% (8 activities)
WiDriver [17]	Wi-Fi CSI	Finite Automata	Driver Posture Recognition and Motion Detection	90% training, 10% testing	accuracy: 96.8% (11 postures), 90.76% (8 motions)
WiBot [55]	Wi-Fi CSI	kNN, Pattern Matching	Driver Head/Arm Motion Recognition	10-fold cross-validation, 40 participants	accuracy: 94.5% (head vs. arm), 90.5% (5 motions)
Xu-2020 [73]	Wi-Fi CSI	Markov Chain, Dynamic Programming	In-Vehicle Occupancy Counting/Breathing Rate Estimation	up to 4 subjects, 1 scenario	counting: 94%-71%, breathing: 96%-77% (1-4 persons)
RF-CAR [69]	RFID	CNN, Domain Adaptation	Driver Activity Recognition	64 different domains	accuracy: 95% (8 activities)
BreathListener [74]	Acoustic on Smartphone	GAN	Driver Breathing Rate Estimation	10 drivers, 10 smartphones	error: 0.11bpm
SteerTrack [75]	Acoustic on Smartphone	Cross-Correlation	Steering Wheel Tracking	5 drivers, 5 smartphones	steering tracking error: 4.61°

focuses on only the driver. Our target is per-seat occupancy which focuses on not only the driver but also passengers, and *CarOSense* works for both single- and multi-person scenarios.

UWB is coming to vehicles for keyless entry [15, 33, 43, 67], personal devices for spatial awareness [5], and industrial plants for asset tracking [13, 19]. For example, UWB is used for indoor positioning [3], liquid sensing [16], and in-vehicle ranging [54]. Sachs gives a review of the basics, potentials, and applications of UWB sensing [58]. We use the UWB keyless infrastructure for a new application, i.e., per-seat car occupancy sensing.

We demonstrate that UWB and deep learning models can be used for per-seat car occupancy sensing that is accurate and robust in different unknown scenarios and efficient to run on embedded devices.

Multi-sensor fusion is another interesting area to explore as in principle it should improve the robustness of the system. This can be achieved by adding additional sensors or utilizing other already available sensing modalities in the car. Augmenting *CarOSense* with additional sensors will increase the cost of the overall system, while utilizing other already available sensors requires timely access to the sensor data. Currently, in a real-world practical scenario, sensors in a vehicle are provided by different vendors, which may or may not even share the same wired/wireless network. This creates a major limitation as the timely data access and actuation for a given sensor cannot be guaranteed. We leave the development of the multi-sensor fusion solution as future work.

6 CONCLUSION

UWB is coming to personal devices and cars for communication and sensing purposes. An example of sensing applications is using UWB for keyless entry for cars. In this paper, we demonstrate the feasibility of reusing the UWB keyless infrastructure for per-seat car occupancy sensing as an alternative to dedicated solutions based on weight sensors, cameras or radars. The automotive indoor environment is small with rich multi-path and very variable with moving seat positions, animate/inanimate objects at different locations (on seat, car floor, or in trunk), etc. It is impractical to train a model with all possible cases, which makes this problem more challenging. We first run experiments and show that standard machine learning algorithms have low accuracy for untrained scenarios such as unknown car locations, front seats moved, back seats fold down, etc. To address this issue, we propose *CarOSense* for accurate, robust and cost efficient (low training effort and lightweight implementation) per-seat occupancy sensing reusing the UWB keyless infrastructure. *CarOSense* uses a new deep learning model, MaskMIMO, to learn spatial/time features by 2D convolutions and per-seat attentions by a multi-task mask. We implement *CarOSense* as a cross-platform demo and evaluate it in different testing scenarios. Evaluation results show that *CarOSense* is accurate and robust for different scenarios: 94.6% average accuracy using 8 UWB nodes (77.3% using 4 nodes) for leave-one-out test of unknown car locations, and 87.0% average accuracy using 8 nodes (69.4% using 4 nodes) for stress test of unseen scenarios. We also demonstrate that *CarOSense* has low computational resource requirements and can run smoothly in real-time on embedded devices. The CIR processing and model inference is 128 milliseconds for Google Coral and 86 milliseconds for Raspberry Pi.

Car manufactures can use *CarOSense* to provide rear seat belt reminder for additional safety ratings without introducing dedicated sensing hardware. This can substantially reduce fatalities for rear seat occupants, as unbelted rear seat occupants are twice likely to die in vehicle crash as compared to belted ones [8]. Another safety application is providing occupancy information for air bag control for rear seats. It can also assist in providing better personalized experience for driver and passengers by adjusting climate and audio control as per occupancy. We believe *CarOSense* is the first step and it could open up new opportunities for in-vehicle sensing. *CarOSense* can also be used in combination with other potential in-vehicle sensing modules to provide more accurate and robust performance. Assisted by new deep learning models, *CarOSense* can be used for other tasks such as occupancy classification and human activity monitoring.

REFERENCES

[1] National Highway Traffic Safety Administration. 2018. Occupant restraint use in 2016: Results from the NOPUS controlled intersection study. Retrieved February 11, 2020 from https://crashstats.nhtsa.dot.gov/Api/Public/Publication/812463 Report No. DOT HS 812 463.

[2] Kiumi Akingbehin and Akinsola Akingbehin. 2005. Alternatives for Short Range Low Power Wireless Communications. In *Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Network*. IEEE, 320–321. https://doi.org/10.1109/SNPD-SAWN.2005.11

[3] Abdulrahman Alarifi, AbdulMalik Al-Salman, Mansour Alsaleh, Ahmad Alnafessah, Suheer Al-Hadhrami, Mai A. Al-Ammar, and Hend S. Al-Khalifa. 2016. Ultra Wideband Indoor Positioning Technologies: Analysis and Recent Advances. *Sensors* 16, 5 (2016). https://doi.org/10.3390/s16050707

[4] Hyundai Motor America. 2017. Hyundai Motor Announces New Rear Occupant Alert Reducing Child Heat Hazards. Retrieved February 15, 2020 from https://www.hyundainews.com/en-us/releases/2383

[5] Apple. 2019. Ultra Wideband technology comes to iPhone. Retrieved November 15, 2019 from https://www.apple.com/iphone-11/

[6] Apple. 2020. Introducing Car Keys. Retrieved July 7, 2020 from https://developer.apple.com/videos/play/wwdc2020/10006

[7] Ahmad E. Assaad. 2018. Angle Detection for Locating Mobile Terminals. Patent Publication No. WO2019034451.

[8] Governors Highway Safety Association. 2019. Rear Seat Belt Use: Little Change in Four Years, Much More to Do. Retrieved May 1, 2020 from https://www.ghsa.org/resources/RearBeltReport19

[9] Gregory S. Bickel. 2006. Inter/Intra-Vehicle Wireless Communication. Retrieved November 18, 2019 from https://www.cse.wustl.edu/~jain/cse574-06/ftp/vehicular_wireless/

[10] Caaresys. [n.d.]. Caaresys Vehicle Occupancy Sensor. Retrieved February 15, 2020 from https://caaresys.com/

[11] François Chollet. 2020. *Deep Learning with Python*. Manning. https://www.manning.com/books/deep-learning-with-python

[12] Chun-Fu Chuang, Chung-Hsien Yang, and Yu-Hui Lin. 2015. HMM-based driving behavior recognition for in-car control service. In *2015 IEEE International Conference on Consumer Electronics - Taiwan*. IEEE, 258–259. https://doi.org/10.1109/ICCE-TW.2015.7216886

[13] FiRa Consortium. 2019. What UWB Does. Retrieved November 18, 2019 from https://www.firaconsortium.org/discover/what-uwb-does

[14] Brian Cooley. 2018. Is technology driving your pet insane? Retrieved February 15, 2020 from https://www.cnet.com/news/is-technology-driving-your-pet-insane/

[15] Nitin Dahad. 2019. NXP Adds Latest Automotive UWB chip as BMW Drives Digital Key 3.0. Retrieved February 15, 2020 from https://www.eetimes.com/nxp-adds-latest-automotive-uwb-chip-as-bmw-drives-digital-key-3-0/

[16] Ashutosh Dhekne, Mahanth Gowda, Yixuan Zhao, Haitham Hassanieh, and Romit Roy Choudhury. 2018. LiquID: A Wireless Liquid IDentifier. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '18)*. ACM, 442–454. https://doi.org/10.1145/3210240.3210345

[17] Shihong Duan, Tianqing Yu, and Jie He. 2018. WiDriver: Driver Activity Recognition System Based on WiFi CSI. *International Journal of Wireless Information Networks* 25, 2 (01 Jun 2018), 146–156. https://doi.org/10.1007/s10776-018-0389-0

[18] Electrek. 2018. Tesla stolen via keyfob hack. Retrieved May 1, 2020 from https://electrek.co/2018/10/21/tesla-stealing-video-keyfob-hack/

[19] Engadget. 2019. Sony and Samsung resurrect ultra-wideband to improve location tracking. Retrieved February 11, 2020 from https://www.engadget.com/2019/08/01/sony-and-samsung-resurrect-ultra-wideband-to-improve-location-tr/

[20] Farzan Erlik Nowruzi, Wassim A. El Ahmar, Robert Laganiere, and Amir H. Ghods. 2019. In-Vehicle Occupancy Detection With Convolutional Networks on Thermal Images. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 941–948. https://doi.org/10.1109/CVPRW.2019.00124

[21] Raspberry Pi Foundation. 2019. Raspberry Pi 4 Tech Specs. Retrieved January 28, 2020 from https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/

[22] Aurélien Francillon, Boris Danev, and Srdjan Capkun. 2011. Relay Attacks on Passive Keyless Entry and Start Systems in Modern Cars. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[23] Lex Fridman. 2018. Human-Centered Autonomous Vehicle Systems: Principles of Effective Shared Autonomy. (2018). arXiv:1810.01835

[24] Kyle Golsch and Steven Sute. 2017. Passive Entry/Passive Start Systems and Methods for Vehicles. US Patent 10328898.

[25] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

[26] Google. 2020. Coral Dev Board datasheet. Retrieved January 28, 2020 from https://coral.ai/docs/dev-board/datasheet/

[27] Google. 2020. TensorFlow Lite guide. Retrieved February 11, 2020 from https://www.tensorflow.org/lite/guide

[28] Charles A Gray, James F Patterson, and Thomas Fischer. 2002. Vehicle Seat Occupant Characterization Method Including Ultralight Child Seat Detection. US Patent 6,438,476.

[29] Abdulmalik Humayed and Bo Luo. 2015. Cyber-Physical Security for Smart Cars: Taxonomy of Vulnerabilities, Threats, and Attacks. In *Proceedings of the ACM/IEEE Sixth International Conference on Cyber-Physical Systems (ICCPS '15)*. ACM, 252–253. https://doi.org/10.1145/2735960.2735992

[30] IEEE. 2016. IEEE Standard for Low-Rate Wireless Networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)* (April 2016), 1–709. https://doi.org/10.1109/IEEESTD.2016.7460875

[31] IEEE. 2019. IEEE 802.15 WPAN Task Group 4z Enhanced Impulse Radio. Retrieved February 11, 2020 from http://www.ieee802.org/15/pub/TG4z.html

[32] Texas Instruments. 2018. Vehicle Occupant Detection Reference Design. Retrieved February 13, 2020 from https://www.ti.com/lit/ug/tidue95a/tidue95a.pdf

[33] HARMAN International. 2020. SmartKey with BLE & UWB At Your Service. Retrieved February 11, 2020 from https://car.harman.com/solutions/telematics/smartkey

[34] ITU-R. 2006. Characteristics of Ultra-Wideband Technology. Retrieved November 18, 2019 from https://www.itu.int/dms_pubrec/itu-r/rec/sm/R-REC-SM.1755-0-200605-I!!PDF-E.pdf

[35] Jedidi Kamouaa. 2013. Turn-Key Passive Entry/Passive Start Solution. Retrieved February 15, 2020 from http://ww1.microchip.com/downloads/en/DeviceDoc/Article_AC10_Turn-Key-Passive-Entry.pdf

[36] James H. Fosterand Duncan Kerr. 2017. System and Method for Vehicle Authorization. US Patent Application 20190039570.
[37] Kia. 2020. Rear Occupant Alert for Kia Telluride. Retrieved February 15, 2020 from https://www.kia.com/us/en/vehicle/telluride/2020
[38] KidsAndCars.org. 2019. Examples of Available Technology to Prevent Hot Car Deaths. Retrieved February 15, 2020 from https://www.kidsandcars.org/2019/09/20/examples-of-available-technology-to-prevent-hot-car-deaths/
[39] Whui Kim, Woo-Sung Jung, and Hyun Kyun Choi. 2019. Lightweight Driver Monitoring System Based on Multi-Task Mobilenets. *Sensors* 19, 14 (2019). https://doi.org/10.3390/s19143200
[40] Manikanta Kotaru, Guy Satat, Ramesh Raskar, and Sachin Katti. 2019. Light-Field for RF. (2019). arXiv:1901.03953
[41] Peter Gorm Larsen and Thierry Mousel. 2019. Advanced Rear Seat Sensing - Further Improving Occupant Safety, Using RF Technology. In *Proc. of 26th Intl. Tech. Conf. on the Enhanced Safety of Vehicles (ESV)*.
[42] Brent M. Ledvina, Robert W. Brumley, and Sriram Hariharan. 2018. Mobile Device for Communicating and Ranging with Access Control System for Automatic Functionality. US Patent Application 20190135229.
[43] Brent M Ledvina, Robert W Brumley, Robert William Mayor, William J Bencze, Alejandro J Marquez, Shang-Te Yang, Xu Chen, Mohit Narang, and Indranil S Sen. 2019. Enhanced Automotive Passive Entry. US Patent 10,285,013.
[44] Seong Kyu Leem, Faheem Khan, and Sung Ho Cho. 2017. Vital Sign Monitoring and Mobile Phone Usage Detection Using IR-UWB Radar for Intended Use in Car Crash Prevention. *Sensors* 17, 6 (2017). https://doi.org/10.3390/s17061240
[45] Decawave Ltd. 2017. DW1000 User Manual. Retrieved November 26, 2019 from https://www.decawave.com/dw1000/usermanual/
[46] Vayyar Imaging LTD. [n.d.]. Vayyar Automotive Radar Sensor. Retrieved February 15, 2020 from https://vayyar.com/auto
[47] Dawei Luo, Jianbo Lu, and Gang Guo. 2017. An Indirect Occupancy Detection and Occupant Counting System Using Motion Sensors. In *WCX™ 17: SAE World Congress Experience*. SAE International. https://doi.org/10.4271/2017-01-1442
[48] Yongsen Ma, Gang Zhou, and Shuangquan Wang. 2019. WiFi Sensing with Channel State Information: A Survey. *ACM Comput. Surv.* 52, 3, Article 46 (June 2019), 36 pages. https://doi.org/10.1145/3310194
[49] Rami Mazraani, Manuel Saez, Leonardo Govoni, and Daniel Knobloch. 2017. Experimental Results of a Combined TDOA/TOF Technique for UWB based Localization Systems. In *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*. 1043–1048. https://doi.org/10.1109/ICCW.2017.7962796
[50] Pedro Melgarejo, Xinyu Zhang, Parameswaran Ramanathan, and David Chu. 2014. Leveraging Directional Antenna Capabilities for Fine-grained Gesture Recognition. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14)*. ACM, 541–551. https://doi.org/10.1145/2632048.2632095
[51] Thierry Mousel, Keisuke Ueda, and Masaki Takahashi. 2015. Advanced Seat Belt Reminder System for Rear Seat Passengers. In *Proc. of 24th Intl. Tech. Conf. on the Enhanced Safety of Vehicles (ESV)*.
[52] MSI. 2019. GS75 Stealth Laptops Specs. Retrieved January 28, 2020 from https://www.msi.com/Laptop/GS75-Stealth-9SX/Specification
[53] Global NCAP. 2017. Creating a Global Market for Vehicle Safety. Retrieved February 11, 2020 from http://www.globalncap.org/wp-content/uploads/2017/06/Market-for-Vehicle-Safety.pdf
[54] Ales Prokes, Tomas Mikulasek, Jiri Blumenstein, Christoph F. Mecklenbrauker, and Thomas Zemen. 2015. Intra-vehicle Ranging in Ultra-Wide and Millimeter Wave Bands. In *2015 IEEE Asia Pacific Conference on Wireless and Mobile (APWiMob)*. 246–250. https://doi.org/10.1109/APWiMob.2015.7374969
[55] Muneeba Raja, Viviane Ghaderi, and Stephan Sigg. 2018. WiBot! In-Vehicle Behaviour and Gesture Recognition Using Wireless Network Edge. In *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. 376–387. https://doi.org/10.1109/ICDCS.2018.00045
[56] Muneeba Raja and Stephan Sigg. 2017. RFexpress! - Exploiting the wireless network edge for RF-based emotion sensing. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. 1–8. https://doi.org/10.1109/ETFA.2017.8247609
[57] Andreas Riener, Myounghoon Jeon, Ignacio Alvarez, and Anna K. Frison. 2017. Driver in the Loop: Best Practices in Automotive Sensing and Feedback Mechanisms. In *Automotive User Interfaces: Creating Interactive Experiences in the Car*, Gerrit Meixner and Christian Müller (Eds.). Springer International Publishing, Cham, 295–323. https://doi.org/10.1007/978-3-319-49448-7_11
[58] Jürgen Sachs. 2012. Ultra-Wideband Sensing - An Overview. In *Handbook of Ultra-Wideband Short-Range Sensing*. John Wiley & Sons, Ltd, Chapter 1, 1–30. https://doi.org/10.1002/9783527651818.ch1
[59] Albin Sellergren. 2018. *Intra-Vehicle Connectivity: Case Study and Channel Characterization.* Master's thesis. Uppsala University.
[60] NXP Semiconductors. 2019. NXP and VW share the wide possibilities of Ultra-Wideband's (UWB) fine ranging capabilities. Retrieved November 15, 2019 from https://media.nxp.com/news-releases/news-release-details/nxp-and-vw-share-wide-possibilities-ultra-widebands-uwb-fine
[61] Manya Sleeper, Sebastian Schnorf, Brian Kemler, and Sunny Consolvo. 2015. Attitudes toward Vehicle-Based Sensing and Recording. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. ACM, 1017–1028. https://doi.org/10.1145/2750858.2806064
[62] Karly A. Smith, Clément Csech, David Murdoch, and George Shaker. 2018. Gesture Recognition Using mm-Wave Sensor for Human-Car Interface. *IEEE Sensors Letters* 2, 2 (2018), 1–4. https://doi.org/10.1109/LSENS.2018.2810093

[63] Hermann Sterner, Wolfgang Aichholzer, and Matthias Haselberger. 2012. Development of an Antenna Sensor for Occupant Detection in Passenger Transportation. *Procedia Engineering* 47 (2012), 178 – 183. https://doi.org/10.1016/j.proeng.2012.09.113 26th European Conference on Solid-State Transducers, EUROSENSOR 2012.

[64] Shane Tuohy, Martin Glavin, Ciarán Hughes, Edward Jones, Mohan Trivedi, and Liam Kilmartin. 2015. Intra-Vehicle Networks: A Review. *IEEE Transactions on Intelligent Transportation Systems* 16, 2 (April 2015), 534–545. https://doi.org/10.1109/TITS.2014.2320605

[65] RTLS UWB. 2019. UWB Technology in Real-time Location Systems. Retrieved November 18, 2019 from https://rtlsuwb.com/

[66] Mickael Viot. 2014. Automotive Security: Why UWB Measures Up. Retrieved February 15, 2020 from https://www.decawave.com/wp-content/uploads/2018/08/embedded_systems_engineering_sept_oct_2014_1.pdf

[67] Volkswagen. 2019. Realtime safety with UWB. Retrieved February 15, 2020 from https://www.volkswagen-newsroom.com/en/stories/realtime-safety-with-uwb-5438

[68] Fangxin Wang, Jiangchuan Liu, and Wei Gong. 2019. WiCAR: WiFi-based In-car Activity Recognition with Multi-adversarial Domain Adaptation. In *Proceedings of the International Symposium on Quality of Service (IWQoS '19)*. ACM, Article 19, 10 pages. https://doi.org/10.1145/3326285.3329054

[69] Fangxin Wang, Jiangchuan Liu, and Wei Gong. 2020. Multi-Adversarial In-Car Activity Recognition using RFIDs. *IEEE Transactions on Mobile Computing* (2020). https://doi.org/10.1109/TMC.2020.2977902

[70] Wikipedia. [n.d.]. Hearing range. Retrieved February 15, 2020 from https://en.wikipedia.org/wiki/Hearing_range

[71] Xiufeng Xie, Kang G. Shin, Hamed Yousefi, and Suining He. 2018. Wireless CSI-based Head Tracking in the Driver Seat. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies (CoNEXT '18)*. ACM, 112–125. https://doi.org/10.1145/3281411.3281414

[72] Yaxiong Xie, Zhenjiang Li, and Mo Li. 2015. Precise Power Delay Profiling with Commodity WiFi. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking (MobiCom '15)*. ACM, 53–64. https://doi.org/10.1145/2789168.2790124

[73] Qinyi Xu, Beibei Wang, Feng Zhang, Deepika Sai Regani, Fengyu Wang, and K. J. Ray Liu. 2020. Wireless AI in Smart Car: How Smart a Car Can Be? *IEEE Access* 8 (2020), 55091–55112. https://doi.org/10.1109/ACCESS.2020.2978531

[74] Xiangyu Xu, Jiadi Yu, Yingying Chen, Yanmin Zhu, Linghe Kong, and Minglu Li. 2019. BreathListener: Fine-grained Breathing Monitoring in Driving Environments Utilizing Acoustic Signals. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '19)*. ACM, 54–66. https://doi.org/10.1145/3307334.3326074

[75] Xiangyu Xu, Jiadi Yu, Yingying Chen, Yanmin Zhu, and Minglu Li. 2018. SteerTrack: Acoustic-Based Device-Free Steering Tracking Leveraging Smartphones. In *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 1–9. https://doi.org/10.1109/SAHCN.2018.8397115

[76] Mengyao Yang, Xiuzhu Yang, Lei Li, and Lin Zhang. 2018. In-Car Multiple Targets Vital Sign Monitoring Using Location-Based VMD Algorithm. In *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*. 1–6. https://doi.org/10.1109/WCSP.2018.8555567

[77] Hubert Zangl, Thomas Bretterklieber, Dirk Hammerschmidt, and Tobias Werth. 2008. Seat Occupancy Detection Using Capacitive Sensing Technology. In *SAE World Congress & Exhibition*. SAE International. https://doi.org/10.4271/2008-01-0908

[78] Mingmin Zhao, Yonglong Tian, Hang Zhao, Mohammad Abu Alsheikh, Tianhong Li, Rumen Hristov, Zachary Kabelac, Dina Katabi, and Antonio Torralba. 2018. RF-based 3D Skeletons. In *Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (SIGCOMM '18)*. ACM, 267–281. https://doi.org/10.1145/3230543.3230579