Journal of
Sensor and Actuator Networks

MDPI

Article

AI-Based Pedestrian Detection and Avoidance at Night Using Multiple Sensors

Hovannes Kulhandjian [1,*], Jeremiah Barron [1], Megan Tamiyasu [1], Mateo Thompson [1] and Michel Kulhandjian [2]

[1] Department of Electrical and Computer Engineering, California State University, Fresno, CA 93740, USA; jjbarron@mail.fresnostate.edu (J.B.); mtamiyasu427@mail.fresnostate.edu (M.T.); mateothompson27@mail.fresnostate.edu (M.T.)
[2] Department of Electrical and Computer Engineering, Rice University, Houston, TX 77005, USA; michel.kulhandjian@rice.edu
* Correspondence: hkulhandjian@mail.fresnostate.edu

Abstract: In this paper, we present a pedestrian detection and avoidance scheme utilizing multi-sensor data collection and machine learning for intelligent transportation systems (ITSs). The system integrates a video camera, an infrared (IR) camera, and a micro-Doppler radar for data acquisition and training. A deep convolutional neural network (DCNN) is employed to process RGB and IR images. The RGB dataset comprises 1200 images (600 with pedestrians and 600 without), while the IR dataset includes 1000 images (500 with pedestrians and 500 without), 85% of which were captured at night. Two distinct DCNNs were trained using these datasets, achieving a validation accuracy of 99.6% with the RGB camera and 97.3% with the IR camera. The radar sensor determines the pedestrian's range and direction of travel. Experimental evaluations conducted in a vehicle demonstrated that the multi-sensor detection scheme effectively triggers a warning signal to a vibrating motor on the steering wheel and displays a warning message on the passenger's touchscreen computer when a pedestrian is detected in potential danger. This system operates efficiently both during the day and at night.

Keywords: machine learning; pedestrian detection; accident prevention; intelligent transportation systems

Citation: Kulhandjian, H.; Barron, J.; Tamiyasu, M.; Thompson, M.; Kulhandjian, M. AI-Based Pedestrian Detection and Avoidance at Night Using Multiple Sensors. *J. Sens. Actuator Netw.* **2024**, *13*, 34. https://doi.org/10.3390/jsan13030034

Academic Editor: Lei Shu

Received: 7 April 2024
Revised: 29 May 2024
Accepted: 6 June 2024
Published: 14 June 2024

1. Introduction

Pedestrian fatalities in traffic crashes in the United States highlight a critical issue, with one pedestrian killed every 88 min, totaling over 16 daily and nearly 115 weekly. In the first nine months of 2020, motor vehicle traffic fatalities rose by 4.5%, with an estimated 28,190 deaths [1]. In 2019, over 6500 pedestrians were killed, the highest annual number recorded, with more than 100,000 injuries [2]. Notably, 75% of pedestrian fatalities occurred in low-light conditions. California saw around 430 pedestrian deaths in the first half of 2018, with a 40% increase in pedestrian fatalities in the first half of 2021 compared to 2020 [3,4]. Research suggests that automatic emergency braking systems with pedestrian detection could prevent up to 5000 vehicle-pedestrian crashes annually, including 810 fatal collisions [5,6]. Advancements in thermal infrared (IR) cameras show potential for improving pedestrian safety. However, current studies on pedestrian detection and avoidance, including [7–11], are in the early stages and lack the integration of multiple sensors with advanced machine learning (ML) for real-time detection and avoidance. Our research addresses this gap by developing AI-based tools using state-of-the-art ML techniques and sensor data fusion, focusing on nighttime detection. By combining data from thermal IR, radar, and visible cameras with advanced ML algorithms, our system aims to detect and prevent pedestrian collisions in real time under various conditions. This integrated mechanism, suitable for both day and night, could also be used in autonomous vehicles.

Related Work: In the existing body of research, various methodologies have been proposed for different aspects of pedestrian detection and safety systems. Al Sobbahi and Tekli [12] conducted comparative studies on deep-learning models used for low-light image enhancement. They presented a comparative analysis of the impact on object detection and classification performance when image enhancement is applied. Xiao et al. [13] proposed two-stage and single-stage pedestrian detection frameworks. In the two-stage scheme, approximate valid regional candidates were first generated and then refined through another sub-network. The single-stage scheme, in contrast, omitted the regional candidates and directly regressed the predefined areas to achieve computational efficiency. These two frameworks were built on different network models. Gonzales et al. [14] assessed the accuracy gain of various pedestrian models when trained with images in the far infrared spectrum. Specifically, they compared detection accuracy on test images recorded during the day and night, training on plain color images, infrared images, or both combined. Lim et al. in [9] focused on discriminating stationary targets in traffic monitoring radar systems, although their approach did not center explicitly on pedestrian detection. Cao et al. in [10] introduced a hierarchical reinforcement and imitation learning (H-ReIL) method, integrating imitation learning (IL) for discrete driving modes and reinforcement learning (RL) for efficient mode-switching strategies. Jain et al. [15] developed a multimodal pedestrian detection system for crowded scenes with reduced visibility due to daytime or weather conditions. They used heuristics and a deep convolutional neural network technique. Their neural network model incorporates a long short-term memory (LSTM) architecture, with hyperparameters adjusted using the shark smell optimization algorithm. Fukui et al. [16] proposed a vision-based pedestrian detection framework based on a CNN that achieves high accuracy across various fields. The authors introduced random dropout and an ensemble inference network to the training and classification processes to improve detection accuracy. Luo et al. in [17] proposed a pedestrian detection scheme utilizing active and passive night vision in different regions of the electromagnetic spectrum, discussing the capabilities of each approach. Similarly, Han and Song in [18] presented a night-vision pedestrian detection system for automatic emergency braking (AEB) using near-infrared cameras and specific algorithms for pedestrian discrimination. Fu in [19] proposed a pedestrian detection method based on the "three-frame difference method," introducing modifications to enhance system accuracy. We summarize and compare our contribution with existing studies in Table 1.

Despite the breadth of these studies, none of them ventured into actual implementation or employed a fusion of multi-sensor data coupled with machine learning techniques specifically for pedestrian detection and driver alert systems. This research aims to bridge this gap by focusing on the practical implementation of a comprehensive system that integrates data from various sensors, leveraging advanced machine learning to detect pedestrians and provide real-time alerts to drivers when pedestrians are detected.

The core innovation of this research lies in the fusion of three distinct sensors—a thermal infrared camera, a radar sensor, and a visible camera—augmented by advanced machine learning (ML) techniques for pedestrian detection and avoidance. This exploration aims to pioneer AI-driven tools for drivers, potentially saving lives. Our research focuses on optimizing pedestrian detection, particularly in low-light conditions, harnessing the information gleaned from these sensors and advanced ML algorithms in real time. By leveraging this multidimensional data, our system can make informed decisions across diverse road conditions, day or night. The envisaged system could seamlessly integrate into smart vehicle setups, offering real-time pedestrian detection and alert mechanisms. It notifies drivers through wheel vibration and dashboard messages to prevent potential collisions. With live experimentation conducted in a vehicle setting, our system achieved an impressive average accuracy of 98%. Utilizing a deep learning algorithm, it effectively identifies pedestrian presence, both day and night, ensuring timely alerts within a real-time monitoring framework.

The rest of the paper is organized as follows: In Section 2, we describe the system overview, followed by pedestrian detection using a video camera, IR camera, and a radar sensor in Sections 3–5, respectively. After illustrating the experimentation results in Section 6, we draw the main conclusions in Section 7.

Table 1. Gap analysis and contributions of this work with respect to the existing studies in the literature.

Article	Features	Pedestrian Detection	Prototype	Using RGB, IR, Radar	Data Fusion	Low-Light Condition
Lim et al. [9]	Discriminating stational targets in traffic monitoring radar systems.			Radar		
Cao et al. [10]	Hierarchical reinforcement and imitation learning (H-ReIL).			RGB		
Sobbahi and Tekli [12]	Low-light image enhancement and object detection.			RGB		✓
Xiao et al. [13]	Concentrates on occlusion and multi-scale pedestrian identification challenges.	✓		RGB		
Gonzales et al. [14]	Assess the accuracy gain of various pedestrian models.	✓		RGB, Far IR		✓
Jain et al. [15]	Multimodal pedestrian detection for crowded scenes.	✓		RGB		✓
Fukui et al. [16]	Vision-based pedestrian detection framework based on CNN.	✓		RGB	✓	
Luo et al. [17]	Pedestrian detection utilizing active and passive night vision.	✓		Near IR		✓
Han and Song [18]	Night-vision pedestrian detection system for automatic emergency breaking via infrared cameras.	✓		Near IR		✓
Fu [19]	Pedestrian detection based on three-frame difference method.	✓		RGB		
This article	**Thermal, visible image and radar fusion and deep learning techniques for low-light pedestrian detection.**	✓	✓	**RGB, IR, Radar**	✓	✓

2. System Overview

2.1. System Design

This system serves as an alert mechanism for drivers when a pedestrian is at risk while crossing the street. It integrates machine learning with a video camera, infrared (IR) camera, and radar sensor to assess road conditions accurately and identify nearby pedestrians. Positioned in the vehicle, the three sensors concurrently scan the road ahead, capturing diverse data types. The wide-view video and IR cameras capture images, while the radar sensor assesses the pedestrian's distance, motion direction, and speed. Once a pedestrian is identified in the images, a computer script automatically crops and formats these images to train respective deep convolutional neural network models—a machine learning algorithm. Trained using labeled data from RGB and IR camera images alongside micro-Doppler

signals, these models predict pedestrian behavior. They operate on a Raspberry Pi paired with a Coral USB Accelerator to expedite real-time processing. Over a 60-second period, continuous detections from all three sensors accumulate. These sensor-derived prediction models are relayed to the driver in real time through the touchscreen display. If the system detects a potential pedestrian hazard, it activates a vibration motor integrated with the steering wheel, promptly alerting the driver and mitigating the risk of collision. Figure 1 illustrates the hardware setup, depicting the sensors capturing images and radar signals, the pedestrian detection algorithm in action, and the subsequent driver alert through a warning message and steering wheel vibration.

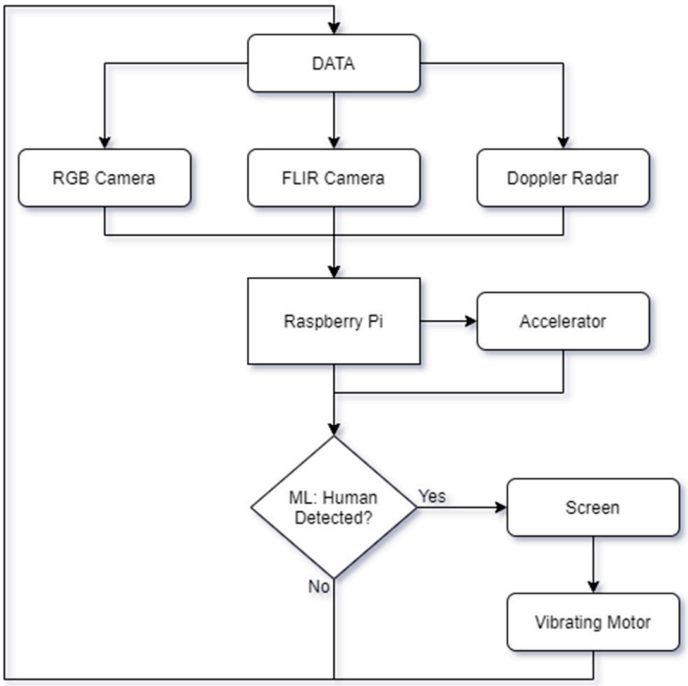


Figure 1. Block diagram of the pedestrian detection hardware design.

Figure 2 presents the pedestrian detection software flow diagram, outlining the process from sensor data acquisition to real-time pedestrian detection. Initially, data from the RGB and FLIR cameras are channeled into a deep convolutional neural network (DCCN) architecture, trained using TensorFlow Lite—an open-source framework for on-device deep learning inference. Once the models are trained, they can be deployed onto an embedded computer, facilitating real-time pedestrian detection. The continuously gathered real-time sensor data are processed through these pre-trained models, consistently assessing for pedestrian presence. If a pedestrian is identified, the system promptly alerts the driver. Otherwise, it continues data collection and model testing in subsequent cycles. Key equipment includes a video camera, FLIR IR camera, radar sensor, micro-computer, vehicle, and an alert system. The alert system incorporates a vibration motor connected to the steering wheel to notify the driver. The machine learning model is initially trained in MATLAB and later integrated into Python for use with TensorFlow. Sensor placement involves mounting some inside and others outside the vehicle to optimize data collection.

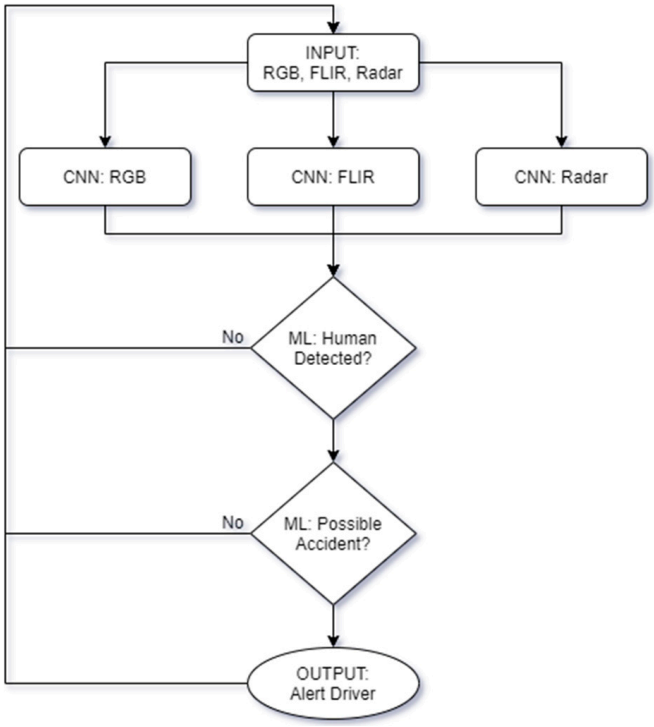


Figure 2. Block diagram of the pedestrian detection software design.

2.2. Multi-Sensor Data Fusion Algorithm

To enhance the accuracy of pedestrian detection, various fusion methods can be employed, including input fusion, early fusion, mid-fusion, late fusion, and a classifier output ensemble. Information fusion is performed before feeding the inputs to the model. Early-, mid-, and late-stage fusion occurs at the initial, intermediate, and fully connected layers of the neural network, respectively. Additionally, classifier ensemble methods offer several approaches to combining classifier outputs. Classifier outputs can be either continuous values or labels. Let us denote the classifier label output as $y_n(x)$, representing the decision of the n-th classifier on the test samples x, $n = 1, \ldots, N$, and l is the number of possible labels. Majority voting is a common scheme where the combined decision is the label predicted by the majority of classifiers. This can be expressed as the following:

$$\hat{y}(\mathbf{x}) = \underset{y_l \in \Omega}{\operatorname{argmax}} \sum_{n} g(y_n(\mathbf{x}),\ y_l), \tag{1}$$

where $g()$ is an indicator is the indicator function that equals 1 if $y_n(x) = y_l$ and 0 otherwise. Another scheme, called unanimity voting, requires all classifiers to agree on the label for a combined decision to be made; otherwise, the input x is rejected. In scenarios where classifiers within the ensemble have varying levels of accuracy, weighted majority voting can be used to improve the final decision. In this method, each classifier is assigned a weight based on its accuracy, and these weights influence the final decision. The combined decision is the label that receives the highest weighted vote. Let us denote the weight of the n-th classifier as w_n. The weighted majority voting can be expressed as the following:

$$\hat{y}(\mathbf{x}) = \underset{y_l \in \Omega}{\operatorname{argmax}} \sum_{n} w_n g(y_n(\mathbf{x}),\ y_l). \tag{2}$$

This method takes into account the different accuracies of the classifiers, aiming to enhance the overall performance of the ensemble.

Each classifier's output is weighted proportionally to its accuracy, with w_n being the assigned weight. This weight can be either fixed or dynamically determined based on the specific input data. One non-linear combination algorithm is the Borda count, which

is based on ranking. In this method, each classifier ranks the classes according to their likelihood of being correct. The combined decision is made by summing the accumulated scores for each label.

In the Borda count method, if there are b possible labels, a classifier assigns a rank r (with 1 being the highest rank and b being the lowest) to each label. The scores are then accumulated across all classifiers. The final decision is the label with the highest total score.

In cases where classifier outputs are continuous values, voting approaches may not fully exploit all available information. Instead, mathematical combination functions can be utilized to integrate these continuous outputs more effectively. Commonly employed functions include the following:

Averaging: This method calculates the average of the continuous outputs from all classifiers. The final decision is based on the average value:

$$\hat{y}(\mathbf{x}) = \frac{1}{N}\sum_{n=1}^{N} y_n(\mathbf{x}), \tag{3}$$

Weighted Averaging: Similar to simple averaging, but each classifier's output is weighted according to its accuracy:

$$\hat{y}(\mathbf{x}) = \frac{\sum_{n=1}^{N} w_n y_n(x)}{\sum_{n=1}^{N} w_n}, \tag{4}$$

where w_n is the weight assigned to the n-th classifier.

Product Rule: This method multiplies the continuous outputs of all classifiers. It is useful when the outputs represent probabilities:

$$\hat{y}(\mathbf{x}) = P(y = y_l|\mathbf{x}) = \prod_{n=1}^{N} P_n(y = y_l|\mathbf{x}). \tag{5}$$

Median: The median of the continuous outputs from all classifiers is used as the final decision. This approach is robust to outliers:

$$\hat{y}(\mathbf{x}) = \text{median}_{\{n=1,\ldots,N\}} y_n(\mathbf{x}). \tag{6}$$

Maximum/Minimum: The maximum or minimum value among the continuous outputs can be used, depending on the context of the problem (e.g., risk assessment, anomaly detection):

$$\hat{y}(\mathbf{x}) = \max(y_1(x),\ y_2(x),\ \ldots,\ y_N(x)), \tag{7}$$

or

$$\hat{y}(\mathbf{x}) = \min(y_1(x),\ y_2(x),\ \ldots,\ y_N(x)). \tag{8}$$

Note that these combiners are called non-trainable, because once the individual classifiers are trained, their outputs can be fused to produce an ensemble decision, without any further training. On the other hand, trainable combiners are Naive Bayes, weighted majority voting.

The proposed ensemble comprises three individual DCCN models, each trained with three input channels as described in Sections 3–5. Each CNN model functions as an individual learner, providing a classification score c (i.e., a probability) for each class. Here, c = 0 represents the non-pedestrian class, and c = 1 represents the pedestrian class. To combine the probabilities from the three CNN models, we employ a weighted average scheme. Therefore, the multi-sensor data fusion algorithm employs a weighted average approach to enhance the accuracy and reliability of pedestrian detection. The algorithm

integrates data from multiple sensors, assigning weights based on each sensor's reliability under current conditions. Specifically:

Nighttime Weights: A weight of 0.7 is heuristically assigned to thermal IR data and 0.3 to visible-light camera data. This emphasis on thermal IR data is due to its superior performance in low-light conditions.

Daytime Weights: A weight of 0.7 is heuristically assigned to visible-light camera data and 0.3 to thermal IR data, reflecting the better performance of visible-light cameras in well-lit conditions.

This approach ensures that the most reliable sensor data are given higher importance, with thermal IR prioritized at night and visible-light cameras during the day. This weighting strategy is based on our analysis, which indicates that thermal IR cameras excel in night detection, while visible-light cameras are more effective during the day.

2.3. Deep Convolutional Neural Network Design

In this project, a deep convolutional neural network (DCNN) was developed for image classification tasks using both RGB and IR camera inputs. The process begins with manually cropping the images to 100 × 120 pixels. The cropped images are processed through a feature extraction network, starting with an initial convolution layer using eight 20 × 20-pixel filters. The output is passed through a rectified linear unit (ReLU) function and a 2 × 2 max pooling layer [20,21]. This process is repeated with a second convolution layer of 16 filters (10 × 10 pixels), followed by ReLU and max pooling and a third convolution layer of 32 filters (5 × 5 pixels), again followed by ReLU and max pooling. In both convolution and max pooling, the stride indicates the sliding window movement. A stride of 1 means the window moves one pixel at a time, extracting maximum values to preserve crucial features like edges. The classifier network includes a fully connected layer with 100 hidden nodes, producing a softmax output to classify the driver's status [22–24]. The DCNN's final output layer provides a probability distribution for each class based on the input images as shown in Figure 3. The initial learning rate is set to 0.001. Learning rate decay is applied, reducing the learning rate by a factor of 0.1 every 10 epochs to stabilize training. A batch size of 32 is used to balance memory usage and training speed. The Adam optimizer is chosen for its adaptive learning rate capabilities, which help in achieving faster convergence. Categorical cross-entropy is used to handle the classification task effectively.

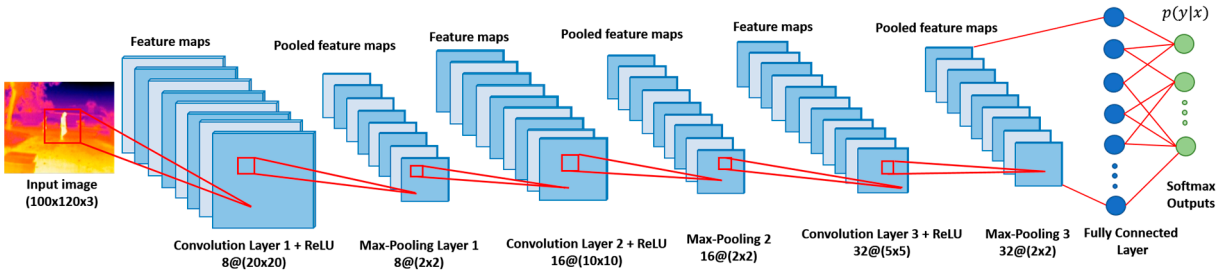


Figure 3. Convolutional neural network architecture for the IR camera image input.

Upon detecting a pedestrian via a maximum ratio combining the data from the three sensors, the algorithm alerts the driver. If no pedestrian is detected, the system continues capturing new images and radar data for further analysis. To ensure the system's ability to capture rapid changes in biometric data, the deep convolutional neural network model was adapted for the Raspberry Pi with the Coral USB Accelerator, as depicted in Figure 4. The model was first converted into a TensorFlow Lite model, optimized for faster inference times and efficient functioning on low-power devices with limited memory. The converted TensorFlow Lite model, initially represented in 32-bit floating-point numbers, underwent post-training quantization to obtain 8-bit fixed-point numbers essential for Edge TPU compatibility. Employing full integer quantization, the model's size decreased by four times, accelerating inference times by a factor of three. The final step involved compiling

the quantized TensorFlow Lite model using the Edge TPU compiler. Several models were compiled to further enhance performance in terms of model accuracy.

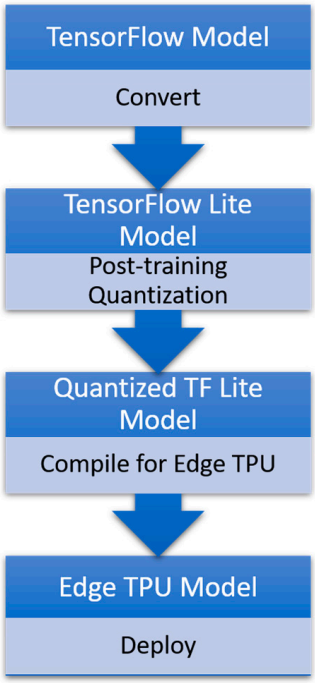


Figure 4. Machine learning model conversion process.

3. Pedestrian Detection Using a Video Camera

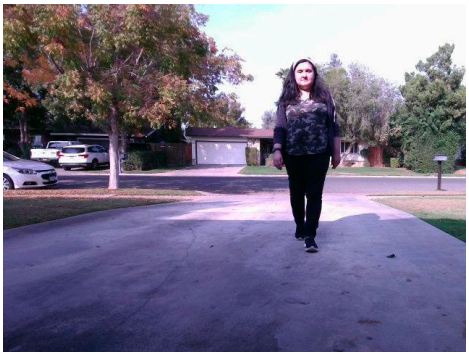
3.1. RGB Camera

Data was collected using an ELP RGB camera, specifically the USB4K02AF-KL100W model. An RGB camera captures images that mirror standard human vision by utilizing red, blue, and green colors. Renowned for its exceptional clarity, the USB4K02AF-KL100W camera boasts a 4K resolution, positioning itself among the market's top performers. Equipped with an array of LED lights, this camera automatically adjusts brightness levels to ensure optimal image capture.

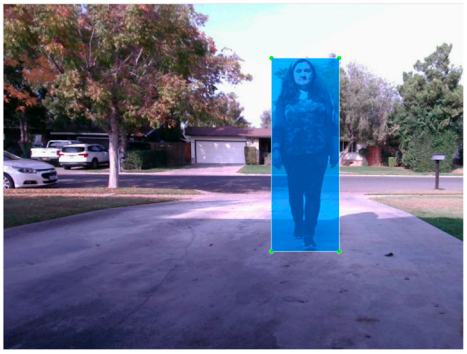
3.2. Data Collection Using the RGB Camera

The RGB camera illustrated in Figure 5 was instrumental in compiling the dataset for configuring a customized machine-learning model aimed at detecting individuals as the model ran. Specifically utilized in daylight scenarios, this camera served a distinct purpose, while the FLIR IR camera, as detailed in Section 4, was responsible for nighttime data collection. Images were captured at various distances, examples of which are showcased in Figure 6. Each image featuring a pedestrian was annotated with bounding boxes, precisely pinpointing their location. This annotation method significantly aided the DCNN algorithm in accurately detecting and classifying pedestrians in the images. A total of 1200 images were gathered, equally divided into 600 images with pedestrians and 600 without. Furthermore, an additional 800 images were sourced from the Teledyne FLIR Thermal Dataset [25], where half contained pedestrians and the other half did not. This accumulation of 2000 images was employed to train and experiment with our ML model.

Figure 5. RGB camera ELP USB4K02AF-KL100W.

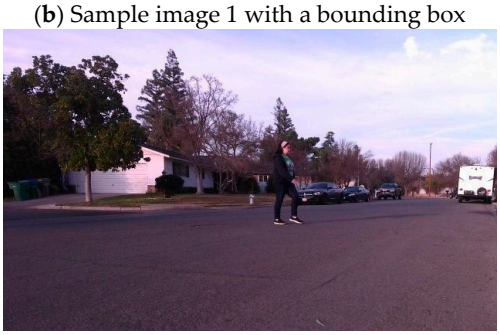


(**a**) Sample image 1



(**b**) Sample image 1 with a bounding box

(**c**) Sample image 2



(**d**) Sample image 3

Figure 6. RGB camera sample images, with a bounding box image on the top right.

The acquired data were instrumental in constructing a specialized machine-learning model utilizing a pre-trained network known as Squeezenet. The evaluation of this trained model demonstrated its high accuracy, as shown by the loss and accuracy data in Figure 7. The loss graph signifies the disparity between the model's prediction and the actual value, which needs to be minimized for optimal accuracy. Meanwhile, the accuracy graph assesses the model's precision by comparing it against the test images utilized for validation. We have also plotted the confusion matrix score for the training results, as shown in Figure 8. MATLAB facilitated the collection of accurate data using the Squeezenet pre-trained network, leveraging standard camera images from the FLIR dataset [26] to differentiate between person images. The camera used to capture those images is a Teledyne FLIR Blackfly S BFS-U3-51S5C (IMX250) camera and a 52.8° HFOV Edmund Optics lens. The training and development of this custom model were conducted on a separate computer before transferring it to the Raspberry Pi 4 as a TFLite model, enabling real-time functionality testing.

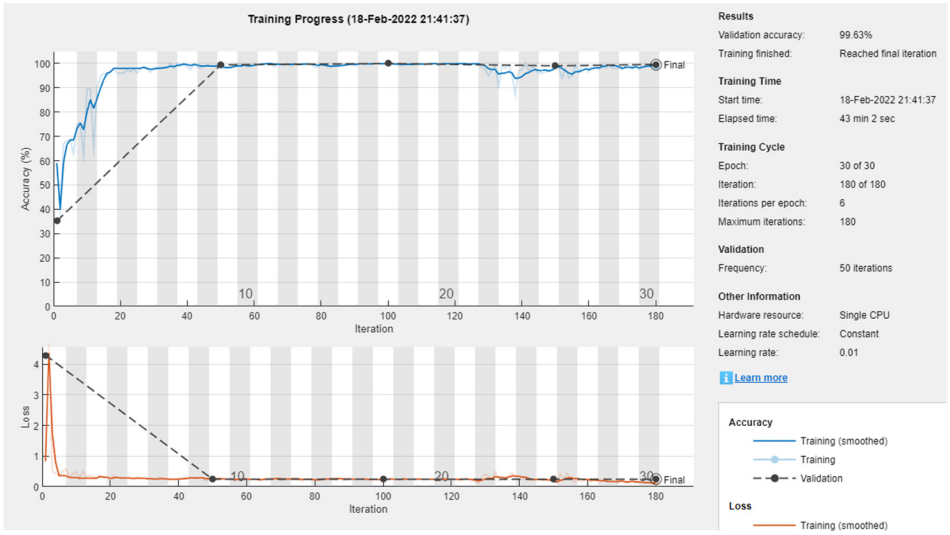


Figure 7. Accuracy and loss training results using the RGB camera.

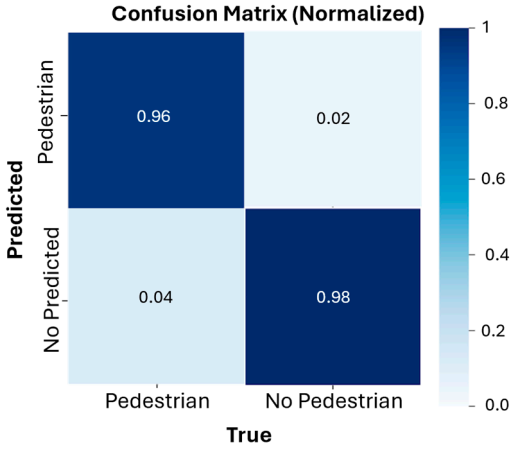


Figure 8. Confusion matrix results using the RGB camera.

Note a loss metric (or loss function) quantifies the difference between the predicted values by a model and the actual values. It measures how well or poorly a model is performing. In our case, cross-entropy loss is used for classification, which is defined as

$$\mathrm{L} = -\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c}) \tag{9}$$

where $\hat{y}_{i,c}$ is a binary indicator (0 or 1) if class label c is the correct classification for observation i, $\hat{y}_{i,c}$ is the predicted probability of observation i being in class c, n is the number of observations, and C is the number of classes.

Accuracy is a metric used to evaluate the performance of a classification model. It is the ratio of correctly predicted instances to the total instances.

$$\text{Accuracy} = \frac{\text{NumberofCorrectPredictions}}{\text{TotalNumberofPredictions}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{10}$$

where TP is the True Positive, TN is the True Negative, FP is the False Positive, and FN is the False Negative.

The F_1 score, also known as the F_1 measure or F_1 value, is a metric used to evaluate the performance of a classification model. It considers both the precision and recall of the

model to provide a balanced assessment of its effectiveness. The formula for calculating the F_1 score is as follows:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (11)$$

Precision is the ratio of true positive (TP) prediction to the total number of positive predictions, calculated as TP/(TP + FP).

Recall, also known as sensitivity or true positive rate, is the ratio of true positive predictions to the total number of actual positive instances, calculated as TP/(TP + FN).

3.3. Performance Results

After meticulously curating the dataset, the deep convolutional neural network underwent training, reaching an impressive validation accuracy of 99.6%, as illustrated in Figure 7. Notably, the validation accuracy begins to stabilize after nine epochs, signifying that the model has reached its optimal performance threshold. In this context, an epoch refers to a full cycle where the entire training dataset is passed forward and backward through the neural network once. Given the limitations of the training dataset, multiple epochs are typically employed, allowing the learning algorithm to iteratively minimize the model's error. The validation accuracy serves as an indicator of the model's practical performance when presented with new samples. On the other hand, the loss metric of the model depicts its responsiveness to training after each iteration or epoch. The loss metric essentially drives the model toward optimization, ensuring that subsequent predictions become increasingly accurate. Ideally, the loss metric should consistently decrease during training, indicating that the model's predictive capabilities are improving. In Figure 8, the confusion matrix results for the RGB camera are presented. The high true positive and true negative rates, coupled with the low false positive and false negative rates, indicate that the model performs effectively in detecting and classifying the data accurately.

3.4. Limitations

The experimental setup had inherent limitations owing to the hardware utilized. Notably, the camera's resolution constraints resulted in reduced accuracy when detecting pedestrians at greater distances. To overcome this limitation, upgrading to a higher-resolution camera with autofocus capabilities could significantly enhance image clarity at longer distances, thereby improving the accuracy of pedestrian detection.

4. Pedestrian Detection Using an IR Camera

4.1. Infrared

In this project, the camera utilizes the long-wave infrared spectrum, operating within the frequencies of 8 µm–14 µm. This range falls within the broader electromagnetic spectrum of infrared, spanning frequencies between 300 GHz–430 THz and wavelengths of 700 nm–1 mm [15].

4.2. Data Collection Using Infrared Camera

Images were collected through the FLIR Lepton 3.5 camera, a compact device smaller than a dime, as depicted in Figure 9. Figure 10 showcases various image samples captured using this camera, highlighting both day and night conditions. These images were then meticulously labeled utilizing LabelImg annotation software [27], an intuitive graphical tool designed in Python with a Qt-based graphical user interface. This software facilitated the identification of individuals within the training and testing sets, as illustrated in Figure 11, which places bounding boxes, precisely pinpointing their location. Notably, all images employed in this study were directly sourced from the camera, without relying on any pre-existing datasets. Approximately 85% of the images were taken during nighttime, while the remainder were captured in daytime settings. The DCNN model was trained using the SSD MobileNet V2 FPNLite pre-trained network. This network was selected for its accelerated processing and its near-native resolution compatibility with the FLIR

camera, minimizing image distortions. While various networks were explored, MobileNet emerged as the optimal choice for the IR dataset, yielding the most promising outcomes.

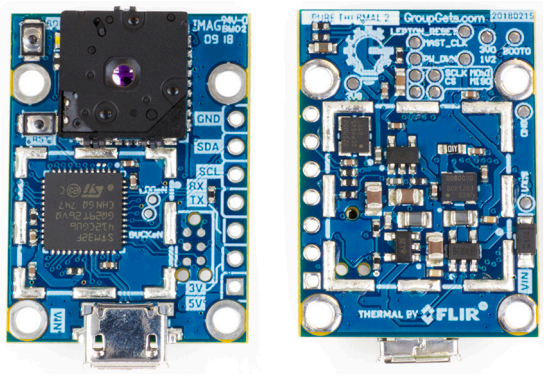


Figure 9. IR Camera: FLIR Lepton 3.5 with Pure Thermal 2 Breakout Board.

Figure 10. IR camera sample images were taken at night.

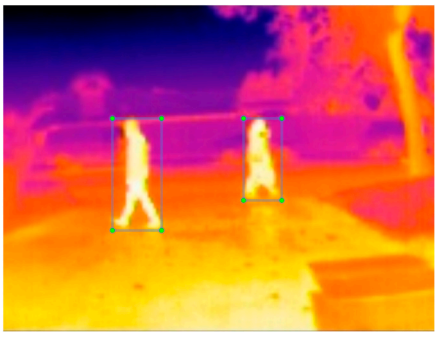


Figure 11. FLIR camera: Object detection (image labeler).

4.3. Performance Results

Once the dataset curation was completed, the deep convolutional neural network underwent training, reaching a validation accuracy of 97.26%, as depicted in Figure 12. Notably, the validation accuracy began to level off after four epochs of training. During testing, the model exhibited successful detection of all human-present images within a 15-meter range of the vehicle. However, accuracy experienced a swift decline beyond this range, particularly when pedestrians exceeded 25 m from the IR camera, leading to algorithmic degradation. In Figure 13, the confusion matrix results for the IR camera are presented. The high true positive and true negative rates, coupled with the low false positive and false negative rates, indicate that the model performs effectively in detecting and classifying the data accurately.

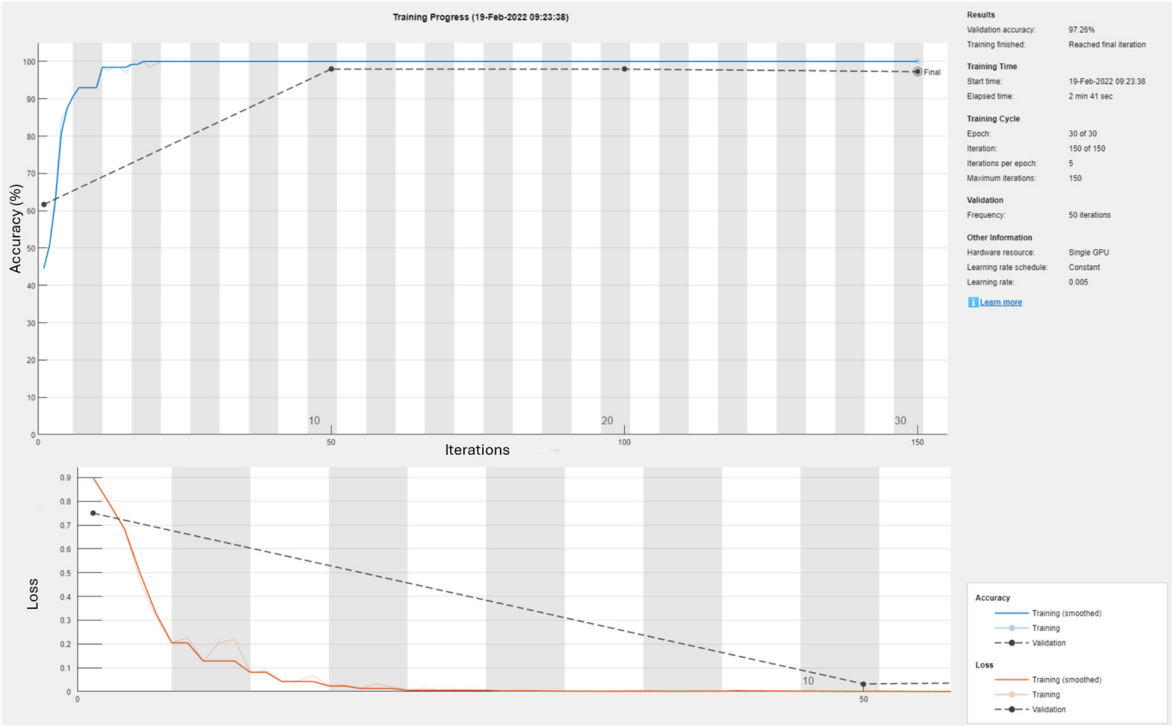


Figure 12. FLIR camera DCNN training and validity accuracy and loss plots.

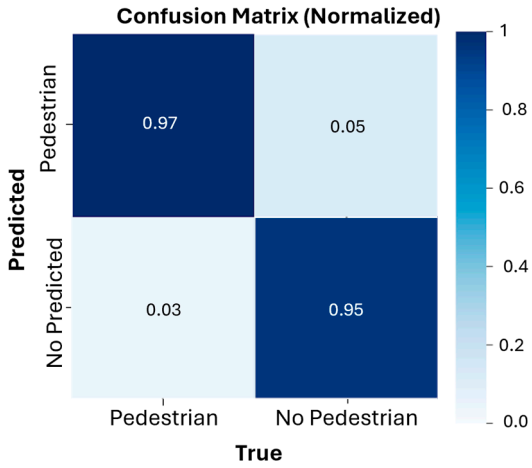


Figure 13. Confusion matrix results using the IR camera.

4.4. Limitations

The FLIR camera's limitation lies in its modest resolution of 160 × 120 pixels. This limited resolution results in poorer image quality, making it challenging for the training network to discern human features for accurate recognition. This becomes particularly pronounced when pedestrians are distant, as they occupy fewer pixels and often appear as bright rectangles. This notably affects the model's performance when detecting pedestrians at longer distances. To overcome this limitation in future iterations, employing a higher-resolution IR camera would enhance the detection of pedestrians, especially at extended distances.

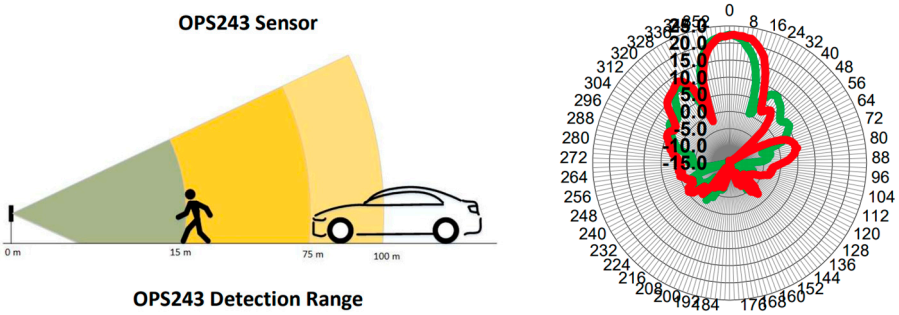
5. Pedestrian Detection Using a Micro-Doppler Radar

5.1. Micro-Doppler Radar Setup

The radar data were captured using OmniPreSense's OPS243-C sensor, shown in Figure 14, a comprehensive short-range radar (SRR) solution offering motion detection,

speed, direction, and range reporting (referenced in Figure 15). Signal processing occurs onboard the sensor, and an easy-to-use application programming interface (API) delivers processed data. It allows adaptable control over reporting formats, sample rates, and module power levels. The sensor gathers information on pedestrian speed, direction, and distance from the vehicle, detecting objects within a range of up to 60 m. Its applications span security, traffic monitoring, drone collision avoidance, robotics, and Internet of Things (IoT) sensor uses. The micro-Doppler radar signals captured were subsequently converted into spectrograms, providing a visual representation of frequency variations in the received signal over time.

Figure 14. OmiPreSence OPS24-C FMCW Doppler sensor.



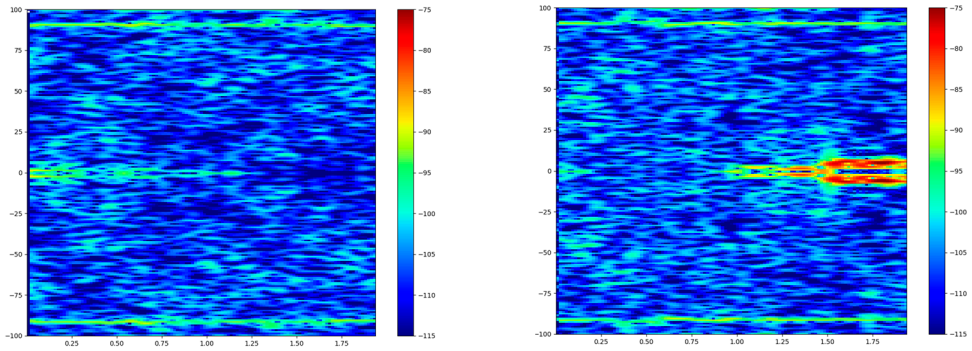
(**a**) (**b**)

Figure 15. OmiPreSence OPS24-C FMCW Doppler detection range and power pattern.

The sensor offers various features, including a detection range from 1 m to 100 m, speed reporting up to 348 mph, range reporting up to 60 m, and a narrow 20-degree beam width (−3 dB), as depicted in Figure 15b's power pattern. Operating within the 24 GHz–24.25 GHz range on the industrial, scientific, and medical (ISM) band, the OPS243 sensor transmits data via USB, UART, RS-232, or Wi-Fi interfaces. This facilitates easy connections to embedded processors (such as Arduino, Raspberry Pi, or PCs) or direct cloud connectivity via Wi-Fi.

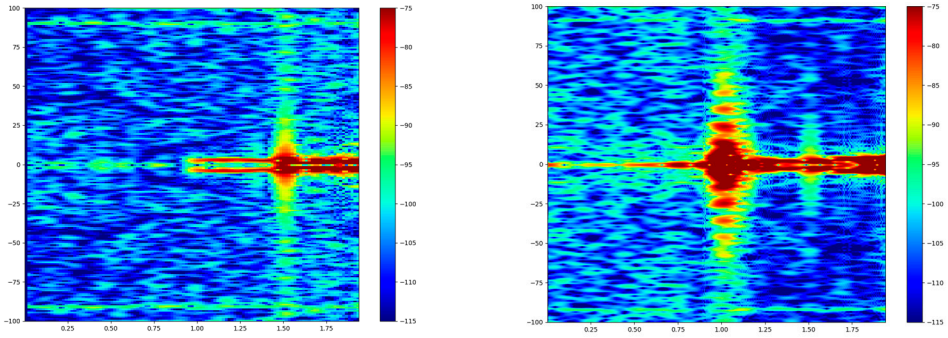
In Figure 15, the radar sensor displays its detection range capabilities on the left, with the power pattern depicted on the right. The power pattern illustrates a narrow 20-degree beam width at −3 dB power. This radar sensor collects data in the form of spectrograms, serving as an input for the DCNN algorithm to predict the driver's status. In Figure 16's spectrogram plot, the vertical axis represents Doppler frequency variations in hertz, while the horizontal axis denotes time in seconds. The redder areas signify increased motion activity, contrasting with the bluer tones indicating minimal variations or motion in front of the radar sensor. Figure 16 portrays scenarios with no pedestrian detected (left) and a pedestrian detected at 20 m (right). In contrast, Figure 17 showcases a pedestrian detected at 10 m (left) and at 5 m (right). Notably, Figure 17's right side exhibits more Doppler

frequency variations due to the pedestrian's closer proximity to the vehicle. In contrast, Figure 16 (right) shows fewer perturbations when the pedestrian is farther away at 20 m.



(**a**) (**b**)

Figure 16. (**a**) No pedestrian detected at 10 m, (**b**) pedestrian detected at 20 m.



(**a**) (**b**)

Figure 17. Pedestrian detected (**a**) at 10 m, (**b**) 5 m.

5.2. Experimental Results Using the Radar Sensor

The OPS243-C FMCW Micro Doppler Radar not only measures the speed of moving objects but also their distance from the Doppler. It operates using two antennas: one transmits an FMCW signal while the other receives the echoed signal. Configured via its onboard API, the OPS243-C FMCW Doppler collected data via a Python script executed on a Raspberry Pi 4. These data aimed to detect human proximity to the vehicle within a range of approximately 25 m, predicting potential accidents using cameras. However, the Micro Doppler's limitations hindered its effectiveness, only detecting human signatures within tens of meters, thwarting the intended use.

Initially intended for spectrogram generation feeding a convolutional neural network, the Doppler's constraints led to merely 2-second spectrogram windows—insufficient for the detection system's response time. Instead, used as a range finder, the radar instantly determined range depths, detecting objects within 10 to 15 m. Objects outside this range were disregarded.

Upon detection within this range, the radar signaled machine learning algorithms to activate the RGB and FLIR cameras. These sensors collectively evaluated detection scores to determine potential accidents with pedestrians. The process involved data gathering, radar-flag setting upon object detection, calculation of camera detection scores with specific weights, and subsequent evaluation. If the combined weights exceeded 50%, a signal would activate the vibrating motor to alert the driver of a potential pedestrian-related accident.

In Figure 18, the pedestrian signature is detected within the range identified for humans. Range measurements served as supplementary data input for the algorithm, offering precise distance information between the pedestrian and the vehicle. During the

actual experimental deployment, these distance plot readings replaced spectrograms due to their real-time capability to provide immediate measurements.

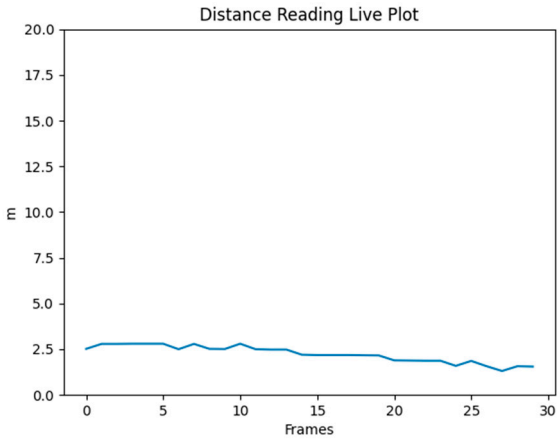


Figure 18. Pedestrian range detection at different frame instances using the micro-Doppler sensor.

In Figure 19, the spectrogram on the left illustrates a vehicle approaching a human at low speeds (1–5 mph), evident from the concavity pointing towards the left. This pattern was noted during vehicle approach at lower speeds. Similarly, the spectrogram on the right depicts a vehicle approaching a human at medium speeds (5–10 mph), displaying a concavity pointing to the left, akin to the phenomenon observed at lower speeds. However, the spectrogram for medium speeds captures less information due to the limited duration it covers within the short distance span (0.5 m to ~7.5 m).

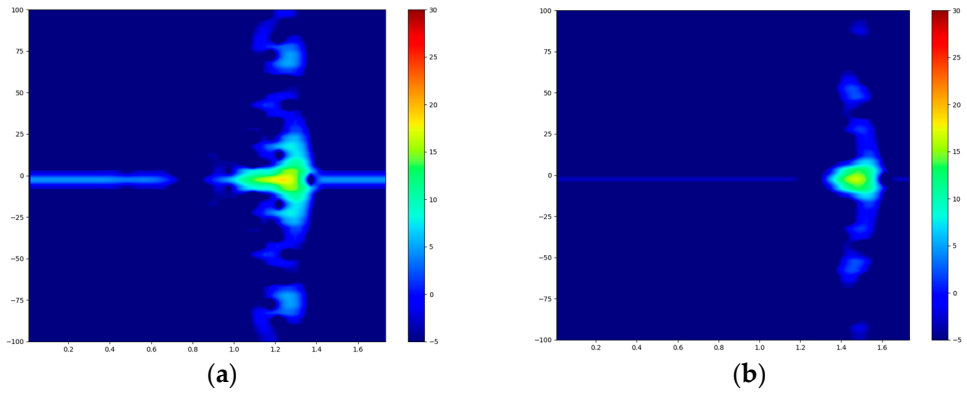
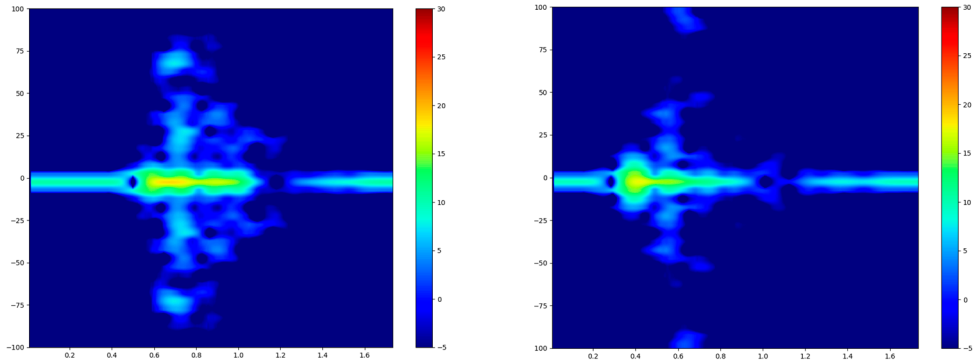


Figure 19. Spectrograms represent the following: (**a**) a vehicle approaching a pedestrian at low speed (1–5 mph), (**b**) a vehicle approaching another vehicle at medium speed (5–10 mph).

In Figure 20, the left spectrogram illustrates a vehicle approaching another vehicle (10–15 mph), displaying a concavity pointing left—similar to the pattern observed when vehicles approached humans at low speeds. This spectrogram captures more information than the human-focused ones because the micro-Doppler effectively records vehicle movement within ~20 m. On the right, the spectrogram depicts a vehicle moving away from the target vehicle starting from rest (0–15 mph), evidenced by the concavity pointing right. This concavity pattern emerges when objects move away from the radar sensor. This spectrogram holds more data compared to the human-focused ones, as the radar sensor effectively captures vehicle movements within ~20 m.



(**a**) (**b**)

Figure 20. Spectrograms represent the following: (**a**) a vehicle approaching another vehicle, (**b**) a vehicle traveling away from the target vehicle at rest.

6. Prototype Experimentation

In this section, we assess the effectiveness of the pedestrian detection scheme specifically designed for vehicle testing.

6.1. System Setup in a Vehicle

The experimental setup for the in-vehicle test includes the utilization of the Google Coral USB Accelerator, integrating an Edge TPU coprocessor into the system. This addition allows high-speed machine learning inferencing on various systems by simply connecting it to a USB port. This on-device ML processing minimizes latency, enhances data privacy, and eliminates the need for a continuous internet connection.

Furthermore, the setup involves the Raspberry Pi 4 Model B, employed for processing real-time data while incorporating the machine learning model. Key features encompass a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K through micro-HDMI ports, hardware video decoding at up to 4Kp60, up to 4 GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, and USB 3.0.

Figure 21 displays the touchscreen display affixed to the car's dashboard, offering real-time information to the driver regarding road conditions and pedestrian detection through three sensors. It indicates the presence or absence of pedestrians and provides distance measurements obtained from the radar sensor. Additionally, the system includes a warning mechanism; upon detecting a pedestrian, it flashes red and activates the steering wheel motor to alert the driver.

Figure 21. Touchscreen display for user interface.

6.2. Testbed Experimentation in a Vehicle

The pedestrian detection system was implemented in a vehicle, as demonstrated in Figures 22 and 23. A motor was affixed to the steering wheel to promptly alert the driver upon detecting a pedestrian on the road who might be at risk.

Figure 22. System Integration in a Vehicle.

Figure 23. System setup on a vehicle.

A live demonstration conducted in a vehicle on a street showcased the system's ability to detect pedestrians, as depicted in Figures 24 and 25. The developed system exhibited robust performance, achieving over 97% accuracy in pedestrian detection during both daytime and nighttime conditions.

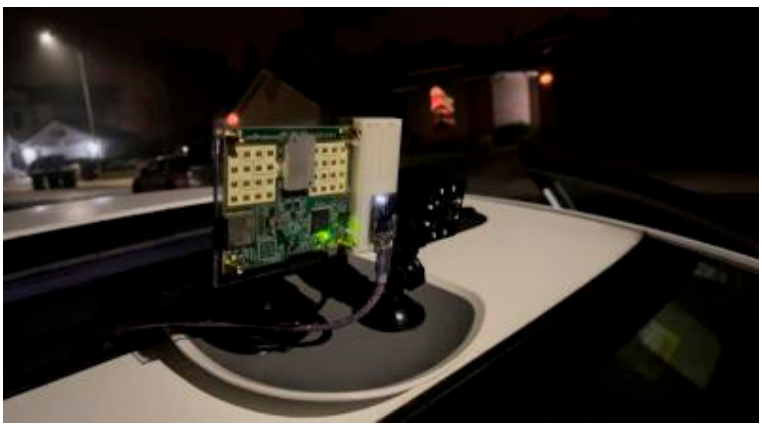


Figure 24. Live demonstration of pedestrian detection at night: system setup on a vehicle.

Figure 25. Live demonstration of pedestrian detection at night: detection scheme in action inside the vehicle.

7. Conclusions

This research project implements a pedestrian detection and avoidance scheme utilizing multi-sensor data collection and machine learning for intelligent transportation systems (ITSs). The system incorporates a video camera, an infrared camera, and a micro-Doppler radar for data gathering and model training. A deep convolutional neural network (DCNN) trains on both RGB and IR camera images, totaling 1200 RGB images (600 with pedestrians) and 1000 IR images (500 with pedestrians, 500 without), with 85% taken during nighttime.

Following dataset curation, two distinct DCNNs were trained. The RGB camera achieved a 99.6% validation accuracy in detecting pedestrians, while the IR camera, predominantly trained on nighttime images, achieved 97.3% accuracy. The radar sensor determined the pedestrian range and travel direction. Vehicle-based experiments confirmed that upon detecting a potentially endangered pedestrian, the multi-sensor scheme activated a signal to the wheel's vibrating motor and displayed a warning message on the passenger's touchscreen. This system functions effectively in both day and night conditions.

Author Contributions: Writing—Original Draft Preparation, H.K.; Conceptualization, H.K., J.B., M.T. (Megan Tamiyasu) and M.T. (Mateo Thompson); Methodology, H.K.; Data Curation, J.B., M.T. (Megan Tamiyasu) and M.T. (Mateo Thompson); Data Fusion Algorithm, M.K.; Formal Simulation and Analysis, H.K., J.B., M.T. (Megan Tamiyasu), M.T. (Mateo Thompson) and M.K.; Experimentation and Implementation, J.B., M.T. (Megan Tamiyasu) and M.T. (Mateo Thompson); Supervision, H.K.; Funding, H.K., Writing—Review and Editing, H.K. and M.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Fresno State Transportation Institute and California State University Transportation Consortium through the State of California's Road Repair and Rehabilitation Act of 2017.

Data Availability Statement: We may provide our collected data on IR camera upon request.

Acknowledgments: Hovannes Kulhandjian would like to acknowledge partial support from Fresno State Transportation Institute (FSTI) and the California State University Transportation Consortium through the State of California's Road Repair and Rehabilitation Act of 2017.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. National Highway Traffic Safety Administration (NHTSA) United States. Early Estimate of Motor Vehicle Traffic Fatalities for the First 9 Months (Jan–Sep) of 2020. Available online: https://www.nhtsa.gov/risky-driving/drowsy-driving (accessed on 5 June 2024).
2. Feese, J. Highway Safety Association, New Projection: 2019 Pedestrian Fatalities Highest Since 1988. February 2020. Available online: https://www.ghsa.org/resources/news-releases/pedestrians20 (accessed on 5 June 2024).
3. Galvin, G. The 10 Worst States for Pedestrian Traffic Deaths. February 2022. Available online: https://www.usnews.com/news/healthiest-communities/slideshows/10-worst-states-for-pedestrian-traffic-deaths?slide=3 (accessed on 5 June 2024).
4. Romero, S. Pedestrian Deaths Spike in U.S. as Reckless Driving Surges. 14 February 2022. Available online: https://www.nytimes.com/2022/02/14/us/pedestrian-deaths-pandemic.html (accessed on 5 June 2024).
5. Valera, D. Your Central Valley News, 64 Percent of Fresno's Deadly Collisions are Vehicle vs. Pedestrian Ones, Police Say. 21 October 2018. Available online: https://www.yourcentralvalley.com/news/64-percent-of-fresnos-deadly-collisions-are-vehicle-vs-pedestrian-ones-police-say/ (accessed on 5 June 2024).
6. Yanagisawa, M.; Swanson, E.D.; Azeredo, P.; Najm, W. *Estimation of Potential Safety Benefits for Pedestrian Crash Avoidance Systems*; National Highway Traffic Safety Administration: Washington, DC, USA, 2017.
7. Tubaishat, M.; Zhuang, P.; Qi, Q.; Shang, Y. Wireless sensor networks in intelligent transportation systems. *Wirel. Commun. Mob. Comput.* **2009**, *9*, 287–302. [CrossRef]
8. El-Faouzi, N.E.; Leung, H.; Kurian, A. Data fusion in intelligent transportation systems: Progress and challenges—A survey. *Inf. Fusion* **2011**, *12*, 4–10. [CrossRef]
9. Lim, H.S.; Lee, J.E.; Park, H.M.; Lee, S. Stationary Target Identification in a Traffic Monitoring Radar System. *Appl. Sci.* **2020**, *10*, 5838. [CrossRef]
10. Cao, Z.; Bıyık, E.; Wang, W.Z.; Raventos, A.; Gaidon, A.; Rosman, G.; Sadigh, D. Reinforcement learning based control of imitative policies for near-accident driving. *arXiv* **2020**, arXiv:2007.00178.
11. Datondji, S.R.; Dupuis, Y.; Subirats, P.; Vasseur, P. A survey of vision-based traffic monitoring of road intersections. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2681–2698. [CrossRef]
12. Al Sobbahi, R.; Tekli, J. Comparing deep learning models for low-light natural scene image enhancement and their impact on object detection and classification: Overview, empirical evaluation, and challenges. *Signal Process. Image Commun.* **2022**, *109*, 116848. [CrossRef]
13. Xiao, Y.; Zhou, K.; Cui, G.; Jia, L.; Fang, Z.; Yang, X.; Xia, Q. Deep learning for occluded and multiscale pedestrian detection: A review. *IET Image Process.* **2021**, *15*, 286–301. [CrossRef]
14. González, A.; Fang, Z.; Socarras, Y.; Serrat, J.; Vázquez, D.; Xu, J.; López, A.M. Pedestrian Detection at Day/Night Time with Visible and FIR Cameras: A Comparison. *Sensors* **2016**, *16*, 820. [CrossRef] [PubMed]
15. Jain, D.K.; Zhao, X.; González-Almagro, G.; Gan, C.; Kotecha, K. Multimodal pedestrian detection using metaheuristics with deep convolutional neural network in crowded scenes. *Inf. Fusion* **2023**, *95*, 401–414. [CrossRef]
16. Fukui, H.; Yamashita, T.; Yamauchi, Y.; Fujiyoshi, H.; Murase, H. Pedestrian detection based on deep convolutional neural network with ensemble inference network. In Proceedings of the 2015 IEEE Intelligent Vehicles Symposium (IV), Seoul, Republic of Korea, 21 June–1 July 2015; pp. 223–228.
17. Luo, Y.; Remillard, J.; Hoetzer, D. Pedestrian detection in near-infrared night vision system. In Proceedings of the 2010 IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 21–24 June 2010; pp. 51–58. [CrossRef]
18. Han, T.Y.; Song, B.C. Night vision pedestrian detection based on adaptive preprocessing using near-infrared camera. In Proceedings of the 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia), Seoul, Republic of Korea, 26–28 October 2016; pp. 1–3.
19. Fu, X. Pedestrian detection based on improved interface difference and double threshold with parallel structure. In Proceedings of the 2019 International Conference on Intelligent Informatics and Biomedical Sciences (ICIIBMS), Shanghai, China, 21–24 November 2019; pp. 133–138.
20. Kulhandjian, H.; Ramachandran, N.; Kulhandjian, M.; D'Amours, C. Human Activity Classification in Underwater using sonar and Deep Learning. In Proceedings of the ACM Conference on Underwater Networks and Systems (WUWNet), Atlanta, GA, USA, 23–25 October 2019.
21. Kulhandjian, H.; Sharma, P.; Kulhandjian, M.; D'Amours, C. Sign Language Gesture Recognition using Doppler Radar and Deep Learning. In Proceedings of the IEEE GLOBECOM Workshop on Machine Learning for Wireless Communications, Waikoloa, HI, USA, 9–13 December 2019.
22. Kulhandjian, H.; Davis, A.; Leong, L.; Bendot, M.; Kulhandjian, M. AI-based Human Detection and Localization in Heavy Smoke using Radar and IR Camera. In Proceedings of the IEEE Radar Conference (RadarConf23), San Antonio, TX, USA, 1–5 May 2023.
23. Kulhandjian, H.; Barron, J.; Tamiyasu, M.; Thompson, M.; Kulhandjian, M. Pedestrian Detection and Avoidance at Night Using Multiple Sensors and Machine Learning. In Proceedings of the IEEE International Conference on Computing, Networking and Communications (ICNC), Honolulu, Hawaii, 20–23 February 2023.
24. Kulhandjian, H.; Martinez, N.; Kulhandjian, M. Drowsy Driver Detection using Deep Learning and Multi-Sensor Data Fusion. In Proceedings of the IEEE Vehicle Power and Propulsion Conference (IEEE VPPC 2022), Merced, CA, USA, 1–4 November 2022.

25. FLIR. Flir Lepton Engineering Datasheet. 2018. Available online: https://cdn.sparkfun.com/assets/e/9/6/0/f/EngineeringDatasheet-16465-FLiR_Lepton_8760_-_Thermal_Imaging_Module.pdf (accessed on 5 June 2024).
26. Teledyne FLIR Thermal Dataset for Algorithm Training. Available online: https://www.flir.com/oem/adas/adas-dataset-form/ (accessed on 5 June 2024).
27. Tzutalin. LabelImg. GitHub Code. 2015. Available online: https://github.com/tzutalin/labelImg (accessed on 5 June 2024).