milliEye: A Lightweight mmWave Radar and Camera Fusion System for Robust Object Detection

Xian Shuai
The Chinese University of Hong Kong
Hong Kong, China
sx018@ie.cuhk.edu.hk

Yulin Shen
University of Electronic Science and Technology of China
Chengdu, China
yulinshen@std.uestc.edu.cn

Yi Tang
The Chinese University of Hong Kong
Hong Kong, China
ytang@ie.cuhk.edu.hk

Shuyao Shi
The Chinese University of Hong Kong
Hong Kong, China
ss119@ie.cuhk.edu.hk

Luping Ji
University of Electronic Science and Technology of China
Chengdu, China
jiluping@uestc.edu.cn

Guoliang Xing*
The Chinese University of Hong Kong
Hong Kong, China
glxing@ie.cuhk.edu.hk

ABSTRACT

Recent years have witnessed the emergence of a wide range of advanced deep learning algorithms for image classification and object detection. However, the effectiveness of these methods can be significantly restricted in many real-world scenarios where the visibility or illumination is poor. Compared to RGB cameras, millimeter-wave (mmWave) radars are immune to the above environmental variability and can assist cameras under adverse conditions. To this end, we propose *milliEye*, a lightweight mmWave radar and camera fusion system for robust object detection on the edge platforms. *milliEye* has several key advantages over existing sensor fusion approaches. First, while *milliEye* fuses two sensing modalities in a learning based fashion, it requires only a small amount of labeled image/radar data of a new scene because it can fully utilize large image datasets for extensive training. This salient feature enables *milliEye* to adapt to highly complex real-world environments. Second, based on a novel architecture that decouples the image-based object detector from other modules, *milliEye* is compatible with different off-the-shelf image-based object detectors. As a result, it can take advantage of the rapid progress of object detection algorithms. Moreover, thanks to the highly compute-efficient fusion approach, *milliEye* is lightweight and thus suitable for edge-based real-time applications. To evaluate the performance of *milliEye*, we collect a new radar and camera fusion dataset for object detection, which contains both ordinary-light and low-light illumination conditions. The results show that *milliEye* can provide substantial performance boosts over state-of-the-art image-based object detectors, including Tiny YOLOv3 and SSD, especially in low-light scenes, while incurring low compute overhead on edge platforms.

*Corresponding author

IoTDI'21, May 18-21 2021, Nashville, Tennessee, USA

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

CCS CONCEPTS

• **Computer systems organization → Neural networks**; • **Computing methodologies → Object detection**.

KEYWORDS

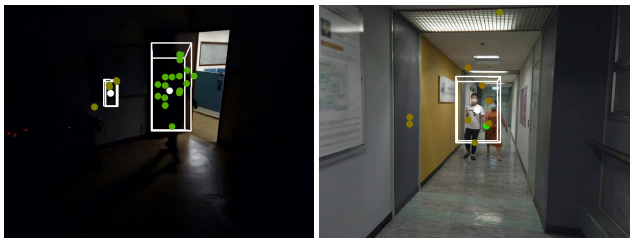
mmWave Radar, Sensor Fusion, Object Detection

ACM Reference Format:
Xian Shuai, Yulin Shen, Yi Tang, Shuyao Shi, Luping Ji, and Guoliang Xing. 2021. milliEye: A Lightweight mmWave Radar and Camera Fusion System for Robust Object Detection. In *Proceedings of Internet of Things Design and Implementation (IoTDI'21).* ACM, New York, NY, USA, 13 pages. https://doi.org/10.475/123_4

1 INTRODUCTION

Accurate machine perception plays a critical role in intelligent systems, such as autonomous vehicles. At the heart of perception is to localize and identify objects of interest, also known as object detection. With the advancement of deep learning, many convolutional neural networks (CNNs) based object detectors have been proposed [23, 30, 31]. While these image-based detectors have achieved promising performance on public benchmarks like PASCAL VOC and COCO [11, 22], they are vulnerable to adverse environmental conditions [6, 32], such as foggy weather, smog or poor illumination. On the other hand, mmWave radars [1, 13, 25] have emerged as a low-cost sensor modality for all-weather conditions and have been widely used in many embedded and edge applications. Owning to the ability to work under darkness and the penetrability to airborne obstacles, mmWave radars are widely adopted in combination with cameras in non-ideal environments where the functionality of RGB cameras is hindered. Despite the above salient characteristics, the point cloud from mmWave radars is usually sparse and noisy due to the specular reflections, signal leakage, multi-path effects and low angular resolution, making the accurate radar-based object detection challenging.

Figure 1 illustrates the complementary characteristics of mmWave radar and camera. Figure 1(a) shows the results of radar-based tracking using DBSCAN clustering and Kalman filter, where the 3D point cloud and 3D boxes are mapped onto 2D image, and colors on points represent different depths. Figure 1(b) shows the results of Tiny



(a) Object tracking results on mmWave radar point cloud.

(b) Object detection results of Tiny YOLOv3 on RGB images.

Figure 1: An example of complementary detection performance of mmWave radar and camera. Radar data are noisy, sparse, and in low-resolution, while effective under poor illumination. The image-based object detector fails to detect the person in darkness, while it can accurately predict bounding boxes under ordinary light conditions.

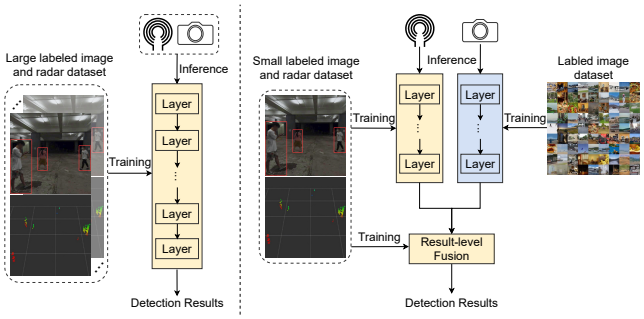


Figure 2: Left: conventional camera and radar fusion architecture that requires a large labeled image and radar dataset (> 10,000 frames) for training. Right: our proposed fusion architecture which enables separated weight training. Image-relevant layers are trained on the large image dataset. Radar-relevant layers and scenario-agnostic result-level fusion layers are trained on the small customized multi-modality dataset (< 1,000 frames).

YOLOv3, a compact image-based object detector. In the first scene, the radar is able to localize the person, while the image-based object detector fails due to insufficient illumination. In the second scene, the radar fails to separate two people because they are close to each other, while the camera separates them well. Neither the radar-based method nor the image-based method alone performs satisfactorily in two scenes, which motivates the necessity of the fusion of two sensing modalities.

Traditional camera and radar fusion methods generally feed observations from individual sensors into a Kalman Filter (or its variants) sequentially [7, 34]. Such an approach either over-simplifies the radar-based detection to a point target detection problem or requires hand-crafted design of the fusion strategy. Only until recently has the deep learning based fusion been investigated [1, 3, 12, 26]. As shown in Figure 2, they resort to an intact end-to-end CNN, which takes raw image and radar data as inputs, and achieves fusion usually by concatenating the internal image and radar feature maps. However, those naive fusion methods are largely impractical for real-world applications. This is because the fusion model needs to be trained from scratch using the multi-modality dataset, and the acquisition of image feature extraction capability requires a mass of labeled data. Unlike the comprehensive image datasets like ImageNet and COCO [9, 22], which cover dozens of object categories and scenarios, the publicly available camera and radar datasets [1, 2] exclusively focus on highly specifical applications and scenarios. Therefore, users usually need to collect and annotate their own datasets according to the object categories and deployment environments, which is typically labor-intensive and prohibitively expensive. To tackle this challenge, we propose a loosely coupled architecture that can adapt to a new scenario using a small amount of labeled multi-modality data. As shown in the right-hand side of Figure 2, the training of layers that are directly exposed to image features is separated from the training of other modules.

Our design is motivated by two considerations. First, there already exist abundant large-scale image datasets [11, 22, 24] for object detection that can be taken advantage of to train the image-relevant layers. Second, compared to the image feature extraction, radar data extraction and result-level fusion are more domain-invariant and less sensitive to the appearance change of objects, thus are trainable by less labeled data. Based on these observations, we propose *milliEye*, a lightweight and practical mmWave radar and camera fusion system for robust object detection. *milliEye* first integrates bounding box proposals from an image-based object detector and a radar-based tracker to handle adverse environmental conditions where the image-based detector alone may fail. *milliEye* then employs a novel fusion-enabled refinement module to refine those box proposals for more accurate detection. The design of the refinement module follows the philosophy that the weight training of appearance-sensitive image-relevant layers is separated from the training of other modules. Compared to current fusion techniques [3, 7, 26, 27], *milliEye* has three system-level advantages as follows:

Adaptability. First, *milliEye* can be trained to adapt to a new environment using a small multi-modality dataset. This is attributed to the separated weight training approach which retains the generalizable and robust image feature extraction capability. Second, by learning-based fusion, it circumvents the rigid hand-crafted fusion strategies and enables the relative importance of data from two sensing modalities to automatically shift in response to environmental dynamics. For example, *milliEye* will automatically rely more on mmWave radar data at night, while mainly rely on camera when the illumination condition is ideal.

Lightweight. The compute overhead of *milliEye* is light even compared to compact image-based object detectors, such as Tiny YOLOv3

[30], which allows *milliEye* to run efficiently on resource-limited mobile and edge devices. This is because *milliEye* reuses the internal feature maps of image-based detector and the proposed fusion method is highly compute-efficient. Therefore, *milliEye* does not impair the real-time performance of the original detector while yielding significant performance improvement, especially in challenging low-light scenarios.

Compatibility. Thanks to the novel fusion architecture, *milliEye* can directly acquire the performance of integrated image-based detectors trained on publicly available large-scale image datasets. Meanwhile, users can easily choose different object detectors without an overhaul re-design of the fusion *milliEye*. These features allow our system to benefit from the rapid progress of object detection in the filed of computer vision.

In brief, we conclude the contributions of this work as follows:

- We collect a practical multi-modality dataset for radar and camera fusion, based on which we investigate the performance of existing image-based object detectors and fusion-based detectors under different illumination conditions. Our dataset will be made available to the community.
- We propose *milliEye*, an mmWave and camera fusion system for real-time robust object detection. Our novel fusion-enabled refinement head can enhance the off-the-shelf image-based object detectors.
- We conduct extensive experiments to demonstrate the superiority of *milliEye* in three aspects: adaptability, compute overhead and compatibility. In challenging scenarios, *milliEye* achieves an mAP of 74.1% (compared to 63.0% of Tiny YOLOv3), while only incurring an additional average delay of 16.8 ms per-frame on Jetson TX2. To the best of our knowledge, this is the first-of-its-kind object detection framework that accounts for the above multiple system-level factors.

The rest of this paper is organized as follows. Section 2 introduces relevant background including the regular mmWave radar processing pipeline and a recap for object detection. In Section 3, we present the design of *milliEye*. In Section 4, we detail the implementation and conduct extensive experiments on *milliEye*. Section 5 presents related work. Finally, Section 6 concludes the paper and discusses several limitations.

2 BACKGROUND

2.1 mmWave Radar Processing Pipeline

A mmWave radar is an active detection and ranging sensor operating within the band frequency from 30GHZ to 300GHz. In this paper, we use a commercial Frequency Modulated Continuous Waveform (FMCW) mmWave radar [16]. Range, doppler and AOA estimation are common post-processing steps to extract the range, radial velocity and angle of objects from the reflected radar signals.

Specifically, an FMCW radar transmits a sequence of linear "chirps" during a frame. The received signals are mixed with the transmitted ones to obtain a series of intermediate-Frequency (IF) signals, whose frequencies are proportional to the distance of the object. By applying the Fast Fourier Transform (FFT) to each IF signal, a frequency spectrum is obtained, in which each peak indicates the range of an obstacle. In addition, the motion of the object will cause the phase of IF signals to change within a frame. The frequency of phase change is proportional to the radial velocity which can be estimated by performing FFT on a group of IF signals. When multiple TX and RX pairs are available, radars can further estimate the AOA using the phase differences of received signals from different virtual antennas. While increasing the number of TX or RX can improve the angular resolution, large MIMO antennas are impractical for commercial single-chip radars. A typical commercial mmWave radar with a 3×4 MIMO array achieves poor angular resolution about 14° in azimuth and 57° in elevation. Subsequent to the above range, doppler and AOA estimation, mmWave radar generates a set of points, where each point includes the 3D spatial position and the radial velocity towards the radar (See Equation 1). Although inaccurate in position, these points can give strong indication of the occupancy and movement information of objects.

2.2 Recap for Object Detection

Current object detection methods can be mainly classified into two categories: single-stage detectors like YOLO [30], Single Shot Detector (SSD) [23] and two-stage ones like Faster-RCNN [31], R-FCN [8] and Light-Head RCNN [20]. Single-stage object detectors perform regression and classification directly on predefined anchored boxes. As a result, they are more amenable for edge and embedded devices due to low computational overhead. Two-stage detectors disassemble the detection task into two stages. Take the representative Faster-RCNN as an example. In the first stage, the region proposal network (RPN) generates about 300 candidate object bounding boxes. Then an RoI (region of Interest) pooling layer crops these candidate boxes one-by-one from feature maps. Lastly, an R-CNN subnet consisting of several fully-connected layers performs classification and box regression for every RoI. In general, two-stage detectors can achieve higher accuracy than one-stage ones for two reasons. First, the RPN narrows down the number of candidate boxes by eliminating the majority of background instances, providing a better foreground and background balance than one-stage detectors during the training. Second, two-stage detectors conduct the box refinement twice, once in RPN and once in the detection layer, while the one-stage detector only performs once. However, two-stage detectors like Faster-RCNN suffer long inference time because they need to run the costly per-RoI detection head hundreds of times for an image. To alleviate this issue, R-FCN proposes position-sensitive score maps and position-sensitive pooling, which enables the replacement of the heavy per-RoI detection head with a simple average pooling operation, hence largely reduces the computation. Light-head RCNN further decreases the computation by using a thinner position-sensitive score map compared to what is taken in R-FCN. Both single-stage and two-stage object detectors output a great number of bounding boxes for one image. For instance, YOLOv3 outputs 10646 boxes, and Faster-RCNN outputs 300 boxes. In order to eliminate redundant boxes, two post-processing steps usually follow the neural network of object detectors. The first step is to filter out the boxes with a low confidence score, i.e, those that have low possibility to contain any instance. The second step is to perform Non-Maximum Suppression (NMS), which further eliminates overlapping bounding boxes.

3 DESIGN OF MILLIEYE

3.1 Overview

As introduced in Section 2.2, compared to one-stage detectors, two-stage detectors contain one additional refinement step (e.g., the R-CNN), which increases the accuracy. Another insight is that the radar data are indicative of the occupancy of objects, as introduced in 2.1, which can help the refinement module better distinguish objects from the background. Motivated by these insights, we design a novel fusion-enabled refinement head to improve the performance of existing image-based object detectors. In addition, to handle challenging scenarios where the image-based object detector fails to propose enough desirable candidate boxes for further refinement, we exploit the radar point cloud and design a radar-based tracker that works as an alternative box proposal module.

An overview of *milliEye* is shown in Figure 3. From a system-level perspective, *milliEye* follows a two-stage fusion paradigm. The first stage aims at aggregating box proposals from the image-based object detector ($\mathcal{I}$) and the radar-based tracker ($\mathcal{R}$): $\mathcal{B} = \{\mathcal{B}^{\mathcal{I}}, \mathcal{B}^{\mathcal{R}}\} = \{b_k\}_{k=1}^{K}$, where $K = |\mathcal{B}|$ is the total number of RoIs. Meanwhile, the global feature extraction is performed on the entire frame for both the image and radar branch to obtain the global multi-modality feature maps $\mathcal{G}^{\mathcal{I}}$ and $\mathcal{G}^{\mathcal{R}}$. Then two RoI-wise cropping operations (i.e., the PS-RoI Align [20] and RoI Align [14]) crop the global feature maps according to the location of box proposals to obtain per-RoI local features: $\mathcal{L}^{\mathcal{I}}, \mathcal{L}^{\mathcal{R}} = Cropping(\mathcal{G}^{\mathcal{I}}, \mathcal{G}^{\mathcal{R}}; \mathcal{B})$, where $\mathcal{L}^{\mathcal{I}} = \{l_k^I\}_{k=1}^{K}$, $\mathcal{L}^{\mathcal{R}} = \{l_k^R\}_{k=1}^{K}$ are the sets of local per-RoI feature maps from two modalities. In the second stage, a fusion-enabled refinement head predicts a new location and a confidence score for each box: $b'_k = Refinement(l_k^I, l_k^R, b_k)$. In Section 4.3, we validate that this new bounding box b'_k is more reliable than the original one b_k, since it further incorporates the information from multi-modality feature maps l_k^I and l_k^R. Based on the new confidence scores, we can determine whether each box proposal should be retained, simply using a threshold. Note that the first stage performs only once for each frame, while the second stage repeats for every RoI.

To sum up, *milliEye* mainly consists of three modules, an image-based object detector, a radar-based object tracker and an RoI-wise refinement head. These three modules are assembled in a loosely coupled manner, which endows *milliEye* the ability to support different image-based object detectors. Furthermore, it enables the separated weight training of image-relevant modules, alleviating the reliance on a large amount of labeled multi-modality data.

3.2 Image-Based Object Detector

In this module, we employ a CNN-based object detector introduced in Section 2.2 to obtain bounding boxes together with category scores and confidence scores. Specifically, given an image I, a feature extractor (*body*) F_{body} first extracts the internal feature maps $f = F_{body}(I)$. A typical feature extractor consists of several convolutional layers, activation layers and pooling layers. Then another network (*head*) F_{head} processes the feature maps and generates a set of output boxes B. Thus, the entire process of an object detector can be summarized as $B = F_{head}(F_{body}(I))$. Note that both one-stage and two-stage detectors follow this paradigm, where the only difference is that the two-stage object detectors include an RPN structure in F_{head}, while the one-stage object detectors not.

In *milliEye*, both the box detection results B and the internal feature maps f are leveraged by the refinement head. Specifically, B is first filtered by a confidence threshold, and is combined with radar boxes as the total candidate box proposals. f is used to construct the position-sensitive score maps [8] through a 1×1 convolutional layer. The reuse of the feature maps f from the image-based detector can help save considerable computation, which is highly desirable for embedded platforms. In additional, *milliEye* is compatible with different object detectors, including YOLO and SSD. Such compatibility helps take advantage of advancement of detection algorithms and allows users to change the object detector easily without re-designing the whole fusion model.

3.3 Radar-Based Object Tracker

The image-based object detector may fail to confidently generate any box under harsh environments when the camera suffers significant performance degradation due to darkness, poor visibility or bad weather conditions. In such a case, to ensure that our system can still generate desirable detection results using the radar point cloud data, we propose a radar-based object tracker. We note that a similar approach is adopted in [38] for radar point cloud tracking. However, we extend the tracking object from points to 3D boxes and enable the automatic 3D box size estimation, instead of generating predefined fixed-size bounding boxes.

As showed in Figure 4, in a frame, the mmWave radar data is a set of points, where each point is represented by a 4-d vector composed of coordinates on x (left to right), y (up to down), z (back to forth) axis and the radial velocity (the velocity along z-axis), respectively. For clarity, we denote the i^{th} point by:

$$p_i := (x_i, y_i, z_i, v_i) \in \mathbb{R}^4 \tag{1}$$

Then the workflow to generate box proposals from the radar point cloud is as follows.

3.3.1 Point Cloud Clustering. Signals from clutter and noise can hugely contaminate radar point cloud and trigger undesirable false positive points. Therefore, we use DBSCAN [10], a density-based clustering method, to identify foreground objects from the clutter. Our intuition is that points from foreground objects can group as clusters, while points from the clutter are usually scattered in low-density. In addition, unlike K-means, DBSCAN requires no prior information of the number of clusters, and hence is well-suited for object detection tasks where the number of objects is arbitrary. We define the distance between two points as follows, which is used as the distance metric in DBSCAN for density-connection check:

$$\begin{aligned} d(i,j) :=& \alpha_x(x_i - x_j)^2 + \alpha_y(y_i - y_j)^2 + \\ & \alpha_z(z_i - z_j)^2 + \alpha_v(v_i - v_j)^2 \end{aligned} \tag{2}$$

where $\alpha = [\alpha_x, \alpha_y, \alpha_z, \alpha_v]$ is the weight vector to balance the contribution of each element. Here we incorporate velocity information during the clustering process because it can help separate two nearby objects with different speeds, such as when two people pass by face-to-face.

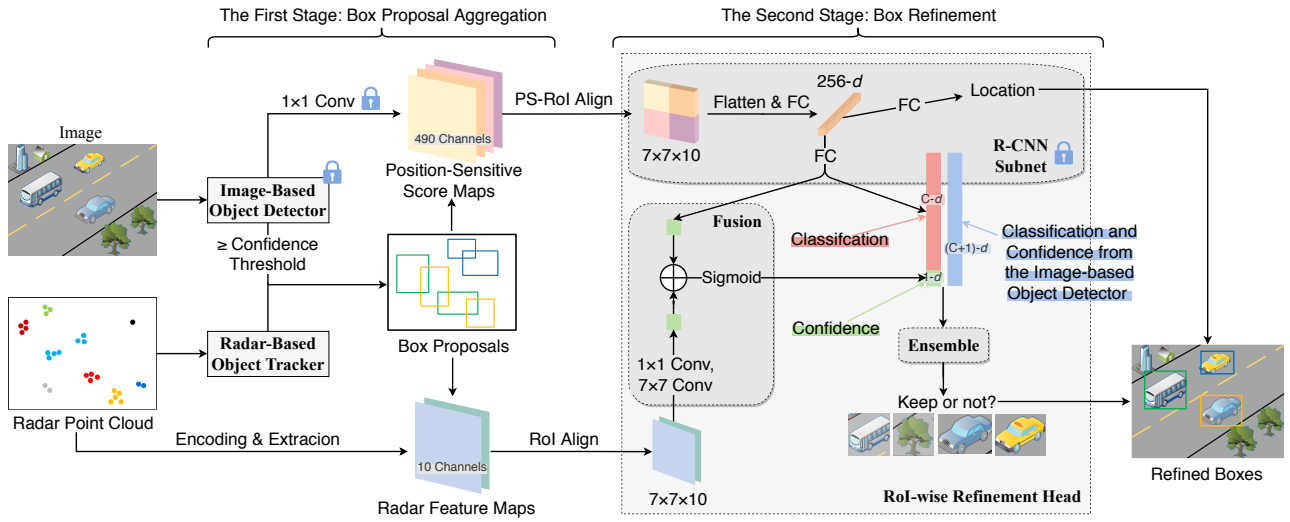


Figure 3: The system architecture of *milliEye*. It includes two stages: box proposal aggregation and box refinement. The blue locks in the diagram mean the weights of these components are frozen during the training on multi-modality dataset, which will be detailed in Section 3.5.

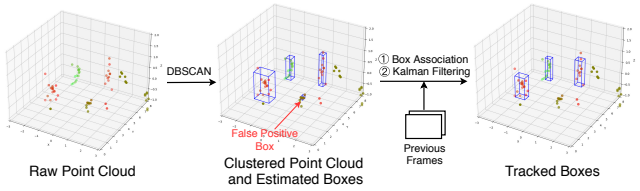


Figure 4: Pipelines of radar-based tracking. Each point is associated with a velocity, represented by the color on it. The velocity information helps separate nearby objects in DBSCAN. Information from previous frames is leveraged to adjust the box size and filter out flickering false positive boxes.

3.3.2 Box Estimation for Clusters. After DBSCAN, each point is labeled by either the index of a cluster or a flag of outliers. After filtering out outliers, we estimate the center position and the velocity along z-axis of each cluster by averaging the corresponding value of all points in this cluster. In addition, for each cluster, we search for the outermost points belonging to it and use these points to approximate the size of 3D bounding box. The estimated 3D bounding box of each cluster can be defined by:

$$z := (x, y, z, v_z, w, h, t) \in \mathbb{R}^7 \tag{3}$$

where w, h, t are the width, height and thickness of the 3D bounding box, respectively.

3.3.3 Box Tracking. It is of great significance to exploit the temporal continuity embedded in adjacent frames to further eliminate the false positive boxes that flicker in frames. Particularly, multiple boxes can be generated from each frame. To associate multiple boxes across frames into temporally consistent tracklets, we use the Hungarian algorithm [19] and take the Euclidean distance between centers of any two boxes as the matching metric. However, the associated boxes from adjacent frames can still jitter severely. We hence use a Kalman Filter to smooth the locations and sizes of boxes. Specifically, in frame $N-1$, assume the Kalman Filter keeps a state vector $s_{i,N-1}$ for box i:

$$s_{i,N-1} := [x, y, z, v_x, v_y, v_z, w, h, t] \in \mathbb{R}^9 \tag{4}$$

In frame N, we first predict the new state vector $s'_{i,N}$ for box i using a constant velocity model. Then the Kalman Filter corrects the state vector $s'_{i,N}$ to $s_{i,N}$ according to the observation $z_{i,N}$, whose form has been introduced in Equation 3:

$$s_{i,N} = s'_{i,N} + K(z_{i,N} - Hs'_{i,N}) \tag{5}$$

where $K \in \mathbb{R}^{9\times7}$ is the Kalman gain matrix and $H \in \mathbb{R}^{7\times9}$ is the observation model matrix. Note that the state vector is of length 9, while the observation vector is of length 7, because mmWave radars do not provide the velocity on x and y directions.

If a box fails to associate with any new box in the next frame, we continue predicting the new state vector using the constant velocity model. If the association fails successively for T_{max_age} frames, we assume the object disappears and stop predicting.

3.3.4 Projection and Synchronization. The above clustering and tracking steps generate the 3D bounding boxes under the radar coordinate and timestamps. To achieve fusion between two sensors, a uniform coordinate system and timestamp are required. Spatially, we slice 3D bounding boxes on z-axis and obtain cross-sections. Then these cross-sections are projected into 2D image. The projection of each point follows:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z} KT \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{6}$$

where K is the 3×3 camera's intrinsic matrix, and T is the 3×4 extrinsic matrix. (x, y, z) is the 3D location in the radar coordinate system, and (u, v) is the projected pixel location in the image. As the relative position of two sensors is fixed, both K and T can be calculated offline. For temporal synchronization, we associate every image frame with its nearest radar frame.

3.4 Fusion-Enabled Refinement

In order to exert the occupancy detection capability of radar, while maintaining the robust image feature extraction performance obtained from large image datasets, we propose a fusion-enabled refinement head, which mainly includes three components: a Region-Based CNN (R-CNN) subnet, a fusion module and an ensemble module, as showed in Figure 3. The R-CNN subnet performs the box regression and classification, leveraging the knowledge learned from abundant image data. The fusion module aggregates the confidence score from two sensing modalities, and the ensemble module further seeks the wisdom of image-based detector for a more reliable prediction. Our key insight is that the fusion and ensemble module are not exposed to the image feature, thus being appearance-insensitive and trainable using a small amount of multi-modality labeled data. We will elaborate on these three components along with the steps of per-RoI feature extraction in the following.

3.4.1 Per-RoI Feature Extraction. The refinement head takes the cropped per-RoI feature maps from both image branch and radar branch as input.

Per-RoI Image Features. We detach the internal features maps of the image-based detector, and employ a single 1×1 convolutional layer to construct a Position-Sensitive (PS) score map with 490 channels. Then a Position-Sensitive Region of Interest Align (PS-RoI Align) layer is used to crop the score map according to the location of each box proposal. For each RoI, we obtain a 7×7×10 feature map. The detail of PS-RoI Align can be found in [20], which is beyond the scope of this paper.

Per-RoI Radar Features. Given that radar point cloud is sparsely scattered in the field of view (FOV) and has uncertain length, it is difficult to leverage the point cloud directly for spatial pattern extraction. To utilize the powerful spatial feature extraction capability of convolutional layers, we encode the unstructured radar point cloud into 2D images through three preprocessing steps: (1) Project the point cloud into the 2D image coordinate. (2) Calculate the 2D histogram of the projected point cloud on three channels: the number of points, mean depth, mean velocity on z-axis. (3) Normalize the value on each channel into the range of [0, 1]. Through these steps, we obtain a 3-channel heatmap. This 3-channel heatmap is then fed into a three-layer CNN to extract occupancy feature maps, which embeds the probability that the target exists at each location. An RoI Align layer [14] then crops the occupancy map and generates the per-RoI radar features, whose size is also 7×7×10.

3.4.2 R-CNN Subnet. Based on previous work [20], we propose a lightweight R-CNN subnet, whose outputs on confidence score will be further fused with the information from the radar branch. Intuitively, this module performs an extra box refinement stage for the image-based object detector, thereby improving the box localization and classification accuracy. Specifically, we first flatten the per-RoI image feature map, and then apply a single fully-connected (FC) layer with 256 channels, followed by two sibling FC layers for box regression and classification (the regression of confidence score is also a kind of classification). The outputs are a 4-d vector, and a $(C + 1)$-d vector, respectively.

Table 1: Architecture of the ensemble module.

Layer	Input Size	Output Size
FC	(c+1)×2	(c+1)×32
Flatten	(c+1)×32	32(c+1)
FC	32(c+1)	2
Softmax	2	2

3.4.3 Sensor Fusion. Although the radar point cloud is sparse and in low-resolution, it can give strong indications of the existence of objects, which is an important supplement to the confidence score from the R-CNN subnet. Therefore, we propose a fusion module to jointly consider two sensing modalities to better estimate the occupancy information. Specifically, for the per-RoI radar features, we use two convolutional layers with kernel size 7×7 and 1×1 to abstract the global representation, which is also the confidence score from the radar. Then confidence scores from two modalities are added together and sent into a sigmoid layer for the final score.

3.4.4 Ensemble. As showed in Figure 3, the outputs of the R-CNN and the fusion module construct a $(C+1)$-d vector, whose dimension is the same as the output of the image-based object detector. In order to incorporate two $(C + 1)$-d results for a more reliable refinement, we introduce a learning-based ensemble module, whose detailed structure is shown in Table 1. The first FC layer fuses the per-class information from two inputs, and the second FC layer captures the global correlation among classes. The final softmax layer outputs a 2-d vector $(P_{foreground}, P_{background})$, forcing the network to make a decision between the foreground and the background. User can assign a threshold to $P_{foreground}$ to determine the list of bounding boxes to keep. Intuitively, a larger inter-category variance a more consistent classification results between two input vectors and render a higher output confidence score. Note that in this module we skip boxes from the radar-based tracker, since there are no corresponding detection results from the image-based object detector.

3.5 Training Strategy and Loss Function

milliEye supports separated training of image-relevant and radar-relevant modules. This feature enables our system to learn robust and generalizable image features on diverse large image datasets, while learning how to jointly leverage radar and camera data through the small multi-modality dataset. We conduct the training in a three-step manner. First, we train image-based object detector. Second, we fix the detector and train the R-CNN subnet, using the same loss function with Faster R-CNN [31]. Since the above two steps only involve image data, they can be performed on large image dataset. Lastly, we fine-tune the radar-relevant parts on self-collected multi-modality dataset using the loss function defined in

Table 2: A summary of the used datasets.

Dataset	Image	Radar	Illumination	# Frames	# Classes
COCO [22]	✓	×	Ordinary-Light	90150	12
ExDark [24]	✓	×	Low-Light	7363	12
Ours	✓	✓	Both	1353	1

the following, with the weights of the image-based object detector and the R-CNN frozen, as represented by the blue locks in Figure 3.

The objective function that the neural network is requested to minimize during the fine-tuning (i.e., training stage 3) on the multi-modality dataset includes two terms. First, the ensemble module determines whether an RoI should be kept or not, which is a regression problem. Therefore, we use the focal loss [21], which is defined as:

$$L_{Focal,i} = \begin{cases} -\alpha(1-p_i)^{\gamma}\log p_i, & y_i = 1 \\ -(1-\alpha)p_i{}^{\gamma}\log(1-p_i), & y_i = 0 \end{cases} \tag{7}$$

where $y_i \in [0,1]$ is the label about keeping or discarding the RoI, and p_i is the predicted confidence score from the ensemble module. α is the factor to balance the positive and negative samples, and γ is a modulating factor that emphasizes hard negatives during training. This loss term is only calculated for box proposals from the image-based detector, since proposals from the radar-based tracker do not involved in the ensemble module.

Moreover, to force the fusion module to mimic the behavior of a binary classifier to generate a reliable confidence score about whether the proposed RoI is a positive instance, we use a binary cross-entropy (BCE) loss, which is given by:

$$L_{BCE,i} = -y_i \log q_i \tag{8}$$

where the definition of y_i is the same with that in Equation 7, and q_i is the confidence score predicted by the fusion module.

For the sake of training stability, we calculate loss only for positive and negative samples, whose IoU with ground truths are larger than 0.7 or smaller than 0.3 respectively. To balance the multi-task training, the final loss is a weighted sum of the above two terms:

$$L = \sum_{i \in pos \cup neg} (\mathbb{1}(i \in img) \cdot L_{Focal,i} + \lambda L_{BCE,i}) \tag{9}$$

4 EVALUATION

In this section, we conduct extensive experiments to demonstrate the performance and advantages of *milliEye*.

4.1 Experimental Settings

4.1.1 Datasets. The experiments involve three datasets in total. For the sake of clarity, we summarize them in Table 4.1 and show some example images in Figure 5.

Microsoft COCO (COCO). COCO [22], released in 2014, is a large-scale object detection, segmentation, and captioning dataset. With more than 200K labeled images, 1.5 million object instances and 80 object categories, COCO has become one of the most popular benchmark datasets for the object detection task. COCO mainly includes images under normal illumination. Although there are 565 low-light images in COCO, they accounted for only 0.23% of the whole dataset, thus the influence is ignorable. In this paper, to make the categories consistent with ExDark, we only use a 12-class sub-dataset of COCO, which contains 90150 images in total.

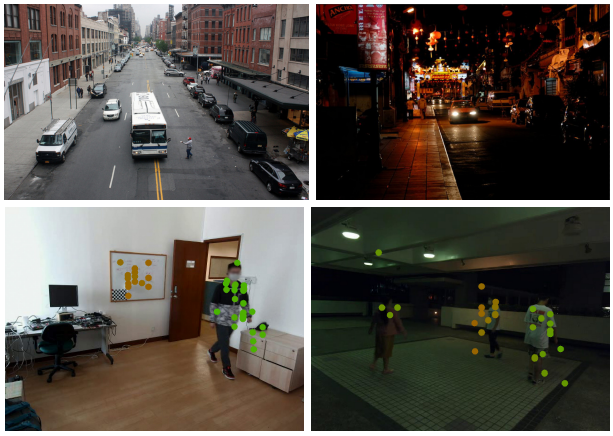


Figure 5: The first row are example images taken from COCO [22] and ExDark [24]. The second row presents example frames from our collected dataset, from ordinary-light scenes and low-light scenes, respectively. From the bottom-left image, we can observe the multi-path effect caused by the radar signal reflected by the wall, which generates many false-positive points (colored in brown).

Exclusively Dark Image Dataset (ExDark). ExDark [24] is a dataset composed exclusively of low-light images. It has total 7363 poor-illuminated images collected in both indoor and outdoor scenes. Labels are annotated in the bounding-box level and image level. The 12 categories contained in this dataset is exactly a subset of the 80 categories of COCO. Images on these 12 categories are distributed relatively evenly, with each category accounting for 7% ∼ 11% of the total number of frames.

Self-Collected Dataset. We collect a single-class human detection dataset. During the experiment, we let volunteers walk randomly in front of the sensors[1]. Figure 6(a) shows the sensor suite used for data collection, where both the radar and camera are fixed on a carrier board and set to the sampling frequency of 20Hz. We sample key frames at 4Hz and finally obtain 1353 frames of images and radar data. We annotate every image key frame with 2D bounding boxes. To encompass diverse configurations, we collect the data in 7 different places including office rooms, corridors, parking lots and semi-open platforms in two buildings. As showed in Figure 6(c), each frame contains 1 to 4 people. Two illumination levels are evenly distributed among the dataset. Since our dataset is small-scale, in the experiment, we follow a 5-fold cross validation paradigm, as demonstrated in Figure 6(b), where we take the average of five trails as the reported results. A key principle we follow in dividing them into 5 folds is to guarantee that data in different folds are collected in different places to demonstrate generalization.

4.1.2 Implementation. The first two training stages introduced in Section 3.5 are conducted on the mixed dataset of COCO [22]

[1]Experiments that involve humans are approved by the IRB of authors' institution.

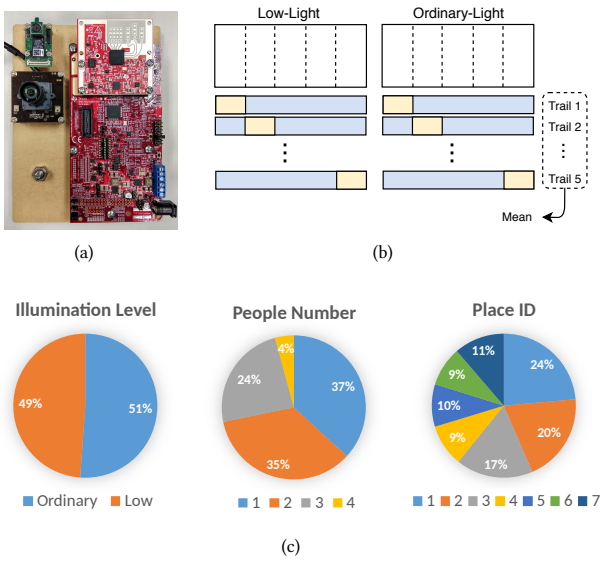


Figure 6: (a) We use a USB 2.0 camera (the black module) and a commodity 60-64GHz mmWave radar TI IWR6843 (the red module) for data collection. (b) The blue parts denote the training data and the yellow parts denote the test data. For each illumination level, the sub-datasets are divided five-fold. During evaluation, we average the results of five trails as the final results. (c) The distribution of our dataset on illumination levels, people number and data collection places.

and ExDark [24]. The third training stage is conducted using our multi-modality dataset. Regarding the choice of hyper-paramters during the training, we use Adam optimizer with a $1e^{-4}$ initial learning rate and batch size of 32. The γ in the focal loss is set to 2 as recommended in [21], and α is set to 0.75 to balance the contribution from positive and negative samples. We manually choose the balancing factor λ of the final loss function to make two terms in the same order of magnitude at the beginning of training. The NMS threshold is chosen to be 0.5 for all experiments. As for the weight vector used in the DBSCAN, we choose $\alpha = [1, 1, 3, 1]$ in our experiments.

4.2 Evaluation Metrics

4.2.1 Precision, Recall and F1 Score. The precision and recall are defined as:

$$Precision = \frac{TP}{TP + FP} \quad , \quad Recall = \frac{TP}{TP + FN} \tag{10}$$

where TP, FP, FN are true-positives, false-positives and false-negatives respectively. The attribute of true or false of the sample is determined by its IoU between ground truth boxes, which is defined as $IoU = \frac{Intersection\ Area}{Union\ Area}$. For instance, with IoU threshold of 0.5, if the IoU between the sample box and any same-class ground truth box is larger than 0.5, the sample is then considered as a true one.

The F1 score is defined as $F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}$, namely, the harmonic mean. F1 score is usually used for depicting whether the model achieves a good trade-off between the recall and precision.

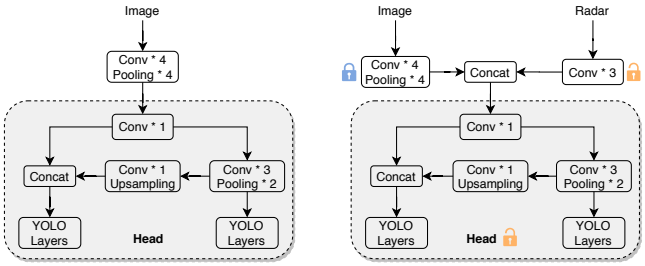


Figure 7: Left: the architecture of Tiny YOLOv3. Right: the naive deep fusion baseline used for performance comparison, which shares the same head design with Tiny YOLOv3. The blue lock means the weights are frozen.

4.2.2 mean Average Precision (mAP). For the object detection task, the mAP is a frequently-used metric to evaluate the performance of the detectors [11, 22]. mAP is the mean of per-class AP. To calculate the AP, we first rank the output boxes in descending order according to their attached confidence scores. Next, we compute the precision-recall (P-R) curve from the ranked boxes. Specifically, at the start, the top-1 box (i.e., the box with the highest confidence score) is taken to compute precision and recall, while in the k^{th} step, top-k boxes are taken and so on. In this way, pairs of precision and recall are obtained, which forms a P-R curve. The AP is then an approximation of the area below the P-R curve.

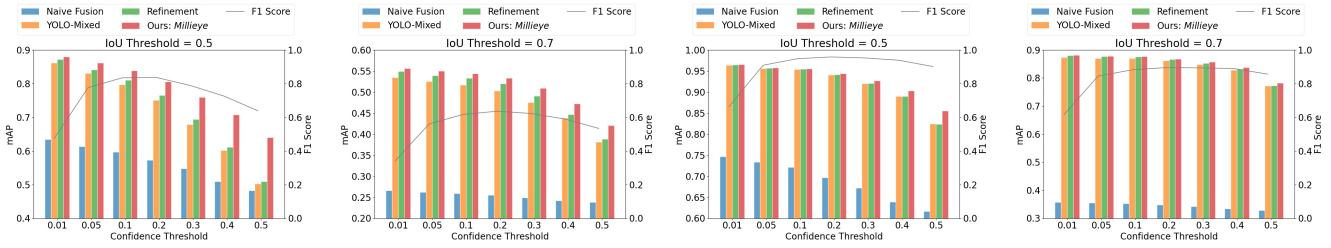
Notably, the mAP has a negative correlation with the confidence threshold because a higher confidence score truncates the P-R curve earlier and AP depends on the area below the P-R curve. As a result, a common practice is to set the confidence threshold as small as possible (e.g., 0.001) to maximize the mAP upper-bound. However, a very low confidence threshold like 0.001 will cause an intolerable number of false positives and is inapplicable to real-world applications. Instead, a moderate confidence threshold is often used to balance the recall and precision. Therefore, we may test the mAP and F1 score under different confidence thresholds to gauge the overall performance of the model.

4.3 Experiments on Radar-Camera Dataset

4.3.1 Baselines. We consider the following competing methods as baselines.

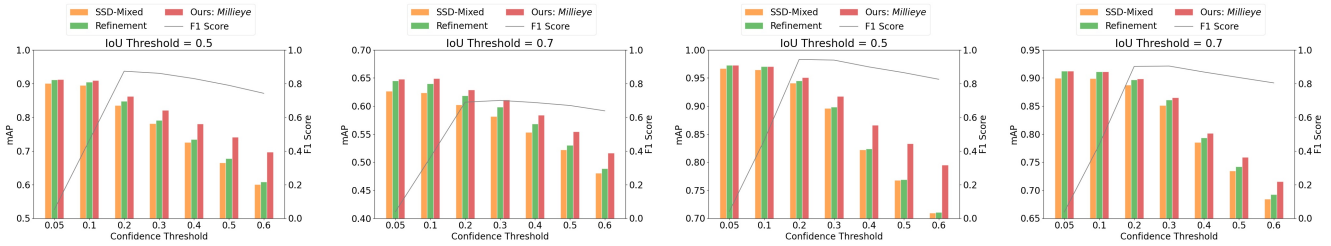
YOLO-Mixed. As a lightweight version of YOLOv3, Tiny YOLOv3 is especially suitable for mobile devices. To let the detector learn both day-time and night-time features, we train the Tiny YOLOv3 on the mixed dataset of COCO and ExDark, and name this baseline YOLO-Mixed. Since the category of our dataset (i.e. the person) is included in the 12 categories of COCO/ExDark, we can directly take YOLO-Mixed and test it on our collected dataset. This baseline also acts as an ablated model of our proposed *milliEye*. For YOLO-Mixed, only image data is used during the inference.

Naive Fusion. Recent methods on deep camera and radar fusion usually follow an end-to-end paradigm. In [3], radar and image data are directly fed into a CNN and the fusion is achieved by the concatenation operation. As a baseline, we follow this idea and design a naive deep fusion detector whose architecture is shown in Figure 7. It is similar to Tiny YOLOv3 except for an additional



(a) Low-light, IoU Threshold=0.5 (b) Low-light, IoU Threshold=0.7 (c) Ordinary-light, IoU Threshold=0.5 (d) Ordinary-light, IoU Threshold=0.7

Figure 8: The mAP of *milliEye* and three baselines. The gray curves show the F1 scores of the image-based object detector (i.e., Tiny YOLOv3). X-axes represent the confidence thresholds set to the imaged-based object detector. Note that the range of y-axes of sub-figures are different for better visualization effect.



(a) Low-light, IoU Threshold=0.5 (b) Low-light, IoU Threshold=0.7 (c) Ordinary-light, IoU Threshold=0.5 (d) Ordinary-light, IoU Threshold=0.7

Figure 9: The mAP of *milliEye* and two baselines when the image-based object detector is changed to SSD.

radar branch and the corresponding concatenation operation. Since our collected dataset is small-scale, we follow a transfer learning paradigm by initializing the weights of image feature extraction layers with YOLO-Mixed, and freezing those weights during the fine-tuning on our collected dataset, as Figure 7 shows.

Refinement. This model combines YOLO-Mixed and the entire fusion-enabled refinement head, but does not include the radar-based tracker. As the ablated version of *milliEye*, it aims to demonstrate the effectiveness of radar information in the refinement head.

4.3.2 Overall Performance. Figure 8 summarizes the overall performance of competing models on our dataset. As can be seen, the performance of naive fusion is significantly inferior to the other three methods. Although we already fixed the weights of image extraction layers for the naive fusion, the detection head which involves lots of image features still requires the training on our collected multi-modality dataset. Therefore, it has limited generalizability and fails to adapt to the test scene using the training data[2]. In contrast, YOLO-Mixed has high generalizability and achieves much higher mAP because it is trained on both day-time dataset COCO and night-time dataset ExDark, whose scales are much larger than ours, showing the importance of taking advantage of well-trained image-based object detectors.

Under ordinary light conditions, YOLO-Mixed is already satisfactory since our dataset is of low difficulty regarding object size, density and diversity. Correspondingly, the benefit of radar is not prominent. However, on the low-light sub-dataset, our proposed method outperforms the YOLO-Mixed by a large margin with the help of radar. Specifically, when the IoU threshold is low (e.g., 0.5), the majority of improvements are attributed to the radar-based tracker, meaning that the radar-based tracker picks up some ground truth boxes neglected by the image-based object detector, although not very accurate in position. When we impose a stricter requirement on the box positions (i.e., a higher IoU threshold 0.7 is set), the refinement head achieve higher improvements, validating the effectiveness of refinement head on box position adjustment. Moreover, as the confidence threshold raises, fewer candidate bounding boxes will be yielded by the image-based detector, leading to greater improvement from the radar-based tracker, as represented by the height differences between the red and green bars.

In practice, users may choose a medium confidence threshold to balance the recall and precision. According to the F1 score curves, a proper confidence threshold should be around 0.2, under which our proposed *milliEye* improves the mAP of YOLO-Mixed by 5.5 and 3.0 percent under IoU thresholds of 0.5 and 0.7, respectively. To sum up, the naive fusion method exhibits poor generalizability and adaptability when the training data is insufficient. This shortcoming hinders its application in practical scenarios. On the contrary, our proposed approach can benefit from abundant diverse image datasets and is highly adaptive to deployment scenes even only a small amount of multi-modality training data is available.

4.3.3 Compatibility to Another Object Detector. We investigate the compatibility of *milliEye* on another lightweight image-based

[2]Our 5-fold cross-validation ensures the training set and the test set are collected in different places

object detector, the VGG16-based SSD-300 [23, 33]. We reproduce the experimental settings in Section 4.3.2 except change the Tiny YOLOv3 to SSD. Particularly, we consider two baselines: the SSD trained on the mixed dataset on COCO and ExDark, named the SSD-Mixed; the ablated version of *milliEye* with the radar-based tracker removed, named the Refinement. The results in Figure 9 exhibit similar patterns with Figure 8. The mAP improvement is up to 2.9 percent for low-light scenes when the confidence threshold is 0.2, showing that *milliEye* can provide considerable performance boosts for different image-based object detectors, especially under low-light illumination conditions.

4.4 Experiments on COCO and ExDark Datasets

In the following, we validate the effectiveness of the refinement head when there are only images as inputs. As introduced in Section 4.1.1, COCO and ExDark are benchmark image datasets with high diversity and low inter-category bias. Testing on them enables us to minimize the deviation brought by the biased dataset, thus to focus on the performance and functionality of the model itself. As there is no radar data in COCO and Exdark datasets, we slightly modify the architecture of *milliEye* by removing the radar feature extraction branch as well as the fusion module, while retaining the ensemble module. YOLO-Mixed, which has been introduced in Section 4.3.2, is again taken to be a baseline. For comparison, we train a refinement head also on the mixed dataset, and use YOLO-Ref to denote the combination of YOLO-Mixed and the refinement head. Besides, Tiny YOLOv3 trained only on COCO and only on ExDark are used as additional two baselines.

In Table 3, we present the mAP of 12 categories that both the COCO and Exdark include. Within our expectation, YOLO-Mixed is superior to YOLO-COCO and YOLO-ExDark due to more training data. On both ExDark and COCO test sets, the refinement head improves the original YOLO-Mixed. Specifically, we observe that the improvements are most prominent when a medium confidence threshold is given, which fits the needs for a threshold in real world application. For example, in Table 3(a), the improvement is up to 2.4 percent when the threshold is 0.1. Since YOLO-Ref performs the refinement based on the box proposals provided by YOLO-Mixed, YOLO-Ref and YOLO-Mixed would have the same recall under the same confidence threshold[3]. We also plot the P-R curves in Figure 10. As can be seen, when the confidence threshold is 0.1, the refinement head raises the whole curve of YOLO-Mixed-0.1. To sum up, we validate that on both the ordinary-light dataset COCO and the low-light dataset ExDark, the refinement head is conducive to more accurate object detection.

4.5 Robustness to Environmental Dynamics

4.5.1 Confidence Scores from Two Modalities. To quantitatively analyze how the importance of two sensors shift under different illuminations, we calculate the average confidence scores from two sensing modalities over the whole dataset, which are represented by the two solid green squares in the fusion sub-module in Figure 3. As can be seen in Table 4, both the radar and the camera are

[3]No confidence threshold is set to the refinement head during the test

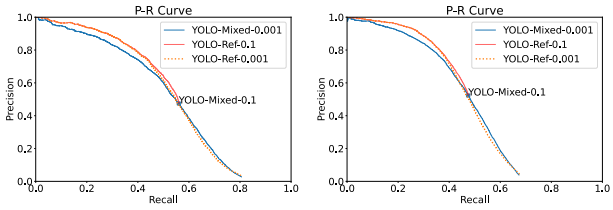
Table 3: mAPs of competing approaches: YOLO-ExDark, YOLO-COCO, YOLO-Mixed and YOLO-Ref (YOLO-Mixed + Refinement Head). We also calculate the improvement from YOLO-Mixed to YOLO-Ref for convenient comparsion. The IoU threshold is set to 0.5.

(a) On ExDark Test set

Conf. Thresh.	0.001	0.01	0.05	0.1	0.2	0.3	0.4	0.5
YOLO-ExDark	45.1	44.4	42.3	39.9	34.7	29.7	24.0	18.9
YOLO-Mixed	49.0	48.5	46.4	43.7	38.2	31.6	25.3	18.5
YOLO-Ref	**50.9**	**50.8**	**48.8**	**46.1**	**40.1**	**32.9**	**26.1**	**18.9**
Improvement	1.9	2.3	2.4	2.4	1.9	1.3	0.8	0.4

(b) On COCO Test set

Conf. Thresh.	0.001	0.01	0.05	0.1	0.2	0.3	0.4	0.5
YOLO-COCO	37.6	37.2	35.0	32.6	26.7	21.3	16.6	12.4
YOLO-Mixed	37.9	37.5	35.5	32.7	27.4	22.4	17.4	13.2
YOLO-Ref	**39.7**	**39.6**	**37.9**	**34.8**	**28.8**	**23.3**	**17.9**	**13.4**
Improvement	1.8	2.1	2.4	2.1	1.4	1.1	0.5	0.2



(a) Test on ExDark (b) Test on COCO

Figure 10: In the legend, numbers in the suffixes are confidence thresholds applied to the image-based object detector. The curve of YOLO-Mixed-0.1 almost overlaps with YOLO-Mixed-0.001 except an earlier stop at the position of the blue square. The IoU threshold is set to 0.5 (best zoomed-in).

good at suppressing negative samples, on which the generated confidence scores only show slight differences. However, with respect to positive samples, confidence scores from the camera are hugely influenced by the illumination, while confidence scores from the radar keep stable regardless of light conditions. As a result, the camera dominates the confidence in ordinary light conditions, while radar plays a more vital role in low light conditions, demonstrating *milliEye*'s robustness to different illuminations. Some visualization results of the radar information are presented in Figure 11. We can observe that regardless of the illumination conditions, the radar feature maps provide robust information about the occupancy of objects, which is coherent with the fact that confidence scores from radar are almost equivalent in ordinary-light and low-light scenes.

4.5.2 Robustness in Challenging Scenarios. Our low-light sub-dataset also includes a portion of data collected under extreme dark scenes, where the function of the camera is greatly hindered. To annotate them, we lift the exposure and the brightness as much as possible using image processing software to make objects detectable by human eyes. The testing mAP results are showed in Table 5. When

Table 4: Average Confidence scores from two sensing modalities under different illuminations.

Samples	Positive: IoU>0.5		Negative: IoU<0.5	
Light	Ordinary	Low	Ordinary	Low
Radar	0.580	0.585	0.097	0.097
Camera	0.808	0.465	0.083	0.059

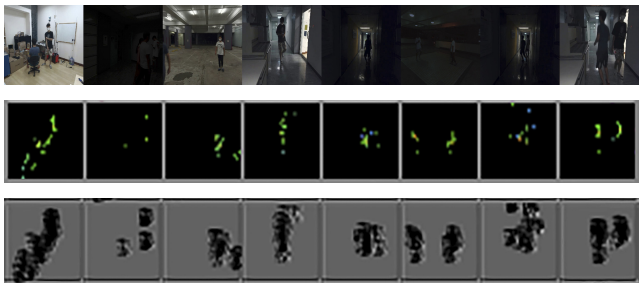


Figure 11: From top to bottom: visualization of images, radar point cloud (before embedding and CNN extraction) and the radar feature maps (after CNN extraction).

Table 5: mAP of our proposed *milliEye* under extreme dark scenes. The IoU threshold is set to 0.5.

Conf. Threshold	0.01	0.05	0.1	0.2	0.3	0.4	0.5
YOLO-Mixed	80.6	75.7	70.6	63.0	50.2	40.4	31.7
Ours: *milliEye*	**83.5**	**81.2**	**78.6**	**74.1**	**67.2**	**60.8**	**56.0**
Improvement	2.9	5.5	8.0	11.1	17.0	20.4	24.3

Figure 12: Qualitative results under very poor illumination conditions. The yellow boxes are results of our proposed *milliEye* and the white boxes are ground truth boxes. The attached numbers represent the predicted confidence scores.

choosing a confidence threshold of 0.2, which is a good balance between the precision and recall, the improvement of mAP is up to 11.1 percent, demonstrating that our *milliEye* greatly enhances the image-based YOLO-Mixed under challenging environments. Some qualitative results are exhibited in Figure 12. As can be seen, with the help pf radar, *milliEye* is able to provide correct bounding boxes even the objects are almost invisible due to darkness.

4.6 System Efficiency

4.6.1 Model Size and FLOPS. We list the number of parameters and the calculated FLOPs in Table 6. As shown, the total amounts of the parameters and the FLOPs of three additional modules are one order of magnitude smaller than those of Tiny YOLOv3, a compact image-based detector, let alone others like YOLOv3, whose FLOPs is up to 65.3B. Therefore, our proposed method incurs negligible extra compute overheads upon the existing image-based object detectors, like Tiny YOLOv3.

Table 6: Amount of parameters and float operations (FLOPs). Details of the last three sub-modules are in Figure 3.

	milliEye			
	Tiny YOLOv3	**Radar Feature Maps Generation**	**Image PS Score Maps Generation**	**Refinement Head**
Params	8.7M	0.19M	0.13M	0.15M
FLOPs	5.45B	0.19B	0.17B	0.14M × #(RoI)

Table 7: Runtime analysis (mean ± std) on different platforms. The unit is millisecond (ms) per-frame.

	Tiny YOLOv3	***milliEye***
Jetson TX2	57.6 ± 2.1	74.4 ± 3.7
Desktop PC	9.8 ± 3.2	14.2 ± 4.1

4.6.2 Execution Latency. In the last experiment, we investigate the execution latency of *milliEye* on two different platforms: the Nvidia Jetson TX2 and a desktop PC with a Xeon Gold 5117 CPU and a NVIDIA RTX2080 GPU. We implement the whole model using PyTorch on Ubuntu. During the test, we set the batch size to be 1 to mimic the inference on stream data. As shown in Table 7, *milliEye* incurs about 30% longer run time than Tiny YOLOv3. Considering that the extra FLOPs are less than 1/10 of the FLOPs of Tiny YOLOv3, we believe this portion of 30% can be reduced when a more powerful image-based object detector is used or some advanced acceleration techniques are applied to make the "fragmental" operations in the refinement head more efficient. In general, *milliEye* maintains the high processing speed of Tiny YOLOv3, and achieves real-time performance on embedded platform TX2 (>13 fps).

5 RELATED WORK

Cross-Domain Object Detection. Fully supervised object detectors like SSD [23], YOLO[30] and RetinaNet [21] have achieved great success in recent years. However, for new scenes, the large quantities of instance-level annotations they need are usually unavailable. To this end, several unsupervised cross-domain object detection methods have been proposed. Inoue et al. [15, 32] use pseudo-labeling, where a subset of the object proposals are selected and used to re-train the original model. Chen et al. [6] add a domain discriminator behind original Faster R-CNN, forcing the original detector to learn domain-invariant features. Khodabandeh et al. [18] formulate the domain adaptation as a problem of training with noisy data which enables the detector to improve itself through high confidence predictions and tracking cues. Our method provides another approach for the cross-domain object detection. With the help of a domain-invariant sensing modality (e.g., the mmWave

radar), we can boot the performance of image-based object detectors, whose predictions in turn can be used as new training data for the image-based detector thus achieving domain adaptation [4, 5].

mmWave Radar Sensing. mmWave radars have been increasingly adopted in IoT applications. For example, leveraging the accurate range sensing ability of mmWave radars, [29] proposes Osprey, a tire wear sensing system via measuring the distance difference between the tread and the groove in real-time. Jiang et al. achieve micrometer-level vibration measurements with mmWave radar [17, 35]. Leveraging the penetrability of radar signals, through-fog robust indoor mapping and high-resolution imaging are achieved [13, 25]. Radar-based multi-person identification systems are proposed in [38], where the radar could be hidden behind the furniture for the purpose of aesthetics and non-intrusion. Zhao et al. [36, 37] demonstrate the possibility of accurate through-wall human pose estimation using merely the mmWave radar. Several studies are also focused on the object detection combining mmWave radars with cameras [3, 7, 26–28]. However, these fusion solutions impose stringent requirements on both the quantity and quality on collected data and may fail to adapt to real-world deployment scenarios where only a small-scale dataset is available. Our proposed system addresses this issue by taking advantage of models trained on large image datasets, and thus is well-suited for applications that need to process highly dynamic scenes.

6 CONCLUSION AND DISCUSSION

We present *milliEye*, a lightweight robust object detection system based on mmWave radar and camera fusion. Via multimodal fusion, our system is able to improve the performance of off-the-shelf object detectors, especially in the low-illumination conditions. Moreover, *milliEye* has the ability to adapt to new scenes using a small amount of labeled data compared to naive deep-fusion-based methods. In terms of modeling, *milliEye* is compute efficient, making it suitable for edge-based real-time using. Our evaluation shows that in challenging scenarios, *milliEye* achieves an mAP of 74.1% (compared to 63.0% of Tiny YOLOv3), while only incurring an additional average delay of 16.8 ms per-frame on Jetson TX2. Nevertheless, *milliEye* exhibits some limitations worthy of further study. First, a natural extension in object detection is from 2D to 3D. The depth information from the mmWave radar is likely to play a critical role in fusion-based 3D object detection. In this paper, we encode the depth information into one of channels of the radar feature maps for 2D occupancy estimation. In the 3D detection scenario, a more advanced method to utilize the depth information can be considered. Second, in our experiments, the point cloud from the radar is not involved in the object classification. As introduced in Section 2.1, the generated point cloud via FFT algorithms suffers low angular resolution, which restricts the perception ability of radars, making radars unsuitable for classification tasks. However, by leveraging more low-level data, such as the raw output of Analog to Digital Converter (ADC), or internal range-azimuth/range-doppler maps, some fine-grained features can be extracted for classification.

REFERENCES

[1] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. 2020. Seeing Through Fog Without Seeing Fog: Deep Multimodal Sensor Fusion in Unseen Adverse Weather. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[2] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuScenes: A Multimodal Dataset for Autonomous Driving. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[3] S. Chadwick, W. Maddern, and P. Newman. 2019. Distant Vehicle Detection Using Radar and Vision. In *2019 International Conference on Robotics and Automation (ICRA)*. 8311–8317. https://doi.org/10.1109/ICRA.2019.8794312

[4] Simon Chadwick and Paul Newman. 2019. Training object detectors with noisy data. In *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1319–1325.

[5] S. Chadwick and P. Newman. 2020. Radar as a Teacher: Weakly Supervised Vehicle Detection using Radar Labels. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 222–228. https://doi.org/10.1109/ICRA40945.2020.9196855

[6] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. 2018. Domain Adaptive Faster R-CNN for Object Detection in the Wild. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[7] Hyunggi Cho, Young-Woo Seo, BVK Vijaya Kumar, and Ragunathan Raj Rajkumar. 2014. A multi-sensor fusion system for moving object detection and tracking in urban driving environments. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1836–1843.

[8] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. 2016. R-FCN: Object Detection via Region-based Fully Convolutional Networks. In *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett (Eds.). Curran Associates, Inc., 379–387. http://papers.nips.cc/paper/6465-r-fcn-object-detection-via-region-based-fully-convolutional-networks.pdf

[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[10] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* (Portland, Oregon) *(KDD'96)*. AAAI Press, 226–231.

[11] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. 2015. The Pascal Visual Object Classes Challenge: A Retrospective. *International Journal of Computer Vision* 111, 1 (Jan. 2015), 98–136.

[12] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. 2020. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems* (2020).

[13] Junfeng Guan, Sohrab Madani, Suraj Jog, Saurabh Gupta, and Haitham Hassanieh. 2020. Through Fog High-Resolution Imaging Using Millimeter Wave Radar. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. 2017. Mask R-CNN. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 2980–2988.

[15] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. 2018. Cross-Domain Weakly-Supervised Object Detection Through Progressive Domain Adaptation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[16] Cesar Iovescu and Sandeep Rao. 2017. The fundamentals of millimeter wave sensors. *Texas Instruments, SPYY005* (2017).

[17] Chengkun Jiang, Junchen Guo, Yuan He, Meng Jin, Shuai Li, and Yunhao Liu. 2020. MmVib: Micrometer-Level Vibration Measurement with Mmwave Radar. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking* (London, United Kingdom) *(MobiCom '20)*. Association for Computing Machinery, New York, NY, USA, Article 45, 13 pages. https://doi.org/10.1145/3372224.3419202

[18] Mehran Khodabandeh, Arash Vahdat, Mani Ranjbar, and William G. Macready. 2019. A Robust Learning Approach to Domain Adaptive Object Detection. In *The IEEE International Conference on Computer Vision (ICCV)*.

[19] H. W. Kuhn and Bryn Yaw. 1955. The Hungarian method for the assignment problem. *Naval Res. Logist. Quart* (1955), 83–97.

[20] Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. 2017. Light-Head R-CNN: In Defense of Two-Stage Object Detector. *ArXiv* abs/1711.07264 (2017).

[21] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. 2017. Focal Loss for Dense Object Detection. In *The IEEE International Conference on Computer Vision (ICCV)*.

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (Eds.). Springer International Publishing, Cham, 740–755.

[23] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. 2016. SSD: Single Shot MultiBox Detector. In *Computer Vision – ECCV 2016*, Bastian Leibe, Jiri Matas, Nicu Sebe, and Max

Welling (Eds.). Springer International Publishing, Cham, 21–37.

[24] Yuen Peng Loh and Chee Seng Chan. 2019. Getting to Know Low-light Images with The Exclusively Dark Dataset. *Computer Vision and Image Understanding* 178 (2019), 30–42. https://doi.org/10.1016/j.cviu.2018.10.010

[25] Chris Xiaoxuan Lu, Stefano Rosa, Peijun Zhao, Bing Wang, Changhao Chen, John A. Stankovic, Niki Trigoni, and Andrew Markham. 2020. See through Smoke: Robust Indoor Mapping with Low-Cost MmWave Radar. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services* (Toronto, Ontario, Canada) *(MobiSys '20)*. Association for Computing Machinery, New York, NY, USA, 14–27. https://doi.org/10.1145/3386901.3388945

[26] M. Meyer and G. Kuschk. 2019. Deep Learning Based 3D Object Detection for Automotive Radar and Camera. In *2019 16th European Radar Conference (EuRAD)*. 133–136.

[27] Ramin Nabati and Hairong Qi. 2019. RRPN: Radar Region Proposal Network for Object Detection in Autonomous Vehicles. In *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 3093–3097.

[28] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp. 2019. A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection. In *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. 1–7. https://doi.org/10.1109/SDF.2019.8916629

[29] Akarsh Prabhakara, Vaibhav Singh, Swarun Kumar, and Anthony Rowe. 2020. Osprey: a mmWave approach to tire wear sensing. In *Proceedings of the 18th International Conference on Mobile Systems, Applications, and Services*. 28–41.

[30] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[31] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 91–99.

[32] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. 2019. Automatic Adaptation of Object Detectors to New Domains Using Self-Training. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[33] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *International Conference on Learning Representations*.

[34] X. Wang, L. Xu, H. Sun, J. Xin, and N. Zheng. 2016. On-Road Vehicle Detection and Tracking Using MMW Radar and Monovision Fusion. *IEEE Transactions on Intelligent Transportation Systems* 17, 7 (2016), 2075–2084.

[35] Binbin Xie, Jie Xiong, Xiaojiang Chen, and Dingyi Fang. 2020. *Exploring Commodity RFID for Contactless Sub-Millimeter Vibration Sensing*. Association for Computing Machinery, New York, NY, USA, 15–27. https://doi.org/10.1145/3384419.3430771

[36] Mingmin Zhao, Tianhong Li, Mohammad Abu Alsheikh, Yonglong Tian, Hang Zhao, Antonio Torralba, and Dina Katabi. 2018. Through-Wall Human Pose Estimation Using Radio Signals. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[37] 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* (Budapest, Hungary) *(SIGCOMM '18)*. ACM, New York, NY, USA, 267–281. https://doi.org/10.1145/3230543.3230579

[38] Peijun Zhao, Chris Xiaoxuan Lu, Jianan Wang, Changhao Chen, Wei Wang, Niki Trigoni, and Andrew Markham. 2019. mID: Tracking and Identifying People with Millimeter Wave Radar. In *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 33–40.