Capricorn: Towards Real-time Rich Scene Analysis Using RF-Vision Sensor Fusion

Ziqi Wang, Ankur Sarker, Jason Wu, Derek Hua, Gaofeng Dong, Akash Deep Singh, Mani Srivastava
Electrical and Computer Engineering Department, University of California, Los Angeles
Los Angeles, California 90095, United States of America
wangzq312@g.ucla.edu,as4mz@virginia.edu,{jaysunwu,derekhua,gfdong,akashdeepsingh,mbs}@ucla.edu

ABSTRACT

Video scene analysis is a well-investigated area where researchers have devoted efforts to detect and classify people and objects in the scene. However, real-life scenes are more complex: the intrinsic states of the objects (e.g., machine operating states or human vital signals) are often overlooked by vision-based scene analysis. Recent work has proposed a radio frequency (RF) sensing technique, wireless vibrometry, that employs wireless signals to sense subtle vibrations from the objects and infer their internal states. We envision that the combination of video scene analysis with wireless vibrometry form a more comprehensive understanding of the scene, namely "rich scene analysis". However, the RF sensors used in wireless vibrometry only provide time series, and it is challenging to associate these time series data with multiple real-world objects. We propose a real-time RF-vision sensor fusion system, Capricorn, that efficiently builds a cross-modal correspondence between visual pixels and RF time series to better understand the complex natures of a scene. The vision sensors in Capricorn model the surrounding environment in 3D and obtain the distances of different objects. In the RF domain, the distance is proportional to the signal time-of-flight (ToF), and we can leverage the ToF to separate the RF time series corresponding to each object. The RF-vision sensor fusion in Capricorn brings multiple benefits. The vision sensors provide environmental contexts to guide the processing of RF data, which helps us select the most appropriate algorithms and models. Meanwhile, the RF sensor yields additional information that is originally invisible to vision sensors, providing insight into objects' intrinsic states. Our extensive evaluations show that Capricorn real-timely monitors multiple appliances' operating status with an accuracy of 97%+ and recovers vital signals like respirations from multiple people. A video (https://youtu.be/b-5nav3Fi78) demonstrates the capability of Capricorn.

CCS CONCEPTS

• **Human-centered computing** → *Ubiquitous and mobile computing systems and tools*; • **Computer systems organization** → *Real-time system architecture.*

ACM Reference Format:
Ziqi Wang, Ankur Sarker, Jason Wu, Derek Hua, Gaofeng Dong, Akash Deep Singh, Mani Srivastava. 2022. Capricorn: Towards Real-time Rich Scene Analysis Using RF-Vision Sensor Fusion. In *ACM Conference on Embedded Networked Sensor Systems (SenSys '22), November 6–9, 2022, Boston, MA, USA.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3560905.3568504

SenSys '22, November 6–9, 2022, Boston, MA, USA

ACM ISBN 978-1-4503-9886-2/22/11.
https://doi.org/10.1145/3560905.3568504

1 INTRODUCTION

I. Motivation. Situation awareness is an operator's internalized mental model of its surrounding environment [20], which is critical for the safety and convenience of autonomous robots or human decision-makers (e.g., first responders) [31]. Building situation awareness requires an understanding of the states of the objects in the scene, which can be divided into two major categories: extrinsic state and intrinsic state. The *extrinsic state* refers to the visually observable properties of an object (e.g., object type, shape, color, and location), which are often acquired by vision sensors like cameras. On the flip side, the *intrinsic state* is concomitant with the internal physical or biological activities of that object (e.g. machine operating status or human health conditions). Simultaneous estimation of both extrinsic and intrinsic states of the objects results in a *rich scene analysis*; hence, it boosts the construction of situational awareness.
II. State-of-the-art and Challenges. Scene analysis using objects' extrinsic states has been a well-studied research area. With advances in computer vision technologies, video scene analysis systems [1] can effectively perform object detection and recognition [7, 51, 52] or semantic segmentation [37, 47, 64, 79]. However, objects' intrinsic states are usually hidden from off-the-shelf vision sensors. While there are exceptions like vibration sensing using high-speed cameras [14, 60] or blood pulse sensing from video [32], these systems require special devices or settings that are generally not available in real life. The intrinsic states are usually measured by attaching a sensor (e.g., IMU sensor [50], ECG sensor [50], PPG sensor [68], or geophones [26]) to the sensing targets. However, attaching sensors to objects is not always possible since the process can be burdensome, and sometime we may not have control of the object.

Recent research has proposed using wireless sensing technologies to remotely sense the objects' intrinsic states, where we emit a traveling wave to the objects and collect the reflected responses. In this case, we can use the reflected wave to sense vibrations (i.e., tiny motions), which reveals information about the intrinsic states. This technology is known as *wireless vibrometry*. Some wireless vibrometry work focused on high-frequency physical phenomenons such as vibration generated by industrial machines or household appliances using millimeter wave (mmWave) [27], radio-frequency identification (RFID) [35]), ultra-wideband (UWB) radar [65] or laser [81]. Meanwhile, others focused on sensing lower-frequency physical phenomenons such as human heartbeat and respiration rate using WiFi [63], frequency-modulated continuous-wave radar (FMCW) [2, 80] or UWB radar [83].

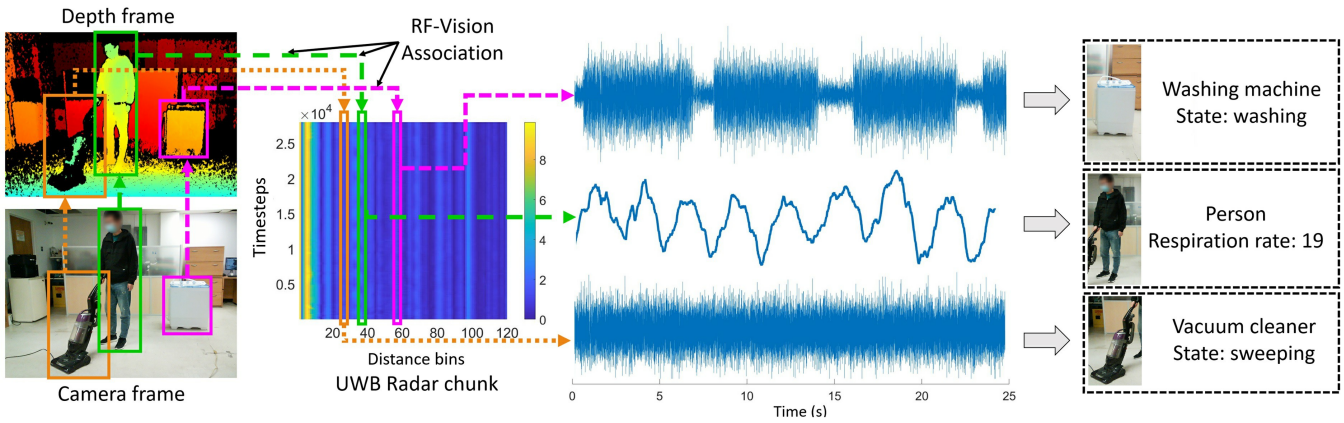

Figure 1: A high-level overview of Capricorn. Information obtained from the depth sensor serves as a bridge for Capricorn to associate RF times series data with camera pixel data. This RF-vision association allows Capricorn to choose the appropriate algorithms using object type information and infer object intrinsic states in-real time.

While the existing radio-frequency (RF) sensing systems can provide inferences about an object's intrinsic states via wireless vibrometry, they face two major challenges. *First*, the existing wireless vibrometry sensing systems presuppose the existence of a particular type of object (e.g., a person) in the scene to process the signal accordingly, facing adaptability issues if this presumption fails. For instance, WiFi-based wireless vibrometry systems have been used to extract acoustic vibrations [35], count human breaths [63], or recognize human gaits [61]. Since processing methods are vastly different for each signal type, these systems often assume the existence of a particular sensing target in the scene and use a fixed, non-adaptive pipeline. *Second*, the existing vibrometry systems have difficulties efficiently handling a dynamic number of objects in real-time. Early RF sensing research [55, 62, 78] focused on only a single target. Recent work has been proposed using blind source separation like a independent component analysis [76, 77] or a blind search in the angle [72] or distance [65] to handle multiple objects. However, in a dynamic environment, the number of target objects is indefinite and often changes. Without the prior knowledge of the number or location of objects in the scene, these techniques rely on empirical thresholds that are prone to missing objects with a low signal-to-noise ratio (SNR) or perceiving nonexistent "ghost" objects. Moreover, these algorithms are often computationally intensive and struggle in resource-constrained devices in real-time.

III. Proposed System. We propose *Capricorn*, a real-time RF-vision sensor fusion system to realize our vision of rich scene analysis. The combination of video scene analysis and wireless vibrometry allows us to better understand the complex natures of a scene. Our system is a special case of general multlimodal sensor fusion systems consisting of vision sensors (cameras and depth sensors) and RF sensors (UWB radar). The key methodology of Capricorn is to utilize the shared geometry information, namely distance, as a bridge to connect RF signals with the visual world. An RF sensor can estimate target distances using signal time-of-flight (ToF). Meanwhile, the distance of the object can be also measured using a depth sensor aligned with the camera. We can then build associations between camera pixels and RF time series using the shared distance. This RF-vision association brings chances to solve the challenges mentioned in the previous paragraph. To handle the *first challenge*, the proposed system leverages the information from the vision sensor as a prior to choose the most suitable filters and machine learning models for processing the RF signals. This scheme allows Capricorn to use a single RF sensor to make various types of inferences of objects and identify their intrinsic states. To handle the *second challenge*, Capricorn first aligns the depth sensor and the camera to form a 3D model of the world and then estimates the distance of each object in the scene. With the distance of each object known, Capricorn can separate the RF signal coming from different objects using ToF without any blind search or blind decomposition technique.

Figure 1 shows an overview of the proposed system in a futuristic smart-home, monitoring the appliance usage and health status of the inhabitants. *First,* the extrinsic sensing pipeline in Capricorn uses object detection, classification, and tracking algorithms to infer the object types and their locations (i.e., bounding boxes and distance) in the scene. *Second*, the intrinsic sensing pipeline in Capricorn uses the distance information estimated above to extract vibration signals for each object from a three-dimensional RF data stream (i.e., time, distance, and intensity). Finally, Capricorn leverages the vibration signals and the object type information from the camera to estimate the objects' intrinsic states (i.e., machine operating states and human respiration rate).

To evaluate Capricorn, we implement the proposed system using Robot Operation System (ROS) and conduct extensive evaluations in several real-world scenarios. First, we place Capricorn in workshop and living room environments for the task of multi-appliance usage detection. Capricorn robustly detects not only the objects in the scenes and their corresponding distances but also identifies the intrinsic states of these objects. Our results show that the system can estimate the operating states of machines with an accuracy of more than 97%. Second, we demonstrate that the sensing capability of Capricorn can benefit complex event detection applications by recognizing a richer set of simple atomic events. We also evaluate Capricorn quantitatively regarding latency. With the fusion between vision, RF, and depth sensors, Capricorn generates inferences about both the object's intrinsic and extrinsic states in less than 200 ms. Third, we showcase that the same system can be applied for the health monitoring of multiple individuals. Capricorn

can simultaneously calculate multiple person's reparation rates at a mean error of 1.06 breath-per-minute.

Through Capricorn, we present a novel problem of *rich scene analysis* to the sensing community, where we simultaneously estimate the extrinsic and intrinsic states of multiple objects in real-time. Apart from the multimodal sensor system integration, the major contributions of this work are as follows:

- ***RF-vision Association.*** The addition of RF sensing to scene analysis systems allows us to make inferences about objects' internal states, which are originally invisible to vision sensors. To achieve this goal, Capricorn establishes a cross-model association between RF time series and visual pixels using the shared distance estimation from both the RF and vision sensors.
- ***Context-awareness.*** The proposed technique enables the adaptive processing of RF signals based on the environmental context provided by vision pipelines. These contexts (e.g., object types and numbers) allow us to modularly implement signal processing algorithms and internal state classifiers for each object type and then adaptively select the most suitable modules. The context information enhances state classification accuracy and speed by reducing search space [11].
- ***Real-time System.*** The key idea in Capricorn 's architecture design is that the inferences from one sensor can serve as the prior information for processing another sensor. The object detection results allow Capricorn to estimate the distance of objects in the scene using the depth map without clustering or background subtraction. The estimated distances then serve as the prior information to separate the RF data for individual objects without any blind search. These optimizations reduce the latency and enable Capricorn to perform the rich scene analysis in real-time. We also provide a multi-view version of Capricorn that employs a network of sensors viewing a scene from different angles to demonstrate the system's scalability. The implementation of Capricorn is open-sourced (https://github.com/nesl/Capricorn).

2 SYSTEM DESIGN

In this section, we first talk about our choice of sensors and then provide an overview of Capricorn. Afterwards, we present three primary units of the proposed system with appropriate figures, descriptions, and algorithms: multimodal data collection, sensor fusion (i.e., extrinsic and intrinsic object state estimations), and information storage. Finally, we scale up Capricorn by using multiple sensor nodes to cover a wider range of views.

2.1 Multimodal Sensors

As we have discussed in Section 1, a rich scene analysis system for situational awareness should simultaneously estimate intrinsic and extrinsic objects states in the scene. To achieve this goal, we carefully choose a set of complementary sensors.

Firstly, as a unique sensing modality that perceives the world similarly to a human, the camera is pervasive in intelligent systems for scene analysis. We included a camera in Capricorn to leverage the previous accomplishments in the domain of video scene analysis. For intrinsic states sensing, we utilized impulse-radio ultra-wideband (UWB) radar as the RF sensing device. The UWB radar is capable of recovering vibrations from multiple objects simultaneously in the same scan [65]. This capability is essential when analyzing a complex scene with many objects. Furthermore, UWB radar is both energy and cost efficient. The typical power consumption of a UWB radar development board is expected to be 600 mW [69], which costs one-third of a mmWave radar unit (1730-2100 mW) [58]. With UWB technologies integrated on flagship cellphones like iPhone, UWB radar is also emerging as a mobile computing sensor of choice.

However, wireless sensing modalities like UWB produces only multiple time series, and it is difficult to build a one-on-one association between these time series with the images of real-world objects. As discussed before in the introduction, we use the shared geometrical information (e.g., distance) to combine the RF and vision modalities. UWB radars can perform decimeter-level ranging as it works with pulses of a wide bandwidth. In other words, UWB radar provides distance information alongside vibrations. On the vision end, we combine a camera and a depth sensor that provide aligned depth and color (RGB) frames to models their surrounding environments in 3D. One of such possible choices is the Intel Realsense Depth Camera that is sufficiently miniaturized and reasonably priced. We considered to use only a 2D camera for the extrinsic sensing, since there are also some works on depth estimation using the image of a single camera. However, these works are still less mature to be applicable in Capricorn. For example, UWB sensors typically obtain a precision of 5 cm and resolution of 10 cm [65]. Meanwhile, some recent works on depth estimation using a monocular camera only achieve an average error of 20.3 cm [42, 43], which may incur misalignment when we connect images to their corresponding UWB time series. Also, monocular camera depth estimations are mostly learning-based that relies heavily on training data. These models may not generalize well in unseen indoor environments since the original image only contains partial relative depth information. In summary, we pick UWB radar as the intrinsic sensor for the extraction of multiple subjects' vibrations, and we employ a RGBD camera as the extrinsic sensors for a complimentary 3D modeling of the scene.

2.2 Proposed Architecture

The architecture of Capricorn consists of three different units: data collection, sensor fusion, and information storage. Figure 2 shows the overall architecture as blocks and interactions between different units as directed arrows.

- *First*, the data collection unit collects raw sensor data from different sensors simultaneously and feeds the data into the sensor fusion unit of Capricorn in a publish–subscribe (Pub-Sub) pattern. A comprehensive description of the data collection process is discussed in Section 3.
- *Second*, the sensor fusion unit processes the sensor data and estimates the extrinsic and intrinsic object states. There are two separate pipelines. The *extrinsic sensing pipeline* utilizes the vision sensor to detect and track the objects in the scene. The *intrinsic sensing pipeline* utilizes RF sensors to detect the intrinsic states of different objects in the scene. These pipelines are bridged by the common object distance information (Section 4 and 5).
- *Third*, the information storage unit generates an in-memory table to store the inference results of the sensing data and facilitate the fusion between different modalities. It also allows for the visualization functionalities of Capricorn (Section 6).

As shown in Figure 2, Capricorn takes a late fusion approach: the vision sensors first make inferences on their own, and then these inference results are shared with the wireless sensors by updating and querying a shared in-memory database. There are three distinct situations where the fusion takes place. *First,* the object bounding boxes estimated from the camera images are leveraged to estimate the distance of the objects in these bounding boxes when we align the depth frames with the camera frames (to be introduced in Section 4.3). This information helps Capricorn reduce the searching space of the distance estimation algorithm without any usage of background subtraction or clustering algorithms [21–23, 39]. *Second,* once the LiDAR camera have built a 3D model of the surrounding world and identifies the objects of interest, the distances of these objects are used by the UWB radar to identify and separate objects' RF signals. This process avoids blindly searching all the possible distances in the UWB data matrix and reduces the time complexity (see Sec. 5.1). *Finally,* Capricorn uses the object type information (from the extrinsic sensing pipeline) to choose a specific object state estimation algorithm in the intrinsic sensing pipeline. Instead of being a single model, the object state estimator in Capricorn is a collection of multiple modular models and algorithms, each responsible for a different type of object. The extrinsic sensors make inferences about the object type, which serves as an important context for the intrinsic sensing pipeline to pick the most suitable signal processing module. This modular design simplifies the accuracy requirements on the object state classifiers by reducing the complexity of its decision boundaries (details to follow in Sec. 5.3).

3 MULTIMODAL DATA COLLECTION UNIT

In the proposed system, there are three drivers to collect the vision, depth, and RF sensor data, respectively. The first two drivers process the raw data coming from the camera and depth sensors, which acquire both the camera and depth frames at a rate of 30 fps. The camera frames are RGB pixel matrices, and the depth frames contain the absolute distance (in meters) as a matrix. The RGB and depth frames are aligned together as follows. Let us consider $\mathbf{K_c}$ and $\mathbf{K_d}$ representing parameters of vision and depth sensors, respectively; and $\mathbf{T_{cd}}$ is the transformation matrix between the RGB plane and the depth plane. All the three transforms are predetermined by the sensor manufacturing and placements. Let us also use $[x, y, d]^T$ to represent a pixel coordinate in the depth plane where d is the depth value at $[x, y]^T$. Then, the corresponding pixel value $[x^{'}, y^{'}, d^{'}]^T$ in the aligned depth plane can be calculated by projecting original $[x, y, d]^T$ to the 3D space and applying the transformation $\mathbf{T_{cd}}$, and then projecting back into the image coordinate as follows [16]:

$$d^{'}[x^{'}, y^{'}, 1]^T = \mathbf{K_c}\mathbf{T_{cd}}\mathbf{K_d}^{-1} d[x, y, 1]^T \quad (1)$$

Afterwards, the two aligned frames are published as the RGB and depth streams, respectively.

The third driver collects the UWB radar data. This driver runs two separate threads to smoothly stream the RF frames to the sensor fusion unit. The first thread listens for a hardware interrupt indicating a new frame's arrival, allocates a memory space for the new frame, and pushes the new frame into a shared buffer. The second thread streams the RF frame out of that shared buffer once the buffer is full.

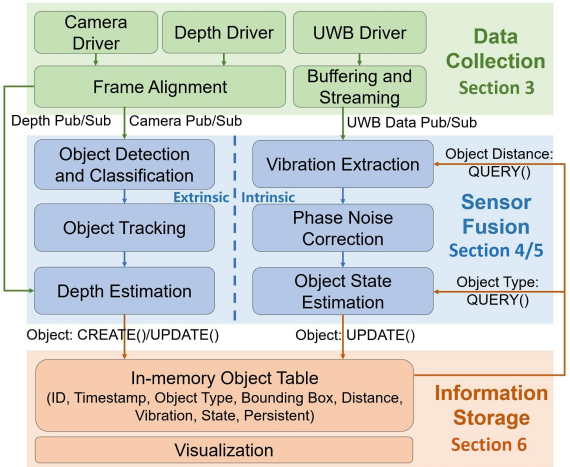

Figure 2: An overview of Capricorn.

4 SENSOR FUSION UNIT: EXTRINSIC SENSING

There are two processing pipelines in the sensor fusion unit: extrinsic sensing and intrinsic sensing. The extrinsic sensing pipeline utilizes the object detection and tracking algorithms to get the object types, distances, and bounding boxes in the scene (the current Section). The intrinsic sensing pipeline utilizes a three-dimensional RF stream (i.e., time, depth, and intensity) and the distance information from the extrinsic pipeline to obtain the vibration patterns of different objects (Section 5). In the following sections, we discuss these two pipelines and the overall sensor fusion mechanism.

While existing wireless vibrometry systems [35, 63, 65, 72, 81] provide a new perspective to infer an object's internal states via vibrations, such systems experience challenges detecting and classifying objects. Also, the RF time series processing depends a lot on the target object type because of the nature of the signals they produce. For example, the algorithms for classifying machine vibration patterns and human respirations require very different filtering and machine learning models. Therefore, we leverage vision sensors' advantage to provide a dynamic context for RF data processing. The extrinsic property sensing pipeline aims to (1) detect each object's location in the 3D space, (2) classify each object, and (3) keep track of these detected objects over time.

4.1 Object Detection and Classification

For object detection and classification purposes, we process the RGB images using a state-of-the-art object detection and classification model, YOLOv5 [28]. The object detection and classification model predicts classes and their corresponding bounding boxes in the scene. From each RGB frame, we get a vector I_j for each detected object o_j as follows:

$$I_j = [o_j, x_j, y_j, w_j, h_j, t_j]^T \quad (2)$$

where o_j is the object type, (x_j, y_j, w_j, h_j) defines its bounding box, and t_j is the timestamp. For a single object in the scene, YOLOv5 can generate multiple overlapping boxes. We apply the soft non-max suppression algorithm [8] to remove redundant boxes while preserving the bounding boxes for visually overlapped objects.

4.2 Object Tracking

The proposed system then applies an object tracking algorithm on the detected objects in the scene. The tracker learns the velocity

model (i.e., (v_x, v_y)) of each object in the scene using a Kalman Filter [5]. Let us consider that the Kalman Filter maintains a state vector $s_{j,N-1}$ for the previously seen j^{th} object from the previous $N-1$ frames as follows:

$$s_{j,N-1} = [x_j, y_j, v_j^x, v_j^y, w_j, h_j, t_j]^T \quad (3)$$

where v_x and v_y are the velocities along x and y axis. In the N^{th} frame, the tracker receives the detected object vector I_j for that object j and generates the state vector $s'_{j,N}$ using the Kalman Filter. Then, the Kalman Filter adjusts the state vector $s'_{j,N}$ to $s_{j,N}$ according to the observation I_j as follows:

$$s_{j,N} = s'_{j,N} + \mathbf{K}(I_j - \mathbf{H}s'_{j,N}) \quad (4)$$

where $\mathbf{K} \in \mathbb{R}^{7\times5}$ is the Kalman gain matrix, and $\mathbf{H} \in \mathbb{R}^{5\times7}$ is the observation model matrix.

The ultimate goal of the tracker is to associate the YOLO-detected bounding boxes in the current frame with the existing objects in the previous frames (whose information is kept by Kalman Filters). The trackers use the Hungarian algorithm to find the optimal associations between the Kalman-predicted bounding boxes and the YOLO-detected boxes, where the Intersection-over-Union (IoU) is used as a metric to measure the distance between any pair of bounding boxes.

The tracker continues predicting the states of an object even if it fails to associate it with any detected object in the subsequent frames. If the association fails successively for next T_{max} frames, the tracker assumes the object is no longer present in the scene.

4.3 Distance Estimation

In the third step of the extrinsic sensing pipeline, we fuse the depth map with the inference results (bounding boxes) generated by the object tracker to estimate the 3D coordinates of objects. Specifically, the bounding boxes serve as prior information to reduce the complexity of this distance estimation process. Without any prior information, we have to manually search the whole space and apply clustering algorithms to discover candidate objects and their distances. The predicted bounding boxes help reduce the search space by "drawing attention" to particular regions on the depth map.

Let us consider that we get the detected object state vector I_j (as in Equation (2)) where (x_j, y_j) and (w_j, h_j) define a bounding box. We can directly apply these bounding boxes to the depth map, as the color frame and the depth frame are aligned in Section 3 (see Figure 3). Within each bounding box, we draw a histogram of each pixel's depth value. On the histogram, we first remove all zero values, and then we set a threshold to filter out the background points, and select the depth of the most significant peak as the estimated depth of the object. Now, for each object j, we have obtained its location (x_j, y_j) in pixels and its depth Z_j in meters under the camera coordinate system. With the known camera intrinsic matrix $\mathbf{K_c} \in \mathbb{R}^{3\times4}$, we can estimate the object's 3D coordinates using

$$[x_j, y_j, 1]^T = \mathbf{K_c}[X_j, Y_j, Z_j, 1]^T. \quad (5)$$

We can solve for X_j and Y_j to obtain the 3D coordinate of object j, and then calculate the object's euclidean distance $d_j = \sqrt{X_j^2 + Y_j^2 + Z_j^2}$. The object distance d_j will be used later to build an association between camera pixels and the UWB radar time series. Now, at the end of the extrinsic sensing pipeline, we have obtained the objects' images (bounding boxes), coordinates in 3D space, and predicted types.

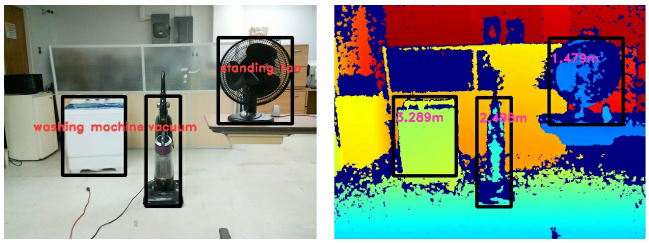

Figure 3: The extrinsic sensing pipeline of Capricorn. (Left) Object types and corresponding bounding boxes. (Right) Distance estimation for the detected objects.

5 SESNOR FUSION UNIT: INTRINSIC SENSING

The existing RF sensor data processing algorithms [65, 67, 72] primarily utilize blind search techniques during intrinsic state estimation due to the lack of object location information. As a result, they are computationally intensive and error-prone. To overcome this challenge, Capricorn firstly uses the estimated distance (as shown in Section 5.1) to extract each object's vibration data from the raw UWB signal in real-time. Then, Capricorn applies the phase noise correction on the extracted vibration data (Section 5.2). Finally, Capricorn estimates the intrinsic states of each object using the most suitable signal processing and machine learning algorithm, guided by the object type information (Section 5.3).

5.1 Vibration Extraction

The vibration extraction module first receives chunks of RF sensor stream data from the multimodal data collection unit. The received data is a two dimensional matrix $\mathbf{M} \in \mathbb{C}^{D\times T}$, where D represents the sensor-target distances, and T stands for time steps. The sensor-target distances are discretized into several distance bins.

Without any prior knowledge of the object locations, existing wireless vibrometry systems [65, 67, 72] rely on a blind search mechanism to locate the distance bins containing vibrations, sequentially processing all the distance bins. For instance, UWHear [65] had to apply phase noise correction (to be introduced later), and numerous filters to remove the reflections caused by static objects. Then, UWHear calculated the Herfindahl-Hirschman index (HHI) for each distance bin. The distance bins with vibrating objects tend to have high HHI values because their frequency spectrums contain dominant frequency components. The entire time complexity is $O(n \cdot Tlog(T))$, where n is the number of total distance bins (e.g., $n = 120$ when the sensing range is 6 meters). Aside from the time complexity, UWHear also suffers drastically in a dynamic scene where the number of objects is not predetermined. If there are k objects in the scene, the top-k distance bins with the highest HHI index can be selected. However, in a dynamic scene with a varying number of objects, the authors had to rely on an error-prone empirical threshold to choose the desired distance bins.

In Capricorn, we address the aforementioned issues using the object distance information gathered from the extrinsic sensing pipeline. For an object j whose distance is d_j, we can locate its

vibrations at the distance bin η_j as follows:

$$\eta_j = \frac{d_j - d_0}{g_0} + adj. \tag{6}$$

Here g_0 is the granularity of distance bins (i.e., how much distance each bin covers). For our hardware setup, we have $g_0 = 0.0514m$ [69]. The first d_0 meters in the received data are usually discarded to minimize the interference of the signal leaking directly from the UWB transmitter to the receiver. adj is an adjustment term to compensate for the UWB radar's distance estimation error (which is theoretically 10.71 cm). We set $adj \in [-2, 2]$ based on a set of empirical calibration measurements.

Once we calculate η_j for each object, we can take a slice from the RF data matrix $\mathbf{M}$ in the distance axis, and extract a vibration signal $\mathbf{V}_j(t) \in \mathbb{C}^T$, such that:

$$\mathbf{V}(t) \leftarrow Slice(\mathbf{M}|D = \eta_j). \tag{7}$$

Ideally, this slicing operation gives us a few UWB time series containing the vibration information of the objects. Any further processing (i.e., phase noise correction and filtering) can be applied to only these chosen UWB time series instead of the entire matrix $\mathbf{M}$. Here, the complexity of the vibration extraction algorithms drops from $O(n \cdot TlogT)$ to $O(m \cdot TlogT)$, where m is the number of the objects of interest, n is the number of total distance bins, and $m << n$.

For our exemplar scene, the vibration profiles (spectrogram of the selected time series) of the three appliances are shown in Figure 4. From the spectrograms, it is visually apparent that in this exemplar scene, the washing machine is in washing mode, the vacuum cleaner is sweeping, while the fan is on speed three.

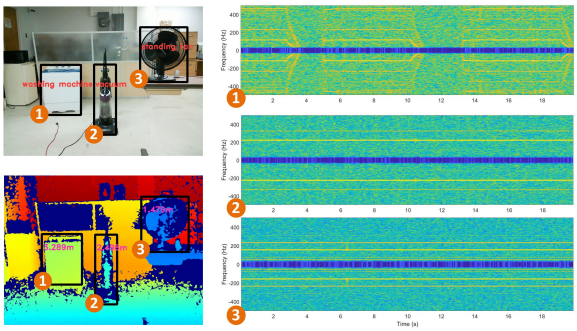

Figure 4: Spectrogram extracted from the three objects using an UWB radar sensor.

5.2 Phase Noise Correlation

Next, Capricorn applies a phase noise correction on the extracted vibration time series $\mathbf{V}(t)$. Usually, the UWB raw data matrix $\mathbf{M}$ is gradually collected frame by frame. *Phase noise correction* aims to mitigate the displacements across frames [3].

To correct for the phase noise, Capricorn first selects a reference bin $\mathbf{R}(t) \in \mathbb{C}^T$ that does not contain vibrations. Typically, the first bin is the reference bin since it is very close to the sensor and unlikely to contain any vibrations. Then, Capricorn calculates the mean phase of this reference bin, denoted as $\overline{\phi}$. For each subsequent time step, Capricorn calculates the phase difference with respect to $\overline{\phi}$ as follows:

$$\Delta\phi(t) = \overline{\phi} - phase(R(t)), t \in [0, T]. \tag{8}$$

Finally, the proposed system applies the phase noise correction to any vibration signal $\mathbf{V}(t) \in \mathbb{C}^T$ using the following:

$$\tilde{V}(t) = V(t) \exp^{\Delta\phi(t)}. \tag{9}$$

5.3 Object State Estimation

The last module of the intrinsic sensing pipeline is the object (intrinsic) state estimation. Capricorn estimates both discrete (e.g., appliance usages) and continuous (e.g., vital signals) types of intrinsic object states. Unlike previous wireless vibrometry work [65, 67, 83], Capricorn does not rely on any pre-assumptions about the existence of a particular type of object in the scene. The proposed system adaptively selects the most appropriate signal processing and machine learning algorithm to estimate the intrinsic states given the object type information. Capricorn utilizes two types of estimators. The first one is a discrete state estimator, built with the Support Vector Machine (SVM) algorithm. The second one is a continuous state estimator, built with the Variational Mode Decomposition (VMD) algorithm.

5.3.1 ***Discrete State Estimation.*** The extracted vibration signal $\tilde{V}(t)$ after the phase noise correction still contains static components and additive noise. As the first step of discrete state estimation, Capricorn applies a high pass filter to remove the DC component and low-frequency noise. The cutoff frequency of the high pass filter is empirically set to 20Hz.

The vibrations from different objects have different patterns, which are often manifested as various frequency peaks in the spectrum. As the second step, Capricorn performs the Fast Fourier Transform (FFT) algorithm on the filtered time series to extract frequency domain features. The features extracted by the FFT algorithm are often too high-dimensional for any simple classifiers since the length of FFT output is the same as the raw signal. We further reduce the feature dimension by linearly grouping the frequencies into b linear bins (i.e., $b = 32$ in our prototype), and use the maximum magnitudes in each bin as the final feature value. This techniques is similar to the *MaxPooling* layer in a neural network.

For the discrete state estimation, we train a group of lightweight state classifiers $\mathcal{M}$, one for each object type. Each model is trained using the features described in the previous paragraph. An alternate choice would be training a unified model for the union of all the objects. However, a unified classification model deals with a more complicated decision boundary (i.e., whose complexity will grow with the types of the object Capricorn can support). Thus, the unified model has to be sufficiently more complicated to achieve the same performance as the individual models, which requires larger amounts of data and longer inference time. Since we already know the object type information from the extrinsic sensing pipeline, we utilize that information to choose a certain model $\mathcal{M}_j$ for a specific object type j from a group of lightweight simple models $\mathcal{M}$. Algorithm 1 shows the psuedocode of the discrete state estimation. Our solution requires some self-collected data to train these simple classifiers for each object type, and further details can be found in section 8.2.3. Thanks to the context information provided by sensor fusion, our design is sufficiently modular so that in order to support more object types, we just need a small amount of new data to train a simple new model, rather than require large amounts of data to retrain the whole classifier.

Algorithm 1: Discrete object state estimation.

Input: Extracted vibration signal $\tilde{V}(t)$ and object type o_j
Output: Discrete intrinsic object state $s^j_{i,d}$

$\mathbf{V}(t) \leftarrow \text{HPF}(\tilde{\mathbf{V}}(t))$; /* apply the high pass filter */
$Q[f] \leftarrow FFT(\mathbf{V}(t))$; /* apply FFT algorithm */
$f_l \leftarrow C$; /* feature size for classification */
$p_s \leftarrow \frac{\text{size}(Q[f])}{f_l}$
$F \leftarrow \text{MaxPool1D}(Q[f]), p_s, \text{stride} = p_s)$; /* applying MaxPool1D(...) */
$\mathcal{M}_j \leftarrow \text{select\_SVM}(S_e)$; /* select a proper SVM model */
$s^j_{i,d} \leftarrow \mathcal{M}_j.\text{predict}(F)$
return $s^j_{i,d}$

Algorithm 2: Continuous object state estimation.

Input: Extracted vibration signal $\mathbf{V}(t)$, object type $o_j = person$
Output: Continuous intrinsic object state $s^j_{i,c}$

$IMF(t) \leftarrow \text{VMD}(\mathbf{V}(t))$; /* calculate the VMD */
$U_0(f), F(f) \leftarrow \text{FFT}(\text{IMF}_0)$; /* calculate the FFT */
$l \leftarrow \arg_{max_f}[F(f) > \zeta_l]$; /* low cut-off frequency */
$h \leftarrow \arg_{min_f}[F(f) < \zeta_h]$; /* high cut-off frequency */
$f_c \leftarrow U_0\big[\arg_{max_f}\{U_0(f), f \in [l,h]\}\big]$; /* Frequency Peak Search */
$s^j_{i,c} \leftarrow f_c \times t$; /* Convert to breath rate */
return $s^j_{i,c}$

5.3.2 ***Continuous State Estimation.*** When the sensing target is a living being, it is more meaningful to estimate the intrinsic states in a continuous manner. Previous works have shown the possibility of continuously estimating and tracking the vital signals of a human (i.e., respiration rate and heart rate) from vibrations [63, 83]. In this case, Capricorn estimates the continuous state, which requires the simultaneous extraction of vital signs from multiple living beings.

For the continuous state estimation, a longer buffer is necessary to make meaningful inferences. Capricorn *first* concatenates the sliced vibration signal $\tilde{V}(t)$'s in a longer buffer (e.g., a duration of 30 seconds). For vital signal extraction, model decomposition methods have been proven to be effective [40, 45, 83]. Thus, *secondly*, Capricorn applies the VMD algorithm [17] to the extracted vibration signal $\tilde{V}(t)$ [30]. The vibration signal from a living being is a combination of respiration, heartbeat, body movement, and environmental noise [83]. The VMD algorithm decomposes the input signal $\tilde{V}(t)$ into a number of band-limited sub-signals $\mathbf{u}_k$ (i.e., also known as intrinsic mode functions or IMFs) by solving the following optimization problem [17]:

$$\min_{\mathbf{u}_k,\omega_k} \left\{ \sum_{k=0}^{K-1} \left\| \partial_t \left[\left(\delta(t) + \frac{j}{\pi t} \right) * \mathbf{u}_k(t) \right] e^{-j\omega_k t} \right\|_2^2 \right\} \text{ s.t. } \sum_{k=0}^{K-1} \mathbf{u}_k = \tilde{V}(t). \tag{10}$$

where the vibration signal $\tilde{V}$ is decomposed into K different IMFs u_k with center frequencies ω_k. The IMF $\mathbf{u}_0$ with the lowest center frequency gives us an estimation of the respiration rate. *Third*, Capricorn performs the spectrum analysis on $\mathbf{u}_0$ to generate its spectrum $\mathbf{U}_0(f)$. The typical respiratory rate for a resting healthy adult is 12–16 breaths-per-minute [4]. Thus, for the respiration estimation, we focus only the frequency components between 0.13-2Hz. Capricorn repeats the continuous estimation process for all living objects in the scene, updating the continuous state using the stored vibration data at regular intervals. Algorithm 2 shows the continuous object state estimation process.

6 INFORMATION STORAGE UNIT

The information storage unit creates an in-memory table and further uses that table to provide the rich scene analysis information via operations including "create", "query", "update", and "delete".

There are eight different entries in the in-memory table: primary ID, time stamp, (object) type, distance, bounding box, vibration (signal), (intrinsic) state, and persistence. Each row in the table is used to keep track of these eight entries of an object as follows:

1. The *primary ID* initially is generated by the object tracker (in the extrinsic sensing pipeline) and it is fixed as long as the object is present in the scene.
2. The *time stamp* represents the most recent time when object states are updated.
3. The *object type* represents the detected object type from the extrinsic sensing pipeline.
4. The *distance* represents the current distance estimation of the object.
5. The *bounding box* represents the current bounding box coordinates of the object from the detection algorithm.
6. The *vibration* stands for the time series data from the RF sensor. It is stored as a data stream (i.e., bytes).
7. The *state* represents the current intrinsic state of object.
8. The *persistent* represents whether the object is currently present or not. As we mentioned in Section 4, the object detection algorithm may fail to detect and identify the bounding box of an object in the scene for a few consecutive frames. To handle this issue, Capricorn persists the object state in the in-memory table for the next T_{max} frames even it is not present in the scene.

7 MULTI-VIEW CAPRICORN

In this section, we discuss how Capricorn can be up-scaled to a sensor network covering the scene from different perspectives. In the intrinsic sensing pipeline (Section 5), the UWB radar separates the vibration of different objects based on their distances. The system's performance can be negatively impacted by multiple nearby objects. Previous research show that UWB radars, when separating vibrations from two objects, assume the targets are placed more than 25 cm apart from the sensor's perspective [65]. Following is a motivational example: since the distance measurement of the UWB radar is one-dimensional, if two objects sit at the same distance d_0 to the sensor, their vibration signals can contaminate each other (see Figure 5 (Left)). There are majorly two possible ways to partially address this problem. One can instrument the sensor system with mobility by placing it on a ground robot. Another approach is to have a network of sensors that views the scene from different perspectives. In Capricorn, we pick the second approach as the proposed architecture can be easily up-scaled to a network of multiple sensor nodes covering a wider range of views.

We add another sensor view as is shown in Figure 5 (Left). The newly added sensor set is controlled by another host machine in the same local area network. The data collection and processing units in Capricorn employ a publish-subscribe mechanism. Therefore, we

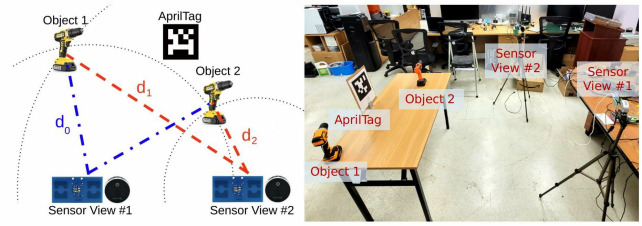

Figure 5: (Left) Multi-view version of Capricorn to distinguish the objects at the same distance. (Right) Real-world setup of the multi-view Capricorn.

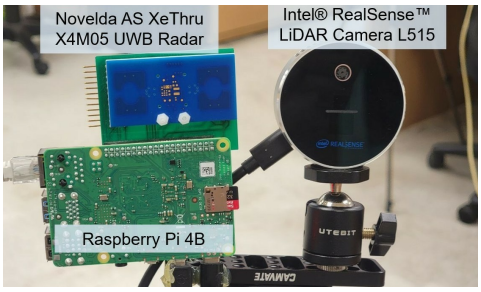

Figure 6: Hardware platform of Capricorn prototype.

only need to add a set of new sensor data topics and new processing functions subscribing to these new topics. For the extrinsic sensing, we leverage an AprilTag [48] to coordinate multiple sensor nodes. When a LiDAR and camera-based system observes an AprilTag in the scene, it can automatically calculate the camera extrinsic matrix $\mathbf{K}_c^{ex}$ [48]. With the help of $\mathbf{K}_c^{ex}$, we transfer the coordinates of each sensor node and object to a global coordinate system defined by the AprilTag [48]. Then, we can compute the distance between each object and each sensor. In the multi-view Capricorn, we use the sensor view #1 as the primary view for visualization, and its UWB radar #1 is also the primary RF data source. In the example shown in Figure 5 (Left), Capricorn found that the distances between sensor view #1 and the two objects are similar (within a threshold). In this case, we switch to the UWB radar in sensor view #2, where the two objects are separable because the two objects have different distances d_1 and d_2 from view #2.

In the real-world deployment (as shown in Figure 5 (Right)), Capricorn firstly calculates the sensor-target distances and picks a viewpoint where the objects are separable. The UWB sensor on the selected viewpoint then becomes the primary source of the RF data. Finally, Capricorn takes slices from the RF data matrix to separate the vibrations time series for each object.

The pub-sub data handling mechanism in Capricorn ensures its scalability. The above approaches can be applied for adding one or more sensors to the system regardless of the sensor types. In the current implementation, most of the computation happens on the sensor node #1, which also runs the ROS broker. However, it is also possible to compute in a distributed manner to reduce the data communication overhead by streaming sensor inference results.

8 IMPLEMENTATION

8.1 Hardware

We implemented a multimodal sensor module of Capricorn with an Intel RealSense LiDAR Camera L515 and a Novelda AS Xethru X4M05 Radar sensor. The L515 LiDAR Camera provides both the RGB images and the depth map. It can be substituted by another stereo vision camera or separate vision and depth sensors. We mounted the sensors onto a tripod using a cheeseplate, as shown in Figure 6. The X4M05 Radar sensor is connected to a Raspberry Pi 4B using Serial Peripheral Interface (SPI). The main system runs on an Intel NUC mini PC consists of an Intel i7-6770HQ processor. The LiDAR Camera is connected to the main system using a USB cable, and the main system handles the communication with the LiDAR Camera via Intel RealSense SDK 2.0. No special hardware accelerator is used in the entire system.

8.2 Software

We implemented the entire architecture in C++ using ROS [57]. We mainly utilized the Pub-Sub mechanism in ROS for sensor data collection. Specifically, we implemented two separate ROS nodes to collect the data from LiDAR camera and UWB Radar, respectively. We implemented another ROS node for the sensor fusion and information storage units of Capricorn. All the nodes are run on the Intel NUC mini PC except for the driver node responsible for the UWB Radar data collection that runs on the Raspberry Pi.

8.2.1 ***Numerous Parameters.*** In Section 2, we introduced a number of parameters. We summarize the values of these parameters used in the prototype as follows. Some of the parameter values are determined by hardware specifications while others are empirically tuned for optimal performance.

1. In the object tracking module, we set T_{max} to 10 to compensate for the bounding box jitters of YOLOv5.
2. The chunk size of the UWB radar data is set to 1024 frames, and the frame rate of the UWB data is 1000 Hz.
3. In the phase noise correlation, we have $d_0 = 0.3$ and $g_0 = 0.0514$.
4. In continuous intrinsic state estimations, $\zeta_l = 0.13$ Hz, $\zeta_h = 2$ Hz.

8.2.2 ***Object Detection and Classification Model.*** There are multiple variants of YOLOs with different model sizes. We used YOLOv5s as the object detection and classification model for a balance between fast inference speed and prediction stability. We fine-tuned a YOLOv5s model on a self-collected dataset since the application scenarios (to be discussed in Section 9) include several objects which are not covered by the pre-trained YOLOv5s model. *Model training.* The training dataset consists of five classes: people, vacuum cleaners, washing machines, table fans, and drills. The images of people are obtained from the Common Objects in Context (COCO) dataset, while the pictures of the household objects are a combination of images captured on a smartphone camera and pictures found online. The online household object pictures are often obtained from shopping websites, which depicted the objects with a blank background. However, this results in the model struggling in scenarios with a more complex background. To combat this, we use a photo editor to remove the background and isolate the object. The processed image was then placed in randomly chosen backgrounds of home spaces, enabling the model to recognize images in the smart home context. Roboflow is used to aggregate the images, and its ability to augment a dataset by applying transformations (i.e., shear, rotate, and brightness adjustment) allowed the dataset to grow exponentially, eventually producing a set of 20000 images.

The composition of the twenty thousand images are as follows: 29.5% washing machines, 21.8% fans, 16.9% vacuums, 21.1% people,

and 10.7% drills. We put more training data in the "washing machine" category to correct the model's tendency of failing to classify washing machines. After training, the mAP@.5 values were 0.479 for person, and over 0.99 on the rest of the objects.

8.2.3 ***Discrete Object State Estimation Model.*** As mentioned earlier, we use a group of SVM models $\mathcal{S}$ as the discrete object state estimation classifier. We collected a UWB dataset covering all the discrete states of each object, during which we randomly changed the object's location and orientation to improve the diversity of the dataset. The UWB dataset is sampled at 1kHz. Table 1 presents the summary of collected UWB dataset (number indicates minutes collected). We collected more data for 'Dry' state of the washing machine and 'Speed 1' and 'Speed 2' states of the table fan, because these states present partially similar features to their neighbouring states. More data help the model to build a better decision boundary.

	Drill	Vaccum	Washing Machine		Fan Speed		
Idle	On	Sweeping	Wash	Dry	1	2	3
14.3	10.3	8	15.3	28.3	19	19.3	4.3

Table 1: A summary of the collected UWB dataset for discrete object states. Number indicates minutes collected.

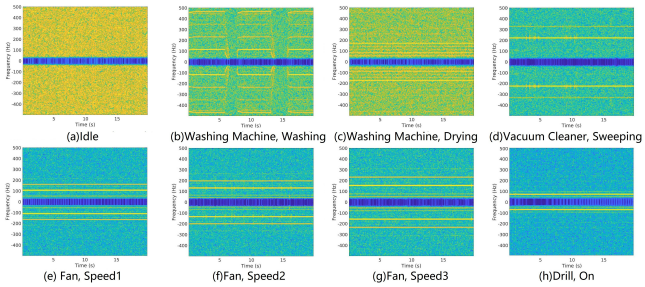

Figure 7: UWB data spectrogram for discrete object states.

We also present the post-processed data spectrogram (i.e., change of frequency over time) of all the discrete object states in Figure 7. As we see in the figure, the returned signal carries the vibration frequency characteristic of the appliances, and there are obvious signal patterns corresponding to different states of the same object. These data characteristics make it feasible to build a classifier to recognize the discrete operating states of these appliances using the FFT spectrum as the feature of the classifier.

9 EVALUATIONS

Next, we evaluated our prototype with extensive experiments, showing that Capricorn works well in many real-life settings in a real-time and scalable manner. This video (https://youtu.be/b-5nav3Fi78) covers the evaluations in this section.

9.1 Workshop Machine Operation Monitoring

In industrial assembly lines or workshops, it is important to monitor the machine's operating states to promote safety and productivity. A workshop setting is shown in Figure 8(a), where we placed four drills in front of Capricorn as a surrogate of machines on an assembly line. The aim is to detect all the drills' operating states. Here we use $s_i = 1/0$ to represent their "on/off" states, where $i = 0, 1, 2, 3$ corresponds to the four drills from left to right. Figure 8(b) is a screenshot of our real-time system. The four drills' states are "on off on

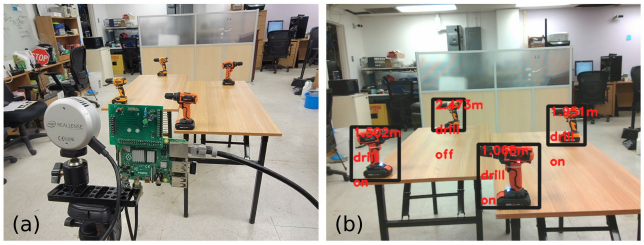

Figure 8: A Workshop scene: (a) Four drills are placed in front of our Capricorn prototype (b) An example output of our system.

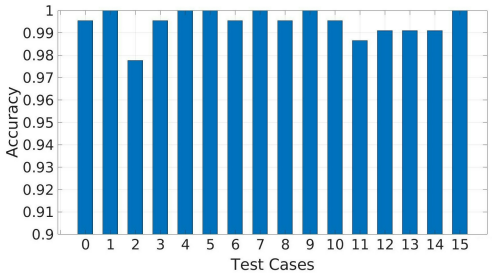

Figure 9: Accuracies of all the sixteen possible states $s_0s_1s_2s_3$ from 0000 (0) to 1111 (15) of the four drills.

on", therefore $s_0s_1s_2s_3$ is 1011. This scene is challenging for purely audiovisual-based systems (e.g., human perception or a combination of a camera and a microphone array), since (1) there is limited visual clues about whether a drill is operating or not, and (2) the scene is noisy with multiple drills operating and it is difficult to achieve a clean audio separation. From the authors' experiences, the audio of two objects cannot be clearly separated without noticeable residuals if they are placed less than 20 degrees apart in terms of angles, using ReSpeaker 2.0 [53] and the state-of-the-art Geometric High-order Dicorrelation-based Source Separation (GHDSS) algorithm [15]. The scene in Figure 8(a) contains four objects with a FOV of 70 degrees, which is challenging for small-scale microphone arrays.

To evaluate the effectiveness of this system, we enumerated all possible $s_0s_1s_2s_3$ cases from 0000 (0) to 1111 (15). For each case, we ran the system for one minute and calculated their accuracies based on the outputs of every 1.024 second (the state estimation is updated every time a new chunk of UWB data arrives). The results in Figure 9 show that all cases can reach more than 97% accuracy, and the average accuracy is 99.47%.

9.2 Home Appliance Usage Tracking

Household robots for home monitoring like Amazon Astro have been emerging in recent years. These household robots can benefit from Capricorn's technologies. We considered the second scene at a smart home (see Figure 10(a)), where the homeowner wants to know the usages of multiple household appliances. This application can provide insights into household appliance usage habits and associated energy consumption. We placed a washing machine, vacuum cleaner, and table fan in the scene and controlled their operating states separately.

A visualization of Capricorn's output is shown in Figure 10, where the object types, object bounding boxes, distances, and estimated internal states are overlaid on top of each object. On the left examplar scene, Capricorn detected that the washing machine

is in the washing mode, the vacuum cleaner is sweeping, and the fan is spinning at speed 2. On the right-hand side, Capricorn found the washing machine in drying mode and the fan at speed 3. These object internal states won't be possible from only video scene analysis. By applying rich scene analysis technologies, we can make these household appliances "smart" without instrumenting them with any electronics.

9.3 Latency Analysis

Capricorn is a real-time system, so it is important to analyze the processing time of its individual building blocks and its end-to-end latency. The numbers in Table 2 are measured using ten random data frames, and we report the mean and standard deviation. For clarity, we ignore those components whose execution requires less than 1 ms.

Capricorn Component	Mean(ms)	Std(ms)
Camera/Depth Pub-Sub Delay	1.08	0.13
YOLOv5	38.35	5.25
YOLOv5 (GPU)	6.28	1.29
Whole Extrinsic Sensing Pipeline	42.81	6.3
UWB Chunk Pub-Sub Delay	171.61	21.87

Table 2: Latency analysis of Capricorn in the appliance usage classification scene.

We refer readers to Figure 2 where we explain Capricorn's architecture for a better understanding of the discussions in this section. First, we discuss the required time of the data collection unit. This unit streams the sensory data in a Pub-Sub mechanism. Recall that the camera and depth sensors are connected to the Intel NUC using a USB cable. Therefore, the data streaming latency is insignificant (1.08 ± 0.13 ms). On the other hand, the UWB radar is hosted on a Raspberry Pi, which buffers and streams a UWB data matrix $\mathbf{M}$ over the network. Therefore, the required time of streaming UWB sensor data cannot be ignored. We connected the Intel NUC and the Raspberry Pi to a Ethernet switch and set up a pair of NTP server and client between these two devices to synchronize their local clock. In this way, we were able to measure 172.61 ± 21.87 ms to stream a 1024 ms chunk of UWB data.

Next, we look into the required time of the sensor fusion unit. In the sensor fusion unit, the extrinsic and intrinsic sensing pipelines are running in parallel threads. Recall that the extrinsic sensing pipeline in sensor fusion unit is triggered by the arrival of camera and depth frames. From Table 2, we can see that the latency of the whole extrinsic sensing pipeline is 42.81 ± 6.30 ms, and almost all the computation time is spent on the YOLOv5 inference (38.35 ± 5.25 ms). The rest component of the pipeline (i.e., object tracking and depth estimation algorithms) all consumes less than 1 ms. This amount

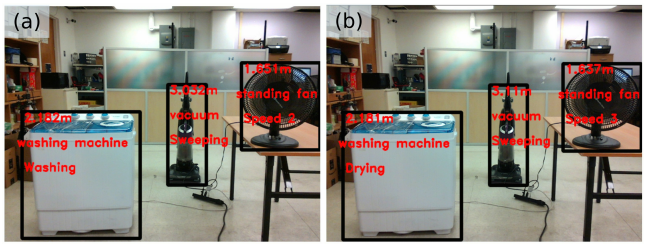

Figure 10: Capricorn's exemplar output in a smart home.

of inference latency is reasonable since we are using an Intel NUC mini PC without any hardware accelerator. We also measured the YOLOv5 latency on a PC with Nvidia Titan X GPU (see Table 2 Line 3) and its latency can be reduced to 6.28 ± 1.29 ms.

We then analyse the latency of the intrinsic sensing pipeline running in parallel. This pipeline is trigger by the arrival of a UWB chunk. The first two steps are vibration extraction and phase noise correction, which take only 0.73 ± 0.03 ms because the prior object depth information from extrinsic sensing pipeline reduces the complexity of the vibration extraction step. In this application scenario, the final step is to select the most suitable SVM model for the intrinsic state estimation based on the object type. In this step, we create a new thread for each object presented in the scene to handle multiple objects simultaneously. Therefore, the overall latency of the discrete state estimation mostly depends on the inference time of the slowest SVM model. The longest inference time of the SVM model (for the washing machine states) requires 0.53 ± 0.10 ms.

The information storage unit is current implemented as a class in the memory shared by all the threads, and its I/O delay is trivial. From the above discussions, we can see that Capricorn generates the simultaneous estimation of extrinsic and intrinsic object states within 200 ms when the objects possess discrete states.

9.4 Complex Event Modeling

Understanding and modeling human behavior has been a hot topic in the sensing community. Recent research has expanded from classifying simple activities to understanding complicated sequences of events. This section demonstrates how Capricorn uplifts complex event detection research. A complex event detection system must first understand multiple simple "atomic" events. Then, the system makes logical reasonings to model a complicated event that spans space and time [71]. Through the rich scene analysis, Capricorn provides a richer set of atomic events and simplifies the design of these systems.

Using Capricorn, we provide a simple example as is shown jointly in Figure 11 and Figure 12. We show how we model the behavior of "a person doing laundry" as a finite state machine in Figure 12.

Figure 11 shows the screenshots of the complex event detection system. We have the current state (defined in Figure 12) displayed on top of each subfigure. The system first started with the "Idle" state (Figure 11(a)). In (b), the extrinsic sensing pipeline detected the human-machine interaction based on their spatial proximity. Then, in (c), the state transition fired, and we entered the "Washing" state as defined in Figure 12. The state transition happens because Capricorn detected that the machine's vibration mode changed from idle to washing. This transition demonstrates the unique rich scene analysis capability of Capricorn: with vision sensors alone, we can only capture the interaction between the user and the machine where the machine states are invisible. On the other hand, by the RF-vision fusion, the proposed system makes inferences about the operating states of the washing machine and provide a wider range of possible atomic events. Nextly, in (d), the system moved to the "Wash Done" state because Capricorn detected that the vibration from washing had stopped, and the machine became idle. Here, the system sent an alarm to remind the user to collect the clothes. The system removed the alarm when another human-machine interaction was detected in (e). Similarly, the system went

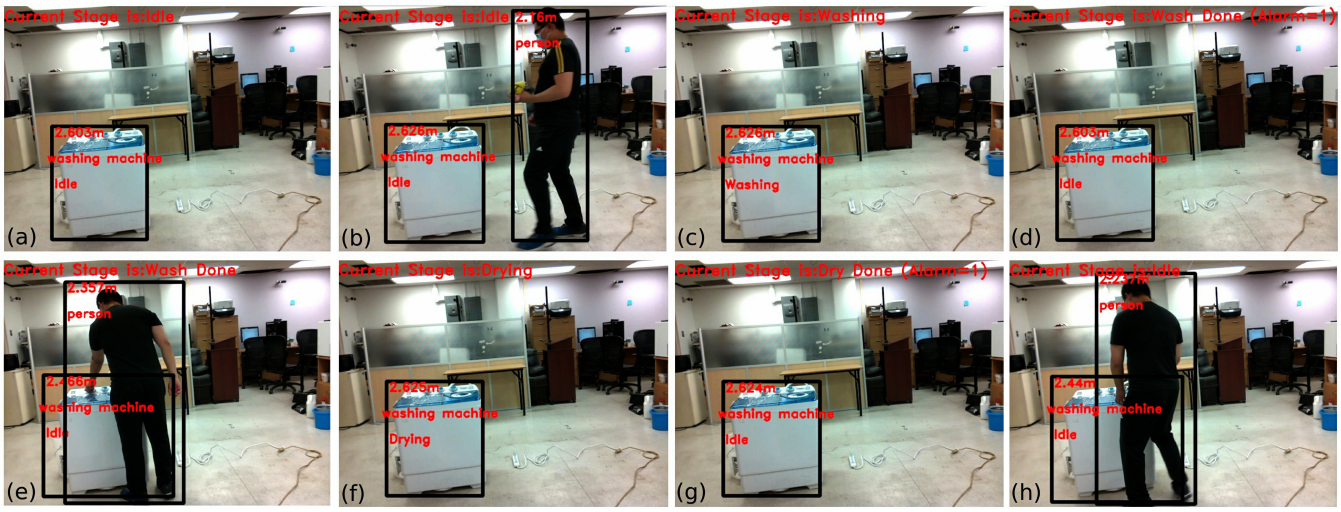

Figure 11: Screenshot of Capricorn performing complex event detection.

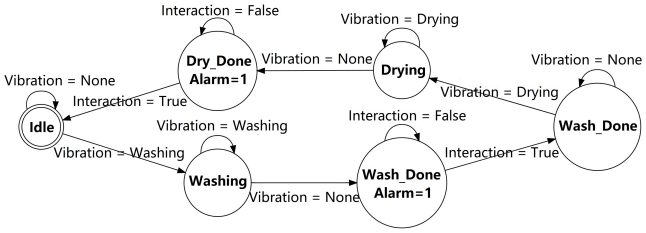

Figure 12: A finite state machine modeling a person doing laundry.

through the state of "Drying"(e) and "Dry Done"(f) based on Capricorn's estimation of the washing machine's intrinsic state. Finally, when Capricorn detected that the user had collected the laundry, it finished a laundry cycle and moved back to idle.

9.5 Multi-view Capricorn

In Section 7, we have proposed scaling up Capricorn by adding another set of sensors and introduced a setting in Figure 5. Figure 13(a) shows a failure case of the single-view Capricorn under that setting: the two drills were both placed at around 1.62m from the sensor, making them fall into the same distance bin. Although only one of the drills is 'on', the system recognized both of them as 'on' because the UWB radar could not separate these two objects based on their distances. With the multi-view scaling-up, Capricorn separated these two drills from UWB 2, as shown in Figure 13(b).

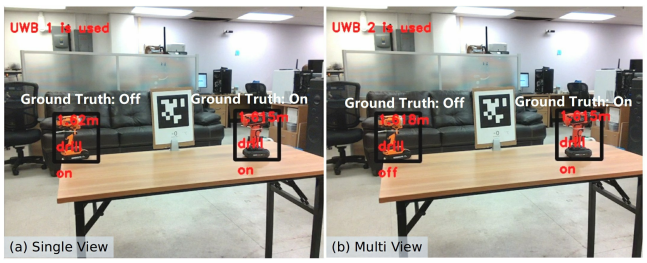

Figure 13: A Multi-view implementation of Capricorn: (a) Single view: two drills are at the same distance from UWB 1's viewpoint, so the left drill is misidentified as 'on' state; (b) Multi-view: when UWB 2 is used, the two drills are separated and recognized correctly.

When the multi-view Capricorn detected that the two objects are sitting at the same distance, it automatically switched the RF data source to sensor view #2 (note that the top-left corner of Figure 13(b) shows "UWB2 is being used"). From the perspective of sensor view #2, the two objects were separable, and the left drill was correctly recognized as 'off' as shown in the figure.

9.6 Multi-person Respiration Estimation

Capricorn's intrinsic sensing pipeline captures not only high-frequency motions such as machine vibrations but also low-frequency movements such as human vital signals. In the last application scenario, we employed Capricorn for the multi-person respiration estimation to evaluate the performance of its continuous intrinsic state estimation algorithms. Here, we simulated a medical triage scene where one needs to rapidly assess the medical condition of people, particularly whether they are alive, so that medical care should be focused on the survivors in a timely manner. As shown in Figure 14(a), we had a mix of multiple persons and inflatable dummies (as proxies for dead bodies). Currently, one has to search for survivors by checking for the presence of vital signs one by one, since technologies to do so from a distance get confused when there are multiple candidates. The existing video scene analysis systems can detect the persons on the scene but fail to distinguish between dead bodies and survivors. However, as the figure suggests, Capricorn accurately identifies and classifies dead bodies and survivors in the same scene. Capricorn captures and separates multiple vibrations at different distances, which allows us to recover the respiration waveforms from living objects. As shown in Figure 14(c), Capricorn robustly recovers the respiration waveforms of multiple human subjects (person 1 and 3).

9.6.1 ***Estimation Accuracy.*** Capricorn calls the continuous intrinsic state estimation when an object is classified as a living being, which is a learning-free estimator based on VMD algorithm. We collected a human respiration dataset from five of the authors. In each session, the volunteer sat in from of the Capricorn sensors for one minute. Each volunteer repeated the data collection session for five times, and we used the self-reported breath counts as the baseline.

We can see that the estimation error is less than 0.2 bpm in about 50% cases and it is less than 1.2 bpm in 80% cases. The mean

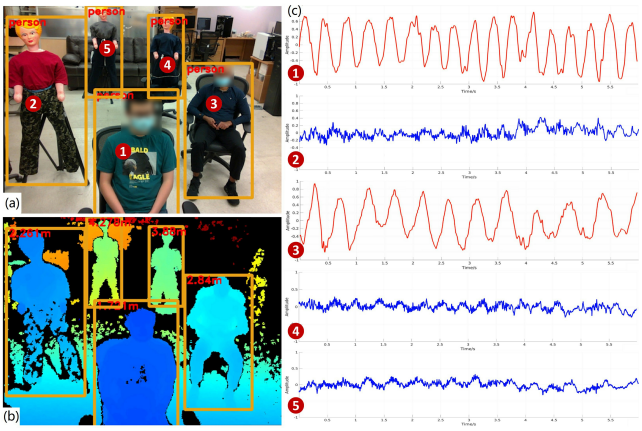

Figure 14: Multi-person respiration rate estimation. (a) Object detection (b) Distance estimation (c) Respiration waveforms recovered from each "person".

estimation error is 1.0586 bpm, the median error is 0.9603 bpm, and the standard deviation is 1.3424 bpm. In one of the sessions, we had one author wearing a respiration monitoring belt (NUL-236 from Neulog) to provide a baseline respiration waveform. In Figure 15, we compared the respiration waveform from Capricorn (orange) with the ground truth (blue). From the waveform, we can say that Capricorn can obtain human respiration waveforms with a reasonable quality.

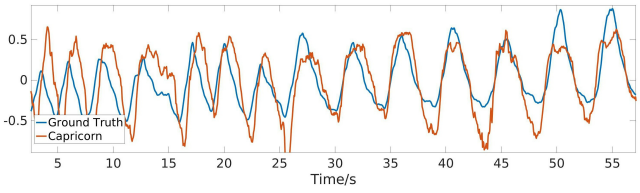

Figure 15: The respiration waveforms recovered from Capricorn compared with the ground truth.

9.6.2 ***Latency Analysis.*** Note that the code for multi-person respiration monitoring is the same as the one we used before for appliance usage detection. The design of Capricorn allows the system to automatically choose the most appropriate signal processing pipeline based on the environmental context. Therefore, the time latency of data streaming, extrinsic state estimation, and UWB data processing remains the same. The only difference is the latency of the continuous state estimation. For this estimator, the running time is 1742.56±215.38ms for a buffer containing 30 seconds of UWB data. Here, the most time-consuming calculation is the VMD algorithm to decouple the respiration vibration from other noises. Again, we created different threads for each person presented in the scene to increase parallelism. The current buffer size is set as 30seconds and is updated on a rolling basis, so the bpm estimation will be updated around every 2 seconds. It is also possible to shrink this interval by using a shorter buffer. At the beginning stage, Capricorn does not start the respiration rate estimation until there are 15 seconds of data in the buffer to avoid generating random bpm results.

10 RELATED WORK

Wireless Vibrometry. Researchers have devoted efforts to measuring vibration remotely using active sensing. A majority of these works employed radio-frequency (RF) sensing modalities to measure the amplitude and frequency of a vibration, such as RFID [35], WiFi [49, 67], IR-UWB Radar [65], and mmWave [27, 72]. Some other work sensed vibrations remotely using a laser [81, 82]. A number of applications are enabled by these explorations, such as monitoring the spin of centrifugal machines [74], managing the usage of smart home appliances [81], and even sensing human speech for user authentication [33]. Wireless Vibrometry has also been employed to sense low-frequency phenomenons such as human vital signals, for example, the subtle chest movements caused by human respiration and heartbeat, even from multiple persons [25, 75–77], or under body movements [12, 84, 85] However, these works suffer from two major limitations. First, many of these systems are not real-time and only work offline. Second, it is difficult to associate the detected vibration with real-world objects because these systems cannot visually "see" the world and they have no knowledge of the extrinsic object states.

Multimodal Sensor Fusion. Application of multimodal sensor fusion involves gait abnormality detection [54], activity recognition [73], 3D imaging [56], security monitoring for intelligent buildings [38], localization [29], and vehicle navigation [9]. A substantial category of sensor fusion is *decision-level fusion*. In this scheme, the fusion happens after the classification [59]. Commonly seen techniques include but are not limited to majority voting [66], score weighing [18], and ranking [41]. In recent years, with the fast development of deep learning, multimodal sensor fusion has started utilizing neural networks. Some of them even applied an end-to-end structure to make inferences about the environment. This type of fusion is also known as *data-level fusion*. For example, [6] proposed a neural architecture to process LiDAR, camera, and radar data, and the system can reliably perform objection detection in adversarial weather. Similar works include [10, 24, 29, 34, 44, 46, 64], where they use an end-to-end neural structure to combine RF and EO (Electro-optical) sensors for semantic segmentation or object detection.

Another class of sensor fusion methods is known as feature-level fusion [59]. In feature-level fusion, algorithms generate intermediate "features" (inferences) from the raw signal, and use these features to improve the task performance. In [19], the authors perform robust human activity recognition combining time-domain features extracted from wearable inertial sensors and histograms of oriented gradients extracted from a RGBD camera. Xin et al. extracted Fisher feature vectors from images, fingerprints, and finger veins, fusing them for human identification [70].

11 LIMITATIONS AND FUTURE DIRECTIONS

1. Accurate Perception Models in Real Deployments. The main contributions of this paper lie in the conceptual design and platform-independent algorithms for real-time rich scene analysis. While our paper makes use of several machine learning models, we do not claim them as our technical contribution and hence, our main efforts went towards designing our real-time framework with platform-independent algorithms and not towards the optimization of these machine learning models. Currently, the experiments are conducted mostly in controlled in-lab configurations. It would be

interesting to research how the system will perform in real production environments on tasks such as machine state monitoring. Also, the current object state estimators are simple classifiers to distinguish a few discrete states of the object. We are working to leverage the combination of signal processing and sequence-to-sequence deep learning models to reconstruct fine-grained waveforms of objects' vibrations. There are more open research questions to be solved. For example, how prior knowledge (e.g., physics) can be exploited to increase efficiency, reduce the need for large training data, and minimize uncertainty simultaneously.

2.Sensor Mobilities and Close-by Objects As discussed in Sec. 7, Capricorn faces challenges when two objects of interest sit at similar distances. We propose using a network of sensor infrastructures and viewing the scene from different angles to mitigate the issue. But since this is a fundamental issue in ToF-based sensing methods, we cannot fully eliminate this effect if two objects are physically placed too close together. Apart from using a sensor network, another possible solution is to employ sensors with mobilities. For example, Capricorn currently places its sensors on a fixed tripod. If the sensors can be migrated to a moving robot or held in hand, it will be much easier to obtain viewpoints where objects are more separable. However, this is not a trivial problem as the movements of the sensor platform mask out the target vibrations and distort the signal significantly. There are already some pioneering works looking into this issue [36], and we are also working towards enabling RF sensing platforms with mobilities.

3. Integrating More Sensing Modalities. Currently, our system fuses LiDAR, camera, and RF sensors only, which is a prototype to demonstrate the novel idea of *rich scene analysis*. The current system does not have sensing capabilities such as audio or thermal, which could also be very informative. For example, microphone arrays can also be useful for intrinsic state estimation if the target phenomenons make a noticeable sound and are angularly separated. This capability is complementary to the wireless vibrometry technologies we employed that work better to separate vibration (maybe inaudible) from similar directions but at different distances. In the future, we expect to build a large sensor network consisting of multiple nodes, each possessing several sensing capabilities (e.g., LiDAR, thermal camera, mmWave radar, microphone arrays). Also, on the UWB radar front, recent research has introduced a MIMO platform with antenna arrays and beamforming ability which can be adopted to improve the robustness of isolating objects' RF signal [13]. Related research questions include the optimal scheduling of data, computations, and neural architecture to fuse similar sensing modalities. To fully unleash the potential of multimodal sensors, it might be promising to investigate neural architectures where there is a common representation for different sensor modalities (as opposed to making independent predictions from different modalities and then fusing those predictions together).

12 CONCLUSIONS

In this paper, we presented a novel concept of *rich scene analysis* where the proposed RF-vision sensor fusion system simultaneously captures the intrinsic and extrinsic object states in real-time. The proposed system demonstrates that the information acquired from the vision sensor helps us to make more sense of the RF data and improve the versatility of the RF system in dynamic environments (e.g, when the sensing object types and their numbers are indefinite). Correspondingly, the RF sensors complements the vision sensors by making inferences about the objects' intrinsic states that are invisible to vision sensors. One limitation of Capricorn is that it relies a lot on the robustness of its individual machine learning components (for object detection and signal classification) to correctly perceive the scene. With self-collected datasets, we were able to train these models enough to demonstrate our core ideas. However, these models might suffer a performance drop if deployed in unseen environments due to the distribution mismatch between the training and real-world data. Thus, an interesting future direction will be developing robust machine learning technologies to offset the distribution shift of the data, especially when large labeled training datasets are unavailable.

ACKNOWLEDGMENTS

The authors would like to thank the shepherd and the reviewers for their comments that helped to improve this work. The research reported in this paper was sponsored in part by: the CONIX Research Center, one of six centers in JUMP, a Semiconductor Research Corporation (SRC) program sponsored by DARPA; by the IoBT REIGN Collaborative Research Alliance funded by the Army Research Laboratory (ARL) under Cooperative Agreement W911NF-17-2-0196; by the NIH mHealth Center for Discovery, Optimization and Translation of Temporally- Precise Interventions (mDOT) under award 1P41EB028242; and, by the National Science Foundation (NSF) under award #CNS-1705135. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the ARL, DARPA, NIH, NSF, SRC, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Qaisar Abbas, Mostafa EA Ibrahim, and M Arfan Jaffar. 2018. Video scene analysis: an overview and challenges on deep learning algorithms. *Multimedia Tools and Applications* 77, 16 (2018), 20415–20453.

[2] Adeel Ahmad, June Chul Roh, Dan Wang, and Aish Dubey. 2018. Vital signs monitoring of multiple people using a FMCW millimeter-wave sensor. In *2018 IEEE Radar Conference (RadarConf18)*. IEEE, 1450–1455.

[3] Novelda AS. 2020. XeThru X4 Phase Noise Correction. https://github.com/novelda/Legacy-Documentation/blob/master/Application-Notes/XTAN-14_XeThru_X4_Phase_Noise_Correction_rev_a.pdf. Accessed: 2020-05-28.

[4] Kim E Barrett, Scott Boitano, Susan M Barman, and Heddwen L Brooks. 2010. Ganong's review of medical physiology twenty. (2010).

[5] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. 2016. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*. IEEE, 3464–3468.

[6] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11682–11692.

[7] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. 2020. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934* (2020).

[8] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. 2017. Soft-NMS–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*. 5561–5569.

[9] Peide Cai, Sukai Wang, Yuxiang Sun, and Ming Liu. 2020. Probabilistic end-to-end vehicle navigation in complex dynamic environments with multimodal sensor

fusion. *IEEE Robotics and Automation Letters* 5, 3 (2020), 4218–4224.
[10] Simon Chadwick, Will Maddern, and Paul Newman. 2019. Distant vehicle detection using radar and vision. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 8311–8317.
[11] Hua-I Chang, Chieh Chien, James Y Xu, and Greg J Pottie. 2013. Context-guided universal hybrid decision tree for activity classification. In *2013 IEEE International Conference on Body Sensor Networks*. IEEE, 1–6.
[12] Zhe Chen, Tianyue Zheng, Chao Cai, and Jun Luo. 2021. MoVi-Fi: motion-robust vital signs waveform recovery via deep interpreted RF sensing. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 392–405.
[13] Zhe Chen, Tianyue Zheng, and Jun Luo. 2021. Octopus: a practical and versatile wideband MIMO sensing platform. In *Proceedings of the 27th Annual International Conference on Mobile Computing and Networking*. 601–614.
[14] Abe Davis, Michael Rubinstein, Neal Wadhwa, Gautham J Mysore, Fredo Durand, and William T Freeman. 2014. The visual microphone: Passive recovery of sound from video. (2014).
[15] HARK development team. 2022. HARK Document Version 3.3.0. (Revision: 9509) : GHDSS. https://www.hark.jp/document/hark-document-en/subsec-GHDSS.html. (Accessed on 10/07/2022).
[16] Changquan Ding, Hang Liu, and Hengyu Li. 2019. Stitching of depth and color images from multiple RGB-D sensors for extended field of view. *International Journal of Advanced Robotic Systems* 16, 3 (2019).
[17] Konstantin Dragomiretskiy and Dominique Zosso. 2013. Variational mode decomposition. *IEEE transactions on signal processing* 62, 3 (2013), 531–544.
[18] Rudresh Dwivedi and Somnath Dey. 2019. Score-level fusion for cancelable multi-biometric verification. *Pattern Recognition Letters* 126 (2019), 58–67.
[19] Muhammad Ehatisham-Ul-Haq, Ali Javed, Muhammad Awais Azam, Hafiz MA Malik, Aun Irtaza, Ik Hyun Lee, and Muhammad Tariq Mahmood. 2019. Robust human activity recognition using multimodal feature-level fusion. *IEEE Access* 7 (2019), 60736–60751.
[20] Mica R Endsley. 2001. Designing for situation awareness in complex systems. In *Proceedings of the Second International Workshop on symbiosis of humans, artifacts and environment*. 1–14.
[21] Enrique J Fernandez-Sanchez, Javier Diaz, and Eduardo Ros. 2013. Background subtraction based on color and depth using active sensors. *Sensors* 13, 7 (2013).
[22] Enrique J Fernandez-Sanchez, Leonardo Rubio, Javier Diaz, and Eduardo Ros. 2014. Background subtraction model based on color and depth cues. *Machine vision and applications* 25, 5 (2014).
[23] Jose L Herrera, Carlos R Del-Blanco, and Narciso Garcia. 2018. Automatic depth extraction from 2D images using a cluster-based learning framework. *IEEE Transactions on Image Processing* 27, 7 (2018).
[24] Zhiyu Huang, Chen Lv, Yang Xing, and Jingda Wu. 2020. Multi-modal sensor fusion-based deep neural network for end-to-end autonomous driving with scene understanding. *IEEE Sensors Journal* 21, 10 (2020), 11781–11790.
[25] Shekh MM Islam, Naoyuki Motoyama, Sergio Pacheco, and Victor M Lubecke. 2020. Non-contact vital signs monitoring for multiple subjects using a millimeter wave FMCW automotive radar. In *2020 IEEE/MTT-S International Microwave Symposium (IMS)*. IEEE, 783–786.
[26] Zhenhua Jia, Musaab Alaziz, Xiang Chi, Richard E Howard, Yanyong Zhang, Pei Zhang, Wade Trappe, Anand Sivasubramaniam, and Ning An. 2016. HB-phone: a bed-mounted geophone-based heartbeat monitoring system. In *2016 15th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*. IEEE, 1–12.
[27] 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*. 1–13.
[28] Glenn Jocher, Alex Stoken, Ayush Chaurasia, Jirka Borovec, NanoCode012, TaoXie, Yonghye Kwon, Kalen Michael, Liu Changyu, Jiacong Fang, Abhiram V, Laughing, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Jebastin Nadar, imyhxy, Lorenzo Mammana, AlexWang1900, Cristi Fati, Diego Montes, Jan Hajek, Laurentiu Diaconu, Mai Thanh Minh, Marc, albinxavi, fatih, oleg, and wanghaoyang0106. 2021. *ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support*. https://doi.org/10.5281/zenodo.5563715
[29] Daejun Kang and Dongsuk Kum. 2020. Camera and radar sensor fusion for robust vehicle localization via vehicle part localization. *IEEE Access* 8 (2020), 75223–75236.
[30] Min-Sung Kim, Raza Haider, Gyu-Jung Cho, Chul-Hwan Kim, Chung-Yuen Won, and Jong-Seo Chai. 2019. Comprehensive review of islanding detection methods for distributed generation systems. *Energies* 12, 5 (2019), 837.
[31] Adam Krasuski, Andrzej Jankowski, Andrzej Skowron, and Dominik Slezak. 2013. From sensory data to decision making: A perspective on supporting a fire commander. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, Vol. 3. IEEE, 229–236.
[32] Mayank Kumar, Ashok Veeraraghavan, and Ashutosh Sabharwal. 2015. DistancePPG: Robust non-contact vital signs monitoring using a camera. *Biomedical optics express* 6, 5 (2015), 1565–1588.
[33] Huining Li, Chenhan Xu, Aditya Singh Rathore, Zhengxiong Li, Hanbin Zhang, Chen Song, Kun Wang, Lu Su, Feng Lin, Kui Ren, et al. 2021. VocalPrint: A mmWave-based Unmediated Vocal Sensing System for Secure Authentication. *IEEE Transactions on Mobile Computing* (2021).
[34] Minle Li, Yihua Hu, Nanxiang Zhao, and Qishu Qian. 2019. One-Stage Multi-Sensor Data Fusion Convolutional Neural Network for 3D Object Detection. *Sensors* 19, 6 (2019), 1434.
[35] Ping Li, Zhenlin An, Lei Yang, and Panlong Yang. 2019. Towards physical-layer vibration sensing with rfids. In *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, 892–900.
[36] Jialin Liu, Dong Li, Lei Wang, Fusang Zhang, and Jie Xiong. 2022. Enabling Contact-free Acoustic Sensing under Device Motion. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 6, 3 (2022), 1–27.
[37] Jonathan Long, Evan Shelhamer, and Trevor Darrell. 2015. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3431–3440.
[38] Ren C Luo, Tung Y Lin, and Kuo L Su. 2009. Multisensor based security robot system for intelligent building. *Robotics and autonomous systems* 57, 3 (2009), 330–338.
[39] Lucia Maddalena and Alfredo Petrosino. 2017. Exploiting color and depth for background subtraction. In *International Conference on Image Analysis and Processing*.
[40] K Venu Madhav, M Raghu Ram, E Hari Krishna, Nagarjuna Reddy Komalla, and K Ashoka Reddy. 2011. Estimation of respiration rate from ECG, BP and PPG signals using empirical mode decomposition. In *2011 IEEE International Instrumentation and Measurement Technology Conference*. IEEE, 1–4.
[41] Emanuela Marasco, Ayman Abaza, and Bojan Cukic. 2015. Why rank-level fusion? And what is the impact of image quality? *International Journal of Big Data Intelligence* 2, 2 (2015), 106–116.
[42] Armin MASOUMIAN, David GF MAREI, Saddam ABDULWAHAB, Julián CRISTIANO, Domenec PUIG, and Hatem A RASHWAN. 2021. Absolute Distance Prediction Based on Deep Learning Object Detection and Monocular Depth Estimation Models. (2021).
[43] Alican Mertan, Damien Jade Duff, and Gozde Unal. 2021. Single Image Depth Estimation: An Overview. *arXiv preprint arXiv:2104.06456* (2021).
[44] Mircea Paul Muresan, Ion Giosan, and Sergiu Nedevschi. 2020. Stabilization and validation of 3D object position using multimodal sensor fusion and semantic segmentation. *Sensors* 20, 4 (2020), 1110.
[45] Mojtaba Nazari and Sayed Mahmoud Sakhaei. 2017. Variational mode extraction: A new efficient method to derive respiratory signals from ECG. *IEEE journal of biomedical and health informatics* 22, 4 (2017), 1059–1067.
[46] Felix Nobis, Maximilian Geisslinger, Markus Weber, Johannes Betz, and Markus Lienkamp. 2019. A deep learning-based radar and camera sensor fusion architecture for object detection. In *2019 Sensor Data Fusion: Trends, Solutions, Applications (SDF)*. IEEE, 1–7.
[47] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. 2015. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*. 1520–1528.
[48] Edwin Olson. 2011. AprilTag: A robust and flexible visual fiducial system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 3400–3407.
[49] Muhammed Zahid Ozturk, Chenshu Wu, Beibei Wang, and KJ Liu. 2021. RadioMic: Sound Sensing via mmWave Signals. *arXiv preprint arXiv:2108.03164* (2021).
[50] Yashaswini Prathivadi, Jian Wu, Terrell R Bennett, and Roozbeh Jafari. 2014. Robust activity recognition using wearable IMU sensors. In *SENSORS, 2014 IEEE*. IEEE, 486–489.
[51] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.
[52] Joseph Redmon and Ali Farhadi. 2017. YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7263–7271.
[53] Respeaker. 2019. ReSpeaker 4 Mic Array for Raspberry Pi. (2019).
[54] Syed Aziz Shah, Ahsen Tahir, Jawad Ahmad, Adnan Zahid, Haris Pervaiz, Syed Yaseen Shah, Aboajeila Milad Abdulhadi Ashleibta, Aamir Hasanali, Shadan Khattak, and Qammer H Abbasi. 2020. Sensor fusion for identification of freezing of gait episodes using Wi-Fi and radar imaging. *IEEE Sensors Journal* 20, 23 (2020), 14410–14422.
[55] Hongming Shen, Chen Xu, Yongjie Yang, Ling Sun, Zhitian Cai, Lin Bai, Edward Clancy, and Xinming Huang. 2018. Respiration and heartbeat rates measurement based on autocorrelation using IR-UWB radar. *IEEE Transactions on Circuits and Systems II: Express Briefs* 65, 10 (2018), 1470–1474.
[56] Talha Ahmad Siddiqui, Rishi Madhok, and Matthew O'Toole. 2020. An extensible multi-sensor fusion framework for 3d imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 1008–1009.
[57] Stanford Artificial Intelligence Laboratory et al. 2018. *Robotic Operating System*. https://www.ros.org

[58] Texas Instruments. 2018. IWR1443 datasheet. https://www.ti.com/document-viewer/IWR1443/datasheet/power-consumption-summary-x7469#x7469. (Accessed on 11/04/2021).

[59] Asad Vakil, Jenny Liu, Peter Zulch, Erik Blasch, Robert Ewing, and Jia Li. 2021. A survey of multimodal sensor fusion for passive RF and EO information integration. *IEEE Aerospace and Electronic Systems Magazine* 36, 7 (2021), 44–61.

[60] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T Freeman. 2013. Phase-based video motion processing. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–10.

[61] Wei Wang, Alex X Liu, and Muhammad Shahzad. 2016. Gait recognition using wifi signals. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 363–373.

[62] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. 2015. Understanding and modeling of wifi signal based human activity recognition. In *Proceedings of the 21st annual international conference on mobile computing and networking*. 65–76.

[63] Xuyu Wang, Chao Yang, and Shiwen Mao. 2017. PhaseBeat: Exploiting CSI phase data for vital sign monitoring with commodity WiFi devices. In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 1230–1239.

[64] Yikai Wang, Wenbing Huang, Fuchun Sun, Tingyang Xu, Yu Rong, and Junzhou Huang. 2020. Deep multimodal fusion by channel exchanging. *Advances in Neural Information Processing Systems* 33 (2020).

[65] Ziqi Wang, Zhe Chen, Akash Deep Singh, Luis Garcia, Jun Luo, and Mani B Srivastava. 2020. UWHear: through-wall extraction and separation of audio vibrations using wireless signals. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 1–14.

[66] Björn Waske and Sebastian van der Linden. 2008. Classifying multilevel imagery from SAR and optical sensors by decision fusion. *IEEE Transactions on Geoscience and Remote Sensing* 46, 5 (2008), 1457–1466.

[67] Teng Wei, Shu Wang, Anfu Zhou, and Xinyu Zhang. 2015. Acoustic eavesdropping through wireless vibrometry. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*. 130–141.

[68] Dustin T Weiler, Stefanie O Villajuan, Laura Edkins, Sean Cleary, and Jason J Saleem. 2017. Wearable heart rate monitor technology accuracy in research: a comparative study between PPG and ECG technology. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 61. SAGE Publications Sage CA: Los Angeles, CA, 1292–1296.

[69] Xethru. 2019. X4M02 Datasheet. http://laonuri.techyneeti.com/wp-content/uploads/2019/02/X4M02_DATASHEET.pdf. (Accessed on 11/04/2021).

[70] Yang Xin, Lingshuang Kong, Zhi Liu, Chunhua Wang, Hongliang Zhu, Mingcheng Gao, Chensu Zhao, and Xiaoke Xu. 2018. Multimodal feature-level fusion for biometrics identification system on IoMT platform. *IEEE Access* 6 (2018), 21418–21426.

[71] Tianwei Xing, Luis Garcia, Marc Roig Vilamala, Federico Cerutti, Lance Kaplan, Alun Preece, and Mani Srivastava. 2020. Neuroplex: learning to detect complex events in sensor networks through knowledge injection. In *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*. 489–502.

[72] Chenhan Xu, Zhengxiong Li, Hanbin Zhang, Aditya Singh Rathore, Huining Li, Chen Song, Kun Wang, and Wenyao Xu. 2019. Waveear: Exploring a mmwave-based noise-resistant speech sensing for voice-user interface. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*. 14–26.

[73] Hongfei Xue, Wenjun Jiang, Chenglin Miao, Ye Yuan, Fenglong Ma, Xin Ma, Yijiang Wang, Shuochao Yao, Wenyao Xu, Aidong Zhang, et al. 2019. Deepfusion: A deep learning framework for the fusion of heterogeneous sensory data. In *Proceedings of the Twentieth ACM International Symposium on Mobile Ad Hoc Networking and Computing*. 151–160.

[74] Lei Yang, Yao Li, Qiongzheng Lin, Huanyu Jia, Xiang-Yang Li, and Yunhao Liu. 2017. Tagbeat: Sensing mechanical vibration period with cots rfid systems. *IEEE/ACM transactions on networking* 25, 6 (2017), 3823–3835.

[75] Yanni Yang, Jiannong Cao, Xiulong Liu, and Xuefeng Liu. 2019. Multi-breath: Separate respiration monitoring for multiple persons with UWB radar. In *2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. IEEE, 840–849.

[76] Shichao Yue, Hao He, Hao Wang, Hariharan Rahul, and Dina Katabi. 2018. Extracting multi-person respiration from entangled rf signals. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 2 (2018), 1–22.

[77] Youwei Zeng, Dan Wu, Jie Xiong, Jinyi Liu, Zhaopeng Liu, and Daqing Zhang. 2020. MultiSense: Enabling multi-person respiration sensing with commodity wifi. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 3 (2020), 1–29.

[78] Fusang Zhang, Daqing Zhang, Jie Xiong, Hao Wang, Kai Niu, Beihong Jin, and Yuxiang Wang. 2018. From fresnel diffraction model to fine-grained human respiration sensing with commodity wi-fi devices. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 2, 1 (2018), 1–23.

[79] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. 2018. Context encoding for semantic segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 7151–7160.

[80] Shujie Zhang, Tianyue Zheng, Zhe Chen, and Jun Luo. 2022. Can We Obtain Fine-grained Heartbeat Waveform via Contact-free RF-sensing?. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*. IEEE, 1759–1768.

[81] Yang Zhang, Gierad Laput, and Chris Harrison. 2018. Vibrosight: Long-range vibrometry for smart environment sensing. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 225–236.

[82] Yang Zhang, Sven Mayer, Jesse T Gonzalez, and Chris Harrison. 2021. Vibrosight++: City-Scale Sensing Using Existing Retroreflective Signs and Markers. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 1–14.

[83] Tianyue Zheng, Zhe Chen, Chao Cai, Jun Luo, and Xu Zhang. 2020. V2iFi: in-Vehicle Vital Sign Monitoring via Compact RF Sensing. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 2 (2020), 1–27.

[84] Tianyue Zheng, Zhe Chen, Shujie Zhang, Chao Cai, and Jun Luo. 2021. MoRe-Fi: Motion-robust and Fine-grained Respiration Monitoring via Deep-Learning UWB Radar. In *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*. 111–124.

[85] Tianyue Zheng, Zhe Chen, Shujie Zhang, and Jun Luo. 2022. Catch Your Breath: Simultaneous RF Tracking and Respiration Monitoring with Radar Pairs. *IEEE Transactions on Mobile Computing* (2022).