IRIS: Wireless ring for vision-based smart home interaction

Maruchi Kim*
Paul G. Allen School, University of Washington, Seattle, WA, USA
mkimhj@cs.washington.edu

Antonio Glenn*
Paul G. Allen School, University of Washington, Seattle, WA, USA
aglenn5@cs.washington.edu

Bandhav Veluri*
Paul G. Allen School, University of Washington, Seattle, WA, USA
bandhav@cs.washington.edu

Yunseo Lee
Paul G. Allen School, University of Washington, Seattle, WA, USA
yunseol1@cs.washington.edu

Eyoel Gebre
Paul G. Allen School, University of Washington, Seattle, WA, USA
eyoel23@cs.washington.edu

Aditya Bagaria
Paul G. Allen School, University of Washington, Seattle, WA, USA
abagaria@cs.washington.edu

Shwetak Patel
Paul G. Allen School, University of Washington, Seattle, WA, USA
shwetak@cs.washington.edu

Shyamnath Gollakota*
Paul G. Allen School, University of Washington, Seattle, WA, USA
gshyam@cs.washington.edu

Figure 1: Smart home interaction with IRIS. (A) A user unlocks the front door by pointing and clicking IRIS at the smart lock. (B) Another user points IRIS at a television and rotates their hand to adjust its volume. (C) The user points IRIS at their living room lights to turn them off before leaving home. (D) A user points and clicks IRIS at the blinds to lower them for privacy.

ABSTRACT

Integrating cameras into wireless smart rings has been challenging due to size and power constraints. We introduce IRIS, the first wireless vision-enabled smart ring system for smart home interactions. Equipped with a camera, Bluetooth radio, inertial measurement unit (IMU), and an onboard battery, IRIS meets the small size, weight, and power (SWaP) requirements for ring devices. IRIS is context-aware, adapting its gesture set to the detected device, and can last for 16-24 hours on a single charge. IRIS leverages the scene semantics to achieve instance-level device recognition. In a study involving 23 participants, IRIS consistently outpaced voice commands, with a higher proportion of participants expressing a preference for IRIS over voice commands regarding toggling a device's state, granular control, and social acceptability. Our work pushes the boundary of what is possible with ring form-factor devices, addressing system challenges and opening up novel interaction capabilities.

CCS CONCEPTS

• **Human-centered computing** → **Human computer interaction (HCI)**; **Ubiquitous and mobile computing**; • **Computing methodologies** → **Machine learning**; • **Computer systems organization** → **Embedded systems**; • **Hardware** → **Emerging technologies**.

KEYWORDS

Smart ring, context-aware interaction, low-power cameras, efficient deep learning, smart homes, IoT, wearables

*Corresponding authors

UIST '24, October 13–16, 2024, Pittsburgh, PA, USA

ACM ISBN 979-8-4007-0628-8/24/10
https://doi.org/10.1145/3654777.3676327

ACM Reference Format:
Maruchi Kim, Antonio Glenn, Bandhav Veluri, Yunseo Lee, Eyoel Gebre, Aditya Bagaria, Shwetak Patel, and Shyamnath Gollakota. 2024. IRIS: Wireless ring for vision-based smart home interaction. In *The 37th Annual ACM Symposium on User Interface Software and Technology (UIST '24), October 13–16, 2024, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3654777.3676327

1 INTRODUCTION

As households transition into smart homes, they are outfitted with an array of interconnected devices, like smart speakers, door locks, and other smart home appliances [9, 27]. While these devices promise improved convenience, the means by which users control them remain ripe for improvement. Issues such as social discomfort in using voice commands and the unreliability of voice input in noisy environments hinder seamless interaction [16, 33]. Despite the availability of smartphone apps for direct control, it is often more convenient to resort to traditional methods, such as light switches or dedicated remotes, rather than unlocking one's phones, locating the appropriate application, and navigating through its interface [25, 34].

We introduce IRIS, short for Interactive Ring for Interfacing with Smart home devices. IRIS is an end-to-end wireless ring system that supports real-time object instance detection using contextual scene semantics. The underlying principle is rooted in the age-old adage that a "picture is worth a thousand words," asserting that capturing images is far more efficient than verbalizing lengthy auditory commands. IRIS enables users to control smart home devices by simply pointing at the target device and performing a corresponding gesture, offering an intuitive alternative to traditional interaction methods. Finally, IRIS requires no additional setup beyond standard smart home device installation, and the experiments and results presented in this work were all conducted on existing smart home devices with no modification.

Achieving this is challenging for three key reasons. First, while sensors integrated in today's ring devices, such as IMUs, are low-power, camera hardware can generate significantly more data, leading to orders of magnitude higher power consumption [45]. It is unclear if wireless camera systems can be designed to meet the small size, weight and power requirements (SWaP) of ring form-factor devices. Secondly, while object detection systems excel in real-time detection, their limitations become evident when confronted with multiple instances of the same object class. For instance, merely determining that a user pointed at a set of blinds is inadequate; the system must also discern which specific set of blinds in the home the user intends to control. Therefore, a system capable of precisely identifying and distinguishing between individual devices of the same class is necessary. Finally, the entire end-to-end system should operate in real-time under around one second to be considered a seamless interaction modality [11, 26].

In IRIS, we address these challenges by making technical contributions spanning hardware, software, and system design. At a high level, the wearer points at the device of interest and presses the button on the ring, as shown in Fig. 1. This wakes up the on-board camera which captures a few frames of the target device, which are streamed to a nearby smartphone for processing via Bluetooth. The wearer also performs a gesture (e.g. rotation) to control the device, the intent of which is captured using the on-board IMU. This is also transmitted to the phone which then, in real-time, controls the target device. Specifically, we make three key contributions.

- **SWAP-constrained wireless camera ring.** We designed the first wireless ring form-factor device for vision-based smart home interaction (Fig. 2). Our hardware is equipped with a camera, Bluetooth radio, IMU, and an on-board battery, while meeting the small size, weight and power (SWaP) requirements expected in ring form-factor devices. We present cross-layer optimization methods across wireless camera hardware and firmware, and a user interface that enables extremely low-power states. We designed our DIY hardware using open-source eCAD software, outsourced fabrication and assembly, and 3D printed the enclosures. The final fabrication cost (PCB and components) was $471 for 20 units. The PCB, battery, and enclosure weigh 4 grams. IRIS is the first ring to stream camera data wirelessly, and operate for over 16 hours on a single charge.
- **Instance-level classification based on scene semantics.** Our machine learning (ML) pipeline surpasses traditional object detection models by enabling scene-based understanding and detection of specific instances within a class, all on resource-constrained devices. To achieve instance-level classification, we start by utilizing a self-supervised vision transformer model, DINOV2 [28], to generate scene-level embeddings. These embeddings capture not only the object itself but also the surrounding environment, providing valuable context and scene semantics. However, DINOV2's search runtime increases linearly with the size of the embedding database, introducing potential latency issues for an interactive system like IRIS. To address this challenge, we reduce the search space of DINOV2 by utilizing YOLO [31] to first detect all potential smart home devices within the image frame. Since YOLO can detect multiple objects, we use a centered-object detection algorithm (CODA). This analyzes the bounding boxes generated by YOLO and selects the object (smart device) closest to the image center. The classification from CODA serves to significantly reduce the search space for DINOV2's query. Our results demonstrate that this optimization effectively reduces DINOV2's query runtime by hundreds of milliseconds.
- **End-to-end system optimization for real-time operation and low-power.** Real-time operation is critical for interactive mobile systems [11, 26]. We optimize our wireless ring's streaming performance to maximize camera throughput, and show that the end-to-end system can control devices within one second, delivering near real-time feedback to users by confirming successful gesture recognition. Furthermore, we extensively optimized IRIS's low-power design for all day use, despite the limited on-board battery capacity.

Figure 2: IRIS hardware inside 3D-printed enclosure and when placed beside a quarter. The battery sits inside the band of the ring. The ring diameter and band thickness are 17.5 and 2.9 mm.

Put together, we design an end-to-end wearable ring system capable of (1) smart home device interaction that complements voice commands, (2) instance-based object detection to correctly distinguish between individual instances of the same class, and (3) real-time operation, on-device processing on a standard mobile

phone. We evaluate our system in five different homes from various angles and lightning conditions. Our real-world dataset also includes 98 examples of blinds, 57 doors, 34 door handles, 228 lights, 24 smart locks, 73 speakers, 64 televisions, 37 windows, and 161 background instances. Our results show that:

- Our wireless ring hardware can stream video at 3.4 frames per second to a phone, supports context-aware gestures, and can operate for 16-24 hours on a rechargeable 27mAh battery.
- A user study with 23 participants who performed 690 interactions with our ring hardware shows that participants generally favored IRIS over a voice assistant for controlling devices and greater social acceptability. IRIS performs in real-time, outpacing voice commands by 2 seconds on average, from command initiation.
- Our system enables instance-level detection, while optimizing for runtime performance. Across 18 unique instances of devices, including 2 HVACs, 2 blinds, 1 door, 4 lighting systems, 2 smart locks, 5 speakers, and 2 TVs, the instance-based classification accuracy was 95% and 98% when provided with 2 and 3 reference images, respectively. Further, we show an average inference latency reduction from 411ms for the DINOV2 model to 112ms for our DINO+YOLO+CODA system.

We believe that this paper introduces a novel approach to smart home device interaction through IRIS – a wireless, low-power, camera-enabled ring backed by real-time instance detection and contextual scene semantics. By working across both hardware and software, IRIS provides an alternate modality to existing voice command and app-based interactions. Our results highlight IRIS's viability, showcasing its real-time responsiveness and user experience achieved with optimization techniques across hardware, firmware, and ML runtime. With the potential to impact the landscape of ring-based human-computer interaction, we believe that IRIS has the potential to be an important step toward a more natural, unobtrusive, and user-friendly interface for smart homes.

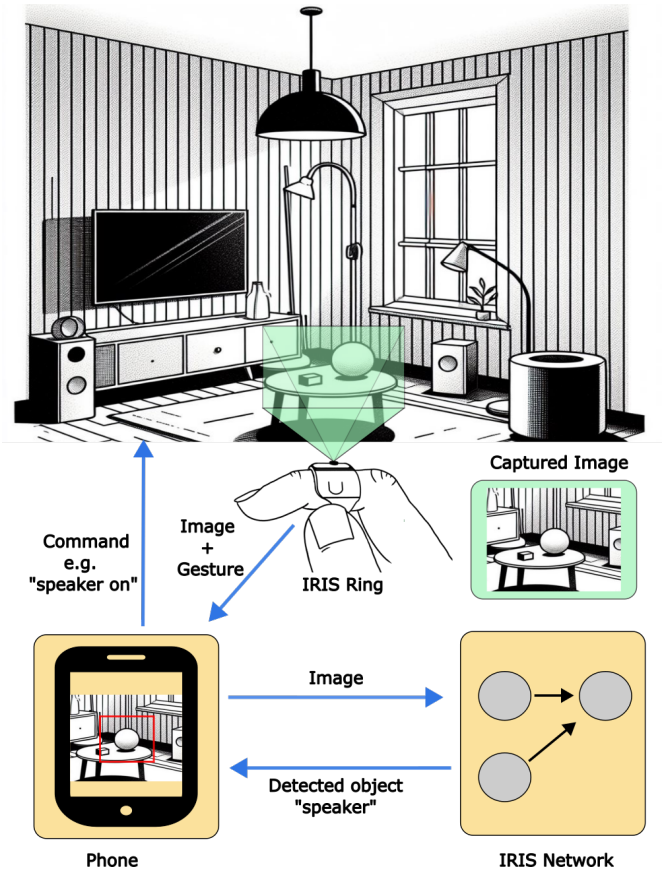

Figure 3: IRIS in Action: Context-aware smart home control. This figure illustrates the application of IRIS, a custom-designed, camera-enabled wireless ring for context-aware control of the smart home. IRIS utilizes instance-based object detection for real-time interaction with the environment.

2 RELATED WORK

To the best of our knowledge, we present the first wireless ring form-factor system for vision-based smart home interaction. Below we present related works across IoT and ring prototypes.

Voice interaction for IoT devices. A common modality for interaction with Internet-of-Things (IoT) devices in smart-homes continues to be through voice-based IoT devices such as Google Home [39], Apple Home Pod [3], and Amazon Alexa [2]. As noted in [19], as the number of IoT devices in the home continues to grow, using voice, and associated apps, to control these object places a burden on the mental load of users which limits their scale.

Cameras for IoT interaction. Cameras can enable contextual understanding of gestures and intent. Snap-to-It [8] and SnapLink [6], allow users to take photos of the IoT devices from their smart phone cameras. However, a fully wireless hand or ring based wearable integrated with a camera for interaction has not yet been realized due to SWaP challenges.

CyclopsRing [5] uses a palm-facing wired RBG camera with a fish eye lens placed between the fingers to capture whole hand gestures. To identify objects for interaction in the environment, CyclopsRing detects tags [12] placed on objects. The camera however is wired to a power source and hence the design do not consider the small size, weight and power (SWaP) constraints of designing an end-to-end wearable ring system, which can drastically change the design decisions. ThermalRing [49] uses a wired infrared (IR) camera to capture IR energy emitted from the wearer's opposite hand to track the hand motion and perform gesture input. For object detection and identification, ThermalRing requires objects to be retrofitted with IR reflective patches called ThermalTags similar to the tags used in in CyclopsRing and elsewhere [23]. As before, ThermalRing uses wired cameras and does not address the SWaP constraints. FingerReader and FingerReader2.0 [4, 36] are wired camera rings designed for visually impaired users to read text or aid in shopping. FingerSight [38] uses a camera-enabled ring for haptic sensing of distant objects, and TouchCam [37] combines IMUs and cameras for gesture support and body location classification. FingerReader, FingerSight, and TouchCam are all wired devices, tethered to a host machine or smartwatch. Unlike these works, we create a fully wireless, standalone, camera-enabled ring that meets the SWaP requirements expected in this form factor.

EyeRing [24] designs a wireless camera-enabled ring for accessibility applications such as navigation, currency detection, and color detection. However, the prototype device is significantly large

(around the size of a smart watch) and does not address the size and weight constraints for ring devices. Further, EyeRing does not support interaction and control of IoT devices. Unlike all this previous work, IRIS does not require instrumenting existing objects in the environment with fiducial markers and can instead recognize objects using vision. Contrary to conventional wisdom, we show that a fully wireless, camera-enabled, wearable ring that meets the SWaP requirements expected in this form factor is possible. Cameras consume significant power, and given the small battery capacities capable of fitting a ring form factor (<27mAh), our work introduces system-level optimizations necessary to integrate a camera into a ring without being tethered to another device.

Other sensors for IoT interaction. SeleCon [1], WristQue [21], and Minuet [19] use ultra-wideband (UWB) equipped watches to enable pointing gesture detection for IoT device selection and control. SeleCon requires instrumenting each IoT device with a UWB radio which limits its scalability. WristQue [21] and Minuet [19] require UWB modules on the user and in the environment to determine the device selection from pointing gestures. Minuet [19] utilizes multi-modal interaction by also capturing the users' voice. Despite the appeal of these UWB enabled point-to-select wearables, all these prior prototypes are wired and/or do not demonstrate real-time operation, and hence do not address the power or latency requirements of end-to-end interactive ring devices. RingIoT [7] consists of a ring instrumented with an IMU, IR transmitter, capacitive sensor, and OLED display. This again uses a wired setup for both power and data transfer and hence does not consider the SWaP requirments in its design. TRing [48] enables device interaction by sensing embedded magnets placed within everyday objects. Similar to [7], this approach is limited as it requires instrumenting devices with specialized sensors. Furthermore, TRing cannot enable interaction from a distance as the magnetic field sensing is limited to the near-field. Magic Ring [18] has an onboard radio and antenna to communicate with an intermediary device that translates the RF signals from the ring into infrared signals used to control the IoT devices and appliances in the home such as TVs or fans. Ring Zero [10] is a commercial ring that proposes to use an onboard IMU to capture gestures and enable fine grained control over music volume and light brightness, however it is unclear how the device selection mechanism is implemented and whether it can enable ad-hoc control over any IoT device without pairing.

3 IRIS

Here, we begin by outlining the requirements for our system design. We then present our wireless ring hardware system and finally describe our real-time neural network pipeline.

3.1 System Requirements

Our proposed interaction modality should be as ubiquitous as voice, while eliminating the need for lengthy voice commands. The input modality should be as simple and swift as flicking a light switch or rotating a dial, ensuring a socially seamless experience. Finally, the system should require no modifications to the smart home devices as that would place an unnecessary burden on device manufacturers. We arrive at three core design principles:

(1) A simple interface as intuitive as flicking a switch
(2) Ubiquity comparable to voice interaction
(3) No modifications to existing IoT devices

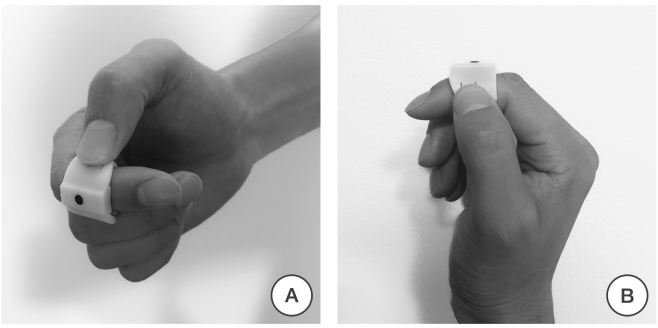

Figure 4: IRIS on a User's Hand. (A) Front View, (B) Top View.

To address this, we design a wireless, vision-enabled smart ring. Achieving this requires us to address the following challenges.

- *Size, weight, and power requirements.* The overall weight and size of the device should be in a similar ballpark as a regular-sized ring. While shrinking the ring to fit within these constraints is challenging, we must also consider all-day battery life so that it does not need to be charged in the middle of the day.
- *Real-time operation.* IRIS must be quick and responsive. An upper bound to not interrupt the user's flow of thought is around one second [26]. So, we target the maximum end-to-end latency for IRIS to be one second, and benchmark it against voice assistant solutions.
- *Instance-based object detection.* An image may not necessarily provide the specificity of a voice command. For example, in a scenario where a user has a set of identical blinds in two different rooms, they could say, "lower the blinds in room A" or "lower the blinds in room B." IRIS instead must detect which instance a particular object is based on the surrounding visual context.

3.2 Wireless vision-enabled ring hardware

Here, we describe the various aspects of our wireless ring design.

3.2.1 Hardware. Our custom hardware design is comprised of an ultra-low-power 1/11" 320x320 QVGA CMOS image sensor (Himax HM01B0), a 6-axis inertial measurement unit (Bosch BMI270), and a Bluetooth Low Energy (BLE) microcontroller (Nordic nRF52840). The system is powered by a 27mAh battery and programmed via SWD over a Micro-USB connector. Our design is fully rechargeable through an on-board power management integrated circuit and provides the system with all necessary voltage rails (Maxim MAX77650). A single-pull single-throw (SPST) switch is used for gesture initiation, and images are streamed to a mobile phone for input into our neural network (see Fig. 3).

3.2.2 Wireless Latency. The first challenge with our design is meeting a <500ms image acquisition latency target. IRIS utilizes the maximum supported data rate of 2 Mbps over BLE [35]. To maximize throughput, we utilize the shortest connection interval available to iOS (15 ms) while transmitting 4 packets per interval (maximum supported by iOS) [41]. With a packet size of 247 bytes the effective data rate is 526,933 bits per second. The effective BLE throughput

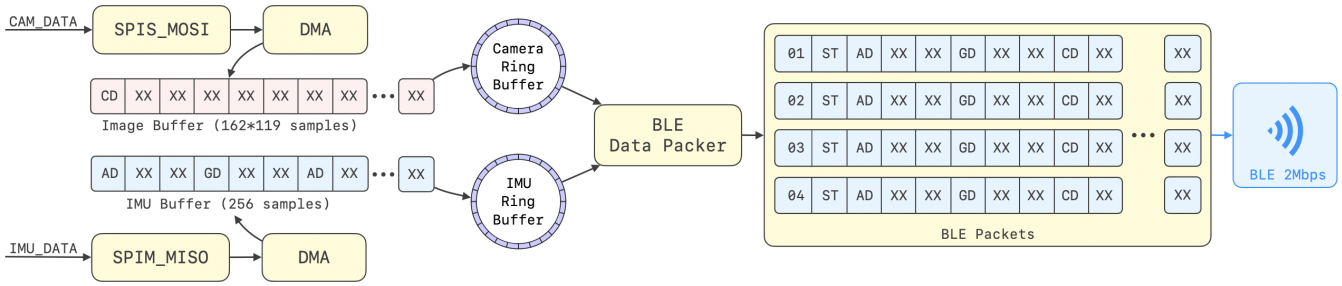

Figure 5: IRIS image and IMU wireless data streaming. BLE packets are formed with the following structure: ST-Status Flags (Start of Frame, IMU Valid, Button State), AD-Accelerometer Data, GD-Gyroscope Data, CD-Camera Data.

can be written as:

$$\text{Throughput} = \frac{1000\,\text{ms} \times \text{packets\_per\_interval} \times \text{packet\_size} \times 8}{\text{connection\_interval (ms)}}$$

The challenge is that a full 320x320 image is 819,200 bits, translating to 1562.5ms of latency (0.64fps) for the full image to be transmitted to the phone. So, we need to reduce the image size while preserving as much information as possible. To do this, first, we enable the QVGA window readout on the image sensor, which reduces the resolution to 320x240. This is still insufficient, so we utilize pixel binning to improve image acquisition time. Pixel binning is the concept of combining electrical charges from multiple adjacent pixels into a single "superpixel" [20, 47]. Pixel binning essentially increases the effective pixel size, providing an improvement in SNR (signal-to-noise ratio) and low light. Furthermore, pixel binning also enables faster frame rates. By combining pixel data, the sensor can read out information from a smaller number of "superpixels" compared to the original number of individual pixels. This reduced readout time effectively leads to faster frame rates.[1] Thus, while binning reduces the resolution of our system by a factor of four to QQVGA (160x120), it offers two key advantages: (1) a 4x improvement in signal-to-noise ratio, and (2) a 4x increase in frame rate. In this configuration, our image resolution is now 160x120 bringing the frame rate to 3.43fps, or about 290ms of end-to-end latency. This meets our design goal of less than 500ms of latency with some margin to spare.

3.2.3 CMOS image sensor integration. Since the BLE SoC (nRF52840) has no dedicated camera interface, we configure the CMOS sensor (HM01B0) to operate in a 1-bit data transfer mode. The HM01B0 can then effectively act as a SPI controller and pass data to the nRF52840 through a SPI (SPIS) port. This is achieved through signalizing the start of frame with a rising edge of Frame Valid (FVLD), and receiving the image bits through Pixel Clock Out (PCLKO) and Data (D0). These map to CS, SCLK, and MOSI, respectively. The table below shows a clearer representation of the mapping between a 1-bit camera interface and SPI:

1-Bit Camera Interface	Serial Port Interface (SPI)
Pixel Clock Out (PCLKO)	Serial Clock (SCLK)
Data (D0)	Master-Out-Slave-In (MOSI)
Frame Valid (FLVD)	Chip Select (CS)

Since the maximum SPI port clock frequency on the nRF52840 is 8MHz, we limit the pixel clock frequency accordingly. We achieve this by inputting an 8MHz signal into the HM01B0's clock (MCLK) pin from the nRF52480. This synchronizes the two chips together and also limits PCLKO to 8MHz. The final detail to make this bus work is to invert the FVLD signal. The nRF52840 only supports an active low CS line, whereas FVLD is an active high signal. We circumvent this by triggering an active-high interrupt from FVLD and loop an external GPIO back to the nRF52840 to trigger an active-low interrupt for the SPI port to start receiving data. This could be optimized by incorporating a NOT gate between FVLD and CS [15].

3.2.4 Low-power design. IRIS manages power consumption through its three distinct power states: SLEEP, IDLE, and ACTIVE. In the SLEEP state, IRIS conserves energy by deactivating all hardware components except for an internal timer within the nRF52840 chip. This timer periodically awakens IRIS to check for the presence of a home WiFi network by receiving data from the connected mobile phone. While away from the user's home WiFi network, IRIS remains in this energy-saving mode, transitioning to IDLE only when a home WiFi connection is established.

In the IDLE state, IRIS enables all power rails and readies all peripherals, but clock-gates the camera and suspends the IMU. This ensures responsiveness while minimizing energy consumption. IRIS exits the IDLE state with a single button press via a hardware interrupt, transitioning it into ACTIVE mode. During this mode of operation, IRIS continuously streams data such as button state, IMU data, and camera pixels to the connected mobile device. After 3 seconds of streaming and inactivity from the button, IRIS exits ACTIVE and returns to IDLE to optimize low-power performance.

Our low-power design allows IRIS to operate continuously for 16-24 hours, depending on usage. Additionally, IRIS supports rapid charging capabilities, achieving a full recharge in just one hour with a 1C charge current, ensuring usage for extended periods.

3.2.5 Fabrication. The hardware schematic and layout for IRIS were designed using the open-source eCAD tool KiCad. A 2-layer flexible printed circuit was fabricated by PCBWay, while assembly was done by a local assembler. The 3D-printed enclosures were designed using AutoDesk Inventor and printed with a Formlabs Form 3 B printer using a liquid resin fabrication process. As illustrated in Fig 4, the camera sits behind the lid on the ring's outer surface, and the button is placed closer to the user's thumb while remaining horizontally aligned with the camera.

3.2.6 Context-aware gestures. We employ a simple gesture set that is adapted to the detected device. This set comprises two intuitive

[1] In IRIS, the BLE throughput is 527kbps, while the HM01B0 camera can stream a 320x320 image at 60fps (6Mbps). Reducing the resolution down to 160x120 allows IRIS to transfer more total frames over time.

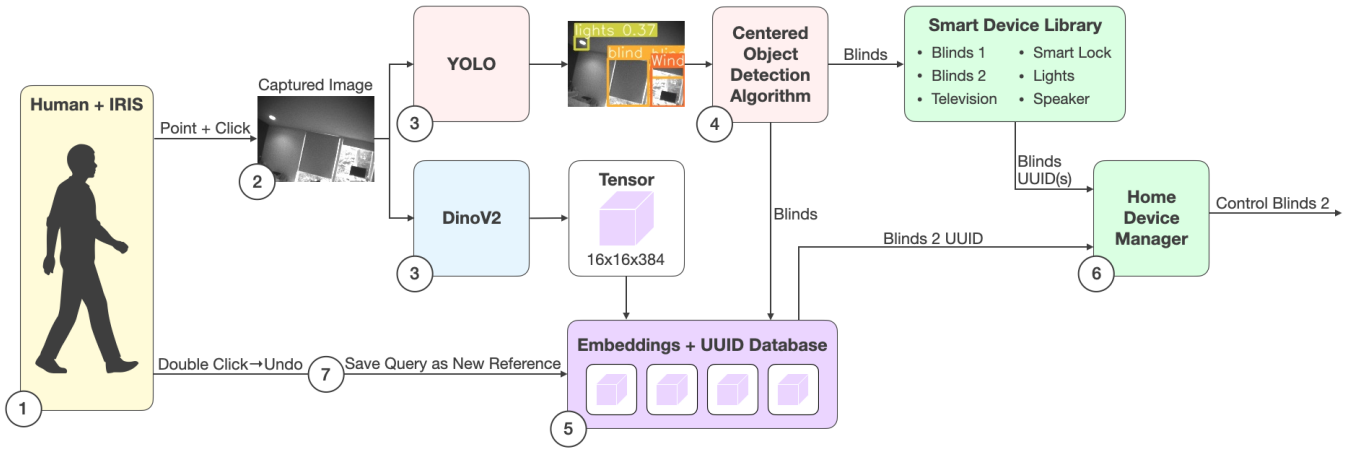

Figure 6: IRIS pipeline. (1) A user points and clicks at the smart device they would like to control. (2) IRIS wirelessly streams the images to a smartphone, and (3) runs YOLO and DinoV2. (4) The centered object detection algorithm (CODA) filters out the multiple objects YOLO may detect and outputs the object closest to the center of the frame. IRIS then queries the smart device library, but since, in this example, there are two instances of Blinds in the home it stops here and utilizes the Dinov2 path. Next, the output embedding from Dinov2 is passed as input to (5) the Embedding + UUID database to find the embedding with the highest similarity, and the output of CODA is also passed as input to reduce the search space. The highest similarity corresponds to Blinds 2 UUID, and the (6) Home Device Manager controls Blinds 2.

gestures: a single press for toggling device state and a press-and-hold with rotation, mimicking the action of a physical dial, for fine-grained control. Table. 1 illustrates how these two gestures can be used across an array of IoT devices. A single press toggles the binary state of the device, while a press-and-hold with rotation allows for granular control. The rotation gesture was calculated by first deriving the tilt (in degrees) for the IMU axis that corresponded to rotation about the wrist from the accelerometer values using trigonometry. This tilt value was then offset by 90 degrees and scaled between 0 and 180 degrees. A running-difference of changes in this scaled tiled value was maintained and each incremental change in tilt was mapped to an increment change in volume or brightness. While the gesture set could be readily extended to additional devices such as garage doors, HVAC systems, and smart appliances, this work focuses on these five devices to maintain a manageable scope and data collection effort.

Device	Single Press	Rotation
Lights	On/Off	Brightness
Speaker	Play/Pause	Volume
Smart Lock	Lock/Unlock	-
TV	On/Off	Volume
Blinds	Up/Down	-

Table 1: Context-aware gesture set.

3.3 Neural Network Pipeline

Our system was designed with several key requirements. First, we aimed for a baseline level of "out-of-the-box" functionality to minimize setup for end-users, addressing one of the main pain points associated with new devices [46]. Secondly, the vision model needs to support instance-based object detection to differentiate between multiple smart devices of the same class within the same environment.

Our pipeline consists of a fused model comprised of YOLOv8 [43] and DinoV2 [28]. A YOLOv8 model was fine-tuned on a custom dataset of 10 classes to control a set of five common smart home devices. A fine-tuned YOLO model enables us to address the first requirement of out-of-the-box functionality. However, we still need to address our other requirements, and YOLO fails to distinguish between two instances of the same class and also doesn't improve without training on additional data. Thus, we augment our system with Dinov2 to address the shortcomings of YOLO. To distinguish between two instances of the same class, IRIS provides a scanning feature, which enables users to take pictures when they have multiple instances of the smart device in their home. We create an embedding database with these collected pictures, and users can map reference images to a specific instance of a smart home device. Finally, users can address specific failure scenarios by adding additional reference images into the embedding database.

3.3.1 Out-of-the box object detection. Our primary focus was on object detection of five household objects: TVs, smart locks, blinds, lights, and speakers. We experimented with several well-known model architectures such as ResNet18 [14], ResNet50 [14], EfficientNetB0 [40], and YOLOv8 [43]. We moved forward with YOLO as it is an object detection model, allowing us to design a heuristic to control a specific device when multiple objects of interest are in the image. While YOLOv8 is trained on the COCO dataset [43], it lacked support for the five smart home devices we aimed to detect. To overcome this limitation, we collected over 2,000 images using a scraper on Google Images and approximately 500 images directly from IRIS's camera. The output of YOLO goes into an algorithm which iterates through each bounding box to retrieve the object that is closest to the center of the frame. This is achieved by calculating the Euclidean distance between the center of each bounding box with respect to the center of the image. This heuristic called CODA (Centered Object Detection Algorithm), allows IRIS to return the object that is closest to what the user pointed at.

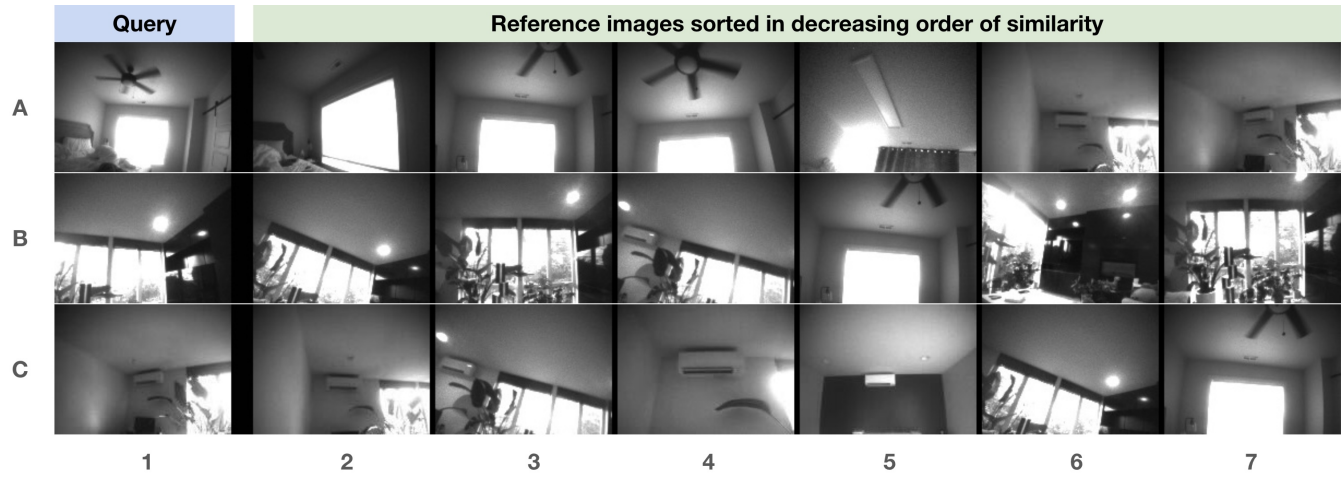

Figure 7: DINOV2 queries based on semantic similarity. Row (A) shows a query image of a blind (A1), while the maximum similarity reference image shows the same instance of the blind (A2). Row (B) shows a different instance of a blind (B1), and similarly shows how semantic similarity is used to find the reference image that represents the same instance (B2). Row (C) shows an image of an HVAC unit (C1), a class which is unlabeled in our YOLO dataset. Again, we observe that the same instance of the HVAC is returned (C2). A different instance of an HVAC (C5) further shows that semantic scene understanding can be used to distinguish between two instances of the same class.

3.3.2 Human-in-the-loop learning with semantic similarity. The above bounding box results combined with our heuristic disambiguate input frames with multiple objects, and provide us with a single object classification. This classification is sufficient in scenarios where there is one object (device) per class that the user wishes to control. However, if a user seeks to control multiple devices belonging to the same class (e.g., two TVs, one in the living room and the other in a bedroom), we would need to classify specific instances of the objects; beyond just object-level classification. Taking this into account, in addition to our out-of-the-box solution based on a pre-defined set of classes, we implemented a human-in-the-loop model that allows users to define a separate class for each device.

The intuition behind such a capability is that a device in a home is characterized by not just its own visual features but that of the surroundings it is placed in as well. For example, two smart speakers of the same make could be distinguished by the visual characteristics of the room it's placed in. This necessitates us to extract not just the features of the object of interest in the frame, but the features of the entire frame. We leverage advances in large-scale self-supervised learning of visual features such as CLIP [30] and DINOv2 [28] to compute semantic features at the image level. Specifically, we use the open-source DINOv2 [28] model in our implementation, to obtain image-level semantic features. For a given image resized to 224x224 resolution, we split the image into 4x4 grid and compute a R^{382} embedding for each patch in the grid. We note that the embedding corresponding to each patch considers not just the visual features within the patch but their relation to the rest of the image patches as well.

The user would first perform a setup with the IRIS system by capturing a few (1-5) images for each device they wish to control. During this setup, the IRIS app computes a $R^{4\times4\times382}$ DINOv2 embedding for each reference image. For a smart home with N connected devices placed at certain locations in the home, our system would create a database of a set of reference embeddings $\mathcal{X} = \{r_{ni} \in R^{4\times4\times382} \mid n \in \{1, 2, .., N\}; i \in \{1, .., 5\}\}$ on the smartphone. Then during the regular use, the user points at the device they wish to control by pointing the IRIS ring at the device. This action leads the ring to capture the image of the scene and send it to the IRIS app. Then, as shown in step 3 in Fig. 6, the app computes a query embedding $q \in R^{4\times4\times382}$ corresponding to the frame captured by the user. In parallel, the YOLO model predicts which class the query belongs to. Based on this prediction, we compute a semantic similarity score between q and each of the embedding belonging to the predicted class in $\mathcal{X}$ using the following algorithm:

```
import torch
import torch.nn.functional as F
def compute_similarity(q: torch.Tensor,
                       r: torch.Tensor):
    q = q.view(-1, 382) # shape = [16,382]
    r = r.view(-1, 382) # shape = [16,382]
    scene_sim = 0.0
    for pe in q:
        # pe is patch embedding
        scene_sim += F.cosine_similarity(
            pe.unsqueeze(0), r, dim=-1
        ).max()
    scene_sim /= q.shape[0]
    return scene_sim
```

This step results in a set of similarities corresponding to each of the reference embeddings in $\mathcal{X}$. If s_{ni} is the similarity corresponding to the ith reference image of nth device, then the device that the user wants to control, $\hat{n}$, could be inferred with:

$$\hat{n} = \arg\max_{n} \{s_{ni} \in [0.0, 1.0] \mid n \in \{1, 2, .., N\}; i \in \{1, .., 5\}\}$$

This approach is illustrated in Fig. 7. On the left most column, we show query images that the ring captured during its regular use. The rest of the columns show different reference images captured during the setup. In each row, the reference images are sorted in decreasing order of similarity with the query image on the left. We can observe that the similarity accounts for not just the objects of interest, but also the surroundings the object is placed in.

3.3.3 Latency reduction by combining the models. A key challenge associated with embedding-based retrieval methods (like DINOV2), lies in the search time increasingly linearly with database size. To address this, IRIS leverages the output from YOLO+CODA, which identifies the object closest to the image center. The output classification from CODA serves as a proxy for the user's point of focus, enabling us to reduce the search space within the user-collected reference images. This targeted search strategy noticeably reduces the runtime required for DINO's query, enhancing end-to-end system latency without compromising accuracy. We evaluate this method across 86 reference images in §4.3.1.

3.4 Training Methodology

While the COCO dataset YOLOv8 is trained on includes appliances in its set of 20 classes, it lacked support for the five smart home devices we aimed to detect. Furthermore, we do not require the detection model to predict bounding boxes for objects not in our five classes of interest. To address these aspects, we fine-tune YOLOv8 by collecting pictures using a scraper on Google Images. While these web-scraped images were easy to collect, they fail to model our camera's characteristics like field-of-view, dynamic range, and low-light sensor noise. Thus, we also collected and created a dataset directly from IRIS's camera. We adopt a hybrid training methodology where we first train on our web-scraped dataset, and then fine-tune on real data from our hardware. Our results show that our network trained in this way generalizes to real-world images captured from IRIS's camera.

3.4.1 Web-scraped data. We collected over 2000 images of our desired classes. In total, this dataset contains 287 lights, 309 speakers, 521 smart locks, 178 door handles, 32 doors, 198 blinds, 197 televisions, and 210 windows from Google Images. We include doors and door handles as it is difficult for YOLO to accurately detect smart locks from far away due to their small size, ultimately resolving to few features from far distances. Additionally, being able to detect windows allow us to reject false positives (e.g. sunlight being recognized as a home light source). The system can fall back to detecting doors at far distances to ultimately trigger a smart lock. These images were then converted to greyscale and resized to 160x160. These transformations were done to mimic IRIS's camera specifications. Further, we applied data augmentations such as horizontal flipping, cropping (0-20%), rotations (-15 deg, 15 deg), brightness (-15%, 15%), and exposure (-10%, 10%). These data augmentations bring the web-scraped dataset to a total of 3,526 images. We split this data into 90% training and 10% validation subsets.

3.4.2 In-the-wild data. While the model trained on our web-scraped data was able to detect objects from our 5 desired classes correctly, it failed to adapt to image characteristics of IRIS's camera. This is because the web-scraped images do not contain characteristics such as the camera's dynamic range, resolution, and sensor noise. To address this, we further fine-tune on a dataset of captured images from our ring. We collected 513 images across 5 different homes from various angles and lightning conditions to generalize to unseen environments. This dataset includes 98 blinds, 57 doors, 34 door handles, 228 lights, 24 smart locks, 73 speakers, 64 televisions, 37 windows, and 161 background (null) instances. Background images contain none of the objects that IRIS aims to detect; this is so that the model outputs fewer false positives. After data augmentations, our in-the-wild dataset comprised of 1225 total images. This dataset was split for 86% training, 7% validation, and 7% testing subsets.

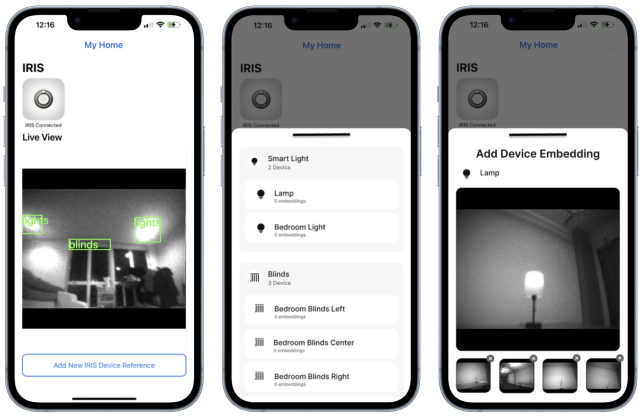

Figure 8: IRIS mobile app. (Left) Default view when user opens the app. Users can observe the IRIS camera output; primarily for debugging and experimental use. (Middle) User can select a specific device to take new reference images for. (Right) User views the IRIS camera through the app to create new reference images.

3.5 Mobile Device Integration

Upon initial app launch and relevant user permissions, i.e., accessing control of smart home devices and Bluetooth, the HomeKit framework retrieves information on user-registered homes, smart home devices, their capabilities, configurations, and unique identifiers. Using this metadata, we build a smart device library and an embeddings+UUID database which our Home Device Manager uses to control the desired end device.

After IRIS connects to the iPhone, it waits for the button to be pressed prior to sending data wirelessly. Once a user presses the button to initiate a gesture, IRIS enters the ACTIVE state and starts streaming wireless data packets, which contain a sequence number, status flags, IMU vectors, and image pixels. Each image frame is split and transmitted across multiple data packets, and the sequence number is a monotonically increasing 8-bit value used to invalidate image frames in the event of dropped packets. Status flags include fields such as the button state and start-of-frame. Since IRIS continuously streams images, the start-of-frame flag marks the starting packet of a new image frame and acts as the boundary between consecutive frames. The start-of-frame flag also signals the end of the previous frame to start inference. The data pipeline is shown in Fig. 5.

Our fine-tuned YOLOv8 model's weights are transformed to a CoreML Package which we then load into Apple's VNCoreMLModel to perform requests and receive predictions and their bounds/bounding boxes. The raw pixel values of the streamed images are transformed into a bitmap image called CGImage and preprocessed into the desired shape requested by the model. CODA then finds the center object by searching for the bounding box closest to the image center, and returns this as the classified object.

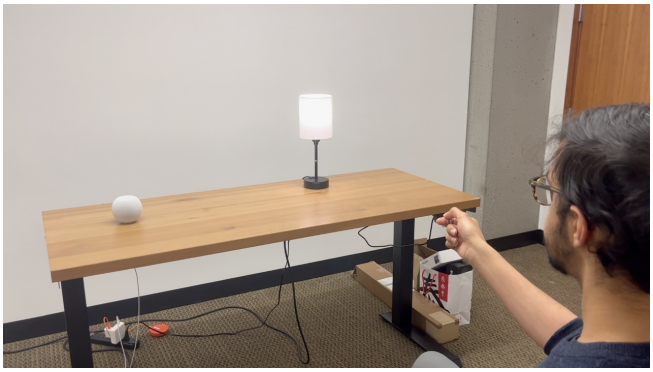

Figure 9: IRIS User Study. A participant from our user study interacts with a light and a speaker using our IRIS hardware.

For IRIS to detect between multiple instances of the same class, the system requires reference images (embeddings) of the device to be saved to persistent storage. We implement a "scan mode" through the app, during which the user can point-and-click at the target device and scene to create embeddings via DINOv2. Users can then associate these reference images with a target device in their home. Due to limitations of CoreML, we relied on ONNX Runtime for simplified cross-platform machine learning integration to host our DINOv2 model. With the image preprocessing baked into the DINOv2 ONNX model, we pass in the raw pixel values to create the embeddings which will then be automatically compared against stored references.

Ultimately, the retrieved embedding with the highest similarity acts as a key, which is then used to fetch the unique identifier (UUID) of the targeted smart device within the user's home. After retrieving the UUID, we control the corresponding device via HomeKit.

4 EVALUATION

4.1 User Study

We recruited 23 participants (16 male, 7 female) aged 18-35 (σ=4.72) for a user study. Our study included an initial placement study to understand user preferences for the placement of IRIS along the index finger. The second part involved participants interacting in real-time with smart devices with IRIS and their voice. The participants were also provided a short questionnaire asking them about their overall experience and to compare IRIS against voice interaction. Prior work [32] has shown that smartphone apps and screen-based interactions are even slower than voice control, and as such, were omitted from this study.

Question	Middle	Proximal	P-value
Comfort	52.2%	47.8%	0.339
Targeting	**73.9%**	26.1%	**0.005**
Preference	43.5%	56.5%	0.202

Table 2: Placement study results.

4.1.1 Placement study. Rings are typically worn on the proximal digit of a user's finger. However, conventional rings are not designed for aiming and controlling smart devices. While creating our hardware, we hypothesized that it would be easier to aim at devices and have more range of motion in one's hand if IRIS were placed on the middle digit of the index finger. We provide participants with a set of metal rings (similar to IRIS's weight) for sizing their middle and proximal digits. Two markers on the rings indicate camera and button positions.

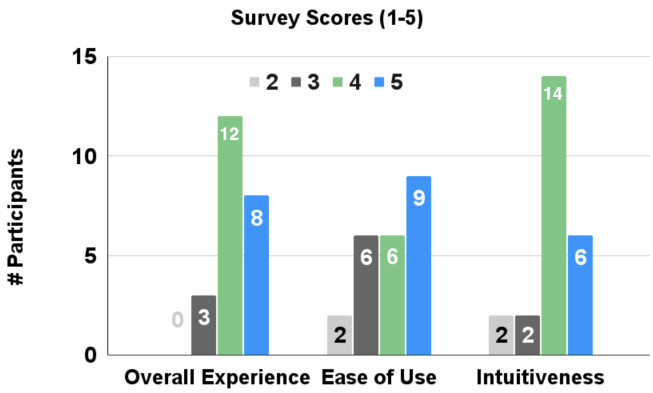

Figure 10: User interaction with our IRIS hardware. Y-axis shows number of participants who ranked using a score from 1-5. Scores for 1 have been omitted as no participants responded with 1.

Once two rings were chosen, we asked participants to imagine that these mock rings were smart rings designed to control smart devices. We informed participants of the gesture set (point-and-click, point-hold-rotate, double-click), and had them perform these gestures on a lamp and a speaker. After evaluating both placements, they were provided with a short qualitative questionnaire.

The question set included: (1) *Which configuration felt more comfortable for use?*, (2) *Which configuration did you have an easier time "aiming" the camera with?*, and (3) *Between the two configurations, which did you prefer?*. The responses to these questions are shown in Table 2. For each question, we conducted a one-tailed binomial test to understand if the result was statistically significant. Overall comfort was comparable between the two configurations with basically an even split across the 23 participants (52.2%/47.8%). While, 56.5% of participants preferred the proximal position due to convention, this result isn't statistically significant enough to design the ring in this position. Ease of targeting the smart device between the two configurations was the one statistically significant result of this study where 73.9% of participants preferred the middle digit. This result corresponds to a p-value of 0.005.

4.1.2 Interaction using IRIS hardware and voice. The goal of the next part of our study was to compare IRIS against a commercial voice assistant. As our neural network implementation and home device management were implemented on iOS, we opted to compare against Siri. Benchmarking against Siri guarantees that the communication under the hood (i.e., HomeKit) remains the same between the two interfaces.

This next phase of our study begins with introducing IRIS to the participants. To familiarize the participants with IRIS, we first play a short 30-second demonstration video. Next, we permitted them to try out IRIS by controlling a pair of smart devices, lights and speaker. After confirming their comfort and confidence using IRIS, participants engaged in a series of trials. The first trial involved a proctor who directed the participants to control either the lights or the speaker, toggling the device's state in response

	IRIS	Siri	No Diff	P-value
Toggling State	56.52%	39.13%	4.34%	0.202
Granular Control	78.26%	17.39%	4.34%	**0.001**
Social Acceptability	69.57%	13.04%	17.39%	**0.017**
	IRIS	**Siri**	**Both**	**P-value**
Day-to-day Use	34.78%	17.39%	47.82%	0.196

Table 3: Head-to-head subjective performance.

to cues. 10 of these trials were conducted for both IRIS and voice. IRIS trials involved a hands-at-rest step to ensure that each event starts independently of the previous. The second trial evaluated the granular control performance between IRIS and voice. Participants were told to set the speaker volume to that of a loud gathering or a pleasant ambient level based on if the proctor said "loud" or "quiet". In total, we collected 690 IRIS and voice trials. Finally, these trials were all collected over video, which we use to assess the speed and efficiency between IRIS and a commercial voice assistant, later.

4.1.3 Qualitative results. We conducted a qualitative analysis of the overall user experience, measured by Mean Opinion Score (MOS), and assessed subjective preferences between IRIS and voice among participants in our study. Participants were given 3 questions:

(1) How was your overall experience using the smart ring? (1: Awful, 5: Amazing)
(2) How difficult was it to use the smart ring? (1: Very difficult, 5: Very easy)
(3) How natural was it to use the smart ring? (1: Very unnatural, 5: Very natural)

Across our participants, the mean opinion scores were 4.22, 3.96, and 4.17, respectively. The distribution of scores across each category is shown in Fig. 10. While these scores indicate a generally positive response, it is difficult to confirm without a direct comparison. To address this, we asked four specific questions regarding user experience between IRIS and a traditional voice assistant.

(1) Between using your voice and the smart ring, which would you prefer to use to toggle a device's state (i.e. turning a light ON or OFF)?
(2) Between using your voice and the smart ring, which would you prefer to use to granularly control a device's output (i.e., a speaker's volume)?
(3) Between using your voice and the smart ring, which would you find more socially acceptable?
(4) Which would you be more inclined to use day-to-day?

As shown in Table. 3, IRIS outperforms voice across all questions within our user study. Out of the participants surveyed, 13 expressed a preference for using IRIS for toggling the device's state, while 9 favored Siri. While the p-value is not statistically significant (0.202), we believe that this is attributed to the fact that our prototype's camera was angled slightly higher than it should have been. This caused some participants to have to point lower than they expected, causing retries. Despite this, more than half of our participants surveyed still preferred IRIS to Siri. In terms of granular control and social acceptability, participants preferred IRIS with p-values of 0.001 and 0.017, respectively. Considering that participants were using a prototype compared to a shipping commercial product, we believe this to be a notable result.

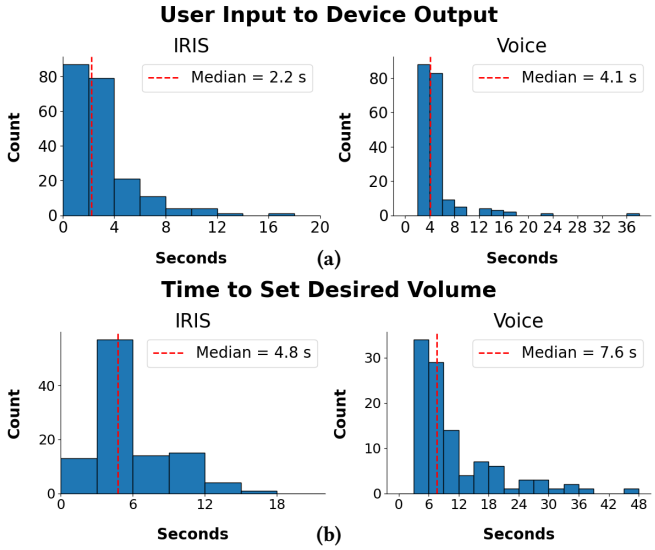

Figure 11: Histogram of User Input to Device Output. (UIDO) (a) and times to set desired volume (b) for IRIS and voice.

Finally, we surveyed participants and asked if they would prefer to use IRIS, Siri, or both for day-to-day use. 11 out of the 23 participants surveyed mentioned that they would use both, and 8 participants preferred IRIS, while 4 favored Siri. A p-value of 0.196 indicates that IRIS is not posed to replace voice assistants outright, but rather coexist alongside them. Participants who preferred to use both systems day-to-day were asked a follow-up question to explain further. Overall, they said that while controlling devices is easier and faster with the ring, there may be times when the ring is not worn, in which case the ubiquity of voice would be advantageous. Another popular comment was that there may be times (i.e., in a conversation) when using voice to control a device would be inappropriate as compared to the ring. A few participants highlighted how much easier it was to control volume with IRIS, and a single participant reported that voice control would be preferable if their hands were occupied and carrying other items.

4.1.4 Quantitative results. We conducted a quantitative analysis of the user input to device output (UIDO) time between IRIS and Siri for toggling the state (on/off) of a smart-light and smart-speaker as well as setting the volume for the smart-speaker. We measured the UIDO time for toggling the device state manually using a stopwatch. We started the timer at the end of each proctor's command issuance and ended it immediately after the light or speaker state changed. For the granular control, the end of the trial was determined by a visual confirmation from the participant that they had set the volume to a level that they deemed appropriate.

The histograms in Fig. 11 show the UIDO time for state toggling and volume control across all participant trials for IRIS and Siri control. Fig. 11(a) shows that most participants were able to control the device state within 0 to 2 seconds using IRIS compared to 2 to 4 seconds when using Siri. Fig. 11(b) shows that for granular volume control a majority of participants were able to set their desired volume within 3 to 6 seconds when using both IRIS and Siri. However, when using Siri there was a wider distribution of times, with a wider positive skew.

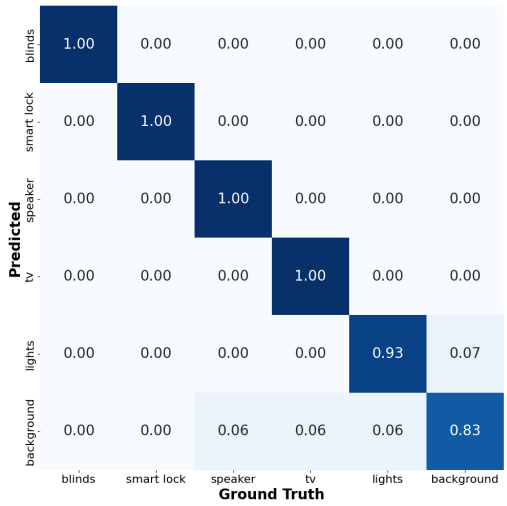

Figure 12: Confusion matrix results across our five supported smart device classes and background. We report a classification accuracy of 95.3% after processing the image through CODA.

For both control scenarios, the median UIDO and time to set desired volume was lower for IRIS (2.2 and 4.7 seconds respectively) than Siri (4.1 and 7.6 seconds respectively). This can be attributed to the fact that it took longer for participants to utter a full voice command than perform the button press and rotation gestures.

Since IRIS is impacted by the pointing accuracy of the user, we also calculated the angular error between the center of the image frame and the center of the bounding box for the intended device. The angular error was calculated by dividing the field-of-view (FOV) of the camera (87 degrees) by the resolution, which was 160 to arrive at an angular-FOV/pixel of 0.54 degrees/pixel. We then multiplied this value by the x and y pixel errors calculated for 2344 photos gathered throughout the duration of the study. The median angular error was 7.6 degrees in the horizontal direction and 23.4 degrees in the y direction. We attribute the larger error in the vertical direction due to the relative position of the button and the camera on IRIS. In the current design, the button is positioned too far forward which required participants to have to rotate the camera towards them for their thumb to reach the button, resulting in the camera being angled higher vertically than intended.

4.2 Vision Pipeline Evaluation

We first assess the performance of YOLO+CODA and DINOV2. These models were evaluated independently as their joint performance primarily reduces runtime by reducing the search space in the DINOV2 (Embedding) database (see Fig. 6). Classification accuracies of our models are independent of one another. We later evaluate the latency of jointly running the models (§4.3.1).

4.2.1 Out-of-the-box object detection. We evaluate our object detection performance across 5 homes, and the 5 classes IRIS supports out-of-the-box (blinds, smart-lock, speaker, tv, lights) along with background (null). The objects were located in a variety of rooms (bedrooms, living rooms, kitchen, and office) and varying lighting conditions and angles. There was no overlap in images between our training and test datasets.

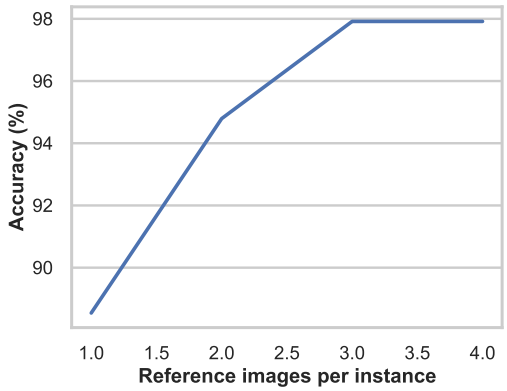

Figure 13: Accuracy of semantic similarity based search as the number of reference images per instance increases. . We report an accuracy of 98% after 3 reference images.

As IRIS's YOLO network is a fine-tuned model on an existing architecture, we validate its performance based on mean average prediction (mAP) values on the hardware dataset's test set. We used the existing YOLOv8 weights and fine-tuned it on both the web-scraped data and in-the-wild data for 75 epochs each. After completing the training against the in-the-wild dataset, the model achieved the following results: 0.793 Precision, 0.784 Recall, 0.81 mAP50, and 0.462 mAP50-95.

While the mAP50-95 score may not necessarily indicate strong overall performance, IRIS is dependent solely on the correct classification of the centered object. This is because if multiple detected objects exist in a frame, IRIS rejects the outlier objects and opts to control the object in the center of the frame. Thus, we computed a confusion matrix to show if the object in the center was classified correctly. Fig. 12 shows the performance of our YOLOv8 model after CODA. Across 86 images, IRIS's YOLO+CODA implementation shows an accuracy of 95.3% for center object classification. With only 3 false negatives and 1 false positive across the test set, we believe that the model performs sufficiently well for in-the-wild applications. We note that with a larger in-the-wild dataset, the performance of this part of our model could improve.

4.2.2 Human-in-the-loop with semantic similarity. For evaluating our instance-based detection method, we collect a user-defined test set of 96 images. These are all collected with the ring and contain 18 unique instances (devices): 2 blinds, 1 door, 4 lighting systems, 2 smart locks, 5 speakers, 2 TVs, and 2 HVACs. Each instance has 3-7 images associated with it taken from different perspectives. We sample one of these images *without replacement* and use it as a query image. The rest of the images are considered references. We then predict the instance associated with the query image using the semantic similarity based search algorithm described in §3.3.2. This prediction is compared against the ground-truth instance this query was sampled from, and marked as a correct or incorrect detection. This process is repeated across all 18 unique instances in the test set. Accuracy is computed over the entire set as the ratio of correct predictions over the total number of queries.

To evaluate how many reference images a user needs to collect, we repeat this process while limiting the number of reference images per instance. This experiment allows us to gain insight by

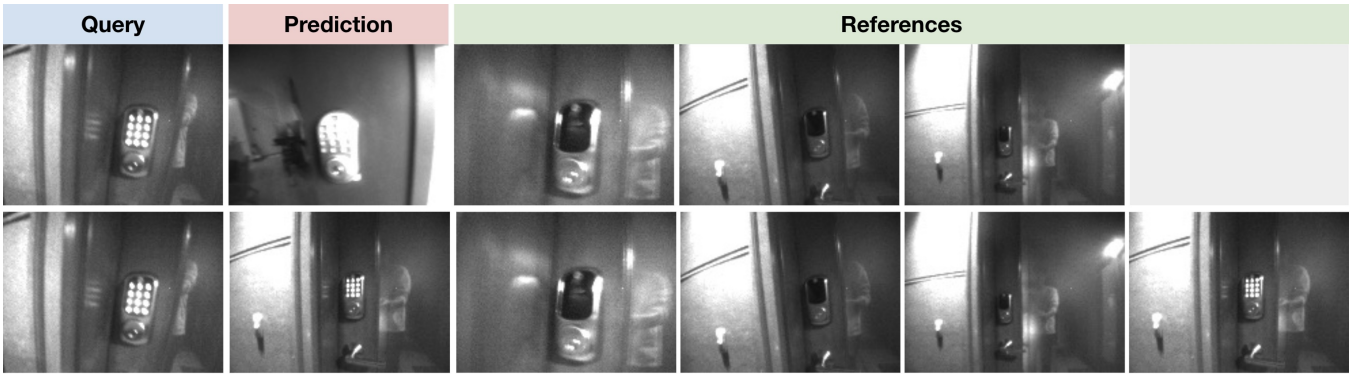

Figure 14: An example failure scenario with semantic similarity search. This primarily occurs due to the lack of sufficiently diverse references. In the top row's inference with 3 reference images, the reference images corresponding to the correct instance do not contain the smartlock of interest in the on-state. So semantic similarity search wrongly picked a different smartlock instance in the on-state. In the bottom row, when we add a reference with the correct smartlock in the on-state as well, we pick the correct smartlock instance.

measuring the accuracy as a function of the number of reference images available per instance. The accuracy obtained from this evaluation is observable in Fig. 13. Our results indicate that 2 reference images are required per instance for 95% accuracy and 3 images reaches an upper bound of 98% accuracy. We note that these reference images are only required for when multiple instances of the same class are within the home and/or a new/unseen IoT device (e.g. HVAC) needs to be detected.

This approach results in very few inaccurate predictions (2%), an example of which are shown in the Fig. 14. Failures with semantic search primarily occur due to a lack of sufficiently diverse references. In the top row example, the model incorrectly predicts a different smart lock as the best match. We can observe that this is caused because none of the references contain the smart-lock in its *on* state, but another reference of a different smart-lock contains an example in its on-state. We observe that when we include a reference with the queried smart-lock in its on-state, the prediction is accurate as shown in the bottom row of Fig. 14.

4.3 System Evaluation

4.3.1 Latency. End-to-end latency in our system is defined as the time elapsed between a participant completing a gesture (e.g. clicking the button) and the corresponding HomeKit command (e.g. toggling lights) being transmitted via an iPhone. This includes multiple components as shown in Table. 4. We start our measurement by first putting IRIS into the IDLE state (camera clock-gated and IMU suspended) as this is the default state of IRIS when not in use. A hardware interrupt triggers upon image transfer completion through the nRF52840's BLE stack. We measured the hardware latency using an oscilloscope, calculating the delta between the button's falling edge and the image completion interrupt pin. This experiment yielded a hardware latency of 293ms, closely aligning with the expected value discussed in §3.2.2.

The other contributors to latency in our system are YOLO and DinoV2. We measured YOLO latency on the iPhone and observed values between 24-28ms. For Dinov2, we first measured the embedding computation time, which was 7.69ms. Next, we evaluated the time required to query the embedding database, noting that this latency scales linearly with the database size. We measured query times across a range of database sizes (4 to 100 embeddings), resulting in values ranging from 9ms to 423ms, respectively. Even in the scenario where IRIS queries 100 embeddings, the total system latency remains at 752ms. These measurements were all performed on an iPhone 13.

Embedding database size	4	50	100
Hardware	293ms	293ms	293ms
YOLOv8	28ms	28ms	28ms
Embedding Generation	8ms	8ms	8ms
Embedding Query	9ms	244ms	423ms
Total latency	338ms	573ms	752ms

Table 4: Latency Breakdown. Query times are across the entire database (DINOV2 only) and are reduced via YOLO+CODA search space reduction. Results are shown in the figure below.

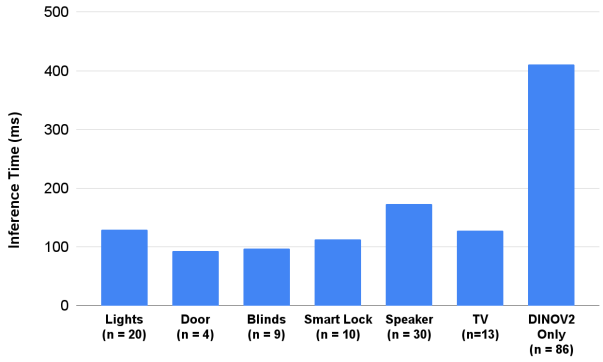

Figure 15: Query times for specific classes against the entire embedding database. With only DINOV2, we report a runtime of 411ms. Utilizing the classification output from YOLO+CODA allows us to reduce the search space to a specific class, reducing latency by hundreds of milliseconds.

To further optimize inference latency, we utilized the YOLO+CODA output to reduce the search space for relevant embeddings that solely align with the classified device. We measure the runtime using this technique across an embedding database size of 86

Component	SLEEP	IDLE	ACTIVE
SoC (ISP1807)	4.23 μW	3.53 mW	19.2 mW
IMU (BMI270)	6.3 μW	6.3 μW	0.76 mW
PMIC (MAX77650)	23.5 μW	23.5 μW	0.148 mW
Camera (HM01B0)	–	–	1.1 mW
Estimated Total	>34.03 μW	>3.76 mW	21.2 mW
Measured Total	86.5 μW	6.63 mW	26.1 mW

Table 5: IRIS hardware power consumption.

Gestures/hr	Battery Life	Battery Life (SLEEP>8 hrs)
10	16.6 hrs	32.9 hrs
30	15.9 hrs	31.5 hrs
60	14.9 hrs	29.5 hrs

Table 6: Battery life (hours) across different gesture rates for the 27 mAh battery on our ring hardware.

images, corresponding to 16 unique device instances (2 blinds, 1 door, 4 lights, 2 smart locks, 5 speakers, and 2 TVs). We show in Fig. 15, that reducing the search space in this way reduces the query runtime on the order of hundreds of milliseconds. This latency optimization can be quite significant for interactive mobile systems. Finally, we note that further latency optimization could have been achieved by utilizing both a dedicated vectorized matrix library and parallelization. However, we leave this to future work.

4.3.2 Power consumption. To calculate the expected battery life during usage, we first connected the battery terminals of IRIS to a power supply that has μA resolution. We measured the current draw of IRIS at 4.2V during each of the three power states (ACTIVE, IDLE, and SLEEP) for at least 25 seconds. For each operating mode, we averaged the current draw and derived the power consumption numbers shown in Table 5.

To calculate the expected battery life of IRIS, we began by assuming an average gesture duration of approximately 3 seconds in ACTIVE mode (26.1mW, 6.23mA). Over the course of a minute, this leaves 57 seconds for which the system is in its IDLE state (6.63mW, 1.58mA). From these measurements, we calculated the average power consumption for N gestures over the course of an hour, giving an average current draw of 1.58mA for a single gesture for one hour. Table 6 shows the expected battery life of IRIS for different numbers of gestures performed per hour. Even at 60 gestures per hour or 1 gesture a minute, IRIS can last 15 hours on a full charge. Finally, we model IRIS's battery life for users who are away from home for up to 8 hours a day. This allows us to enter SLEEP mode (IRIS's lowest power state), as the user is away from their home Wi-Fi, and IRIS need not be constantly waiting for user input. In these use cases, IRIS is able to last beyond 24 hours.

While the IRIS network consumes 774 MIPS during operation, this is exclusive to the ACTIVE state. Since IRIS's use case is episodic, the overall impact on a smartphone's battery life is minimal.

4.3.3 Distance. Our implementation enables us to observe IRIS's detection capability in real-time using an iPhone display. Thus, the effective detection range of IRIS was evaluated for various smart home devices through a visual feedback-based approach. For each device, we started at the minimum distance at which a bounding box was observed and progressively stepped backward. IRIS's detection capability at farther distances was evaluated based on the presence

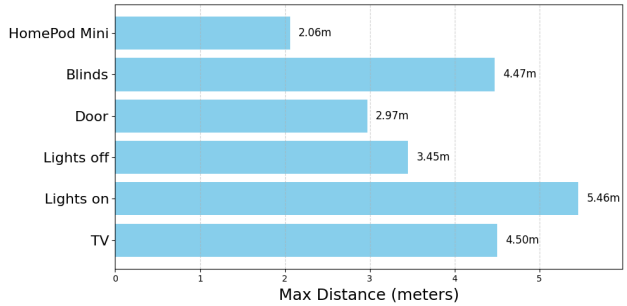

Figure 16: IRIS Distance Performance.

or absence of a bounding box around the target device. The furthest distance at which a bounding box was consistently generated for the device was recorded as the maximum detection range. Full detection range performance is observable in Fig. 16.

This approach offered a practical way to assess IRIS's performance in a real-world setting. Our results aligned well with our expectations as small devices with few features failed first (HomePod Mini). Similarly, larger objects like blinds, televisions, and bright lights could still be picked up by YOLO at farther distances but fail once the object became small with respect to the image dimensions. The techniques described in [17] could potentially be used in future to improve performance further for small object detection.

4.3.4 Low-light conditions. We evaluate IRIS's object detection performance in low-light environments with a series of controlled experiments. We assessed IRIS's ability to detect smart home objects across a range of illuminance levels (lux). A controllable lighting system capable of adjusting brightness from 0 to 100% was used to create a controlled testing environment during nighttime hours. We systematically decreased light intensity within the room until IRIS failed to detect the target device. The minimum lux at which a bounding box was consistently generated for each device was recorded as the minimum lux required for proper functionality.

Across the devices tested, we observed that IRIS detection failures began when the room's illuminance level (lux) approached or fell below 1. This indicates successful operation in quite dark environments. These results are largely due to two reasons. First, our camera is configured for auto-exposure, reducing its shutter speed in bright scenarios, while increasing its shutter speed in dark scenarios. We note that the camera's maximum exposure time is 1/160th of a second, or 6.3ms. This is typically the upper bound for modern DSLR cameras, without in-body image stabilization, to be able to take images without motion blur from the user's hand. Second, YOLO's feature extraction capabilities remain relatively strong until there is an absence of light. Overall, IRIS is able to maintain some level of detection for the target devices even in extremely low-light conditions. Passing and failing examples from this experiment are shown in Fig. 17.

5 LIMITATIONS AND DISCUSSION

Our work with IRIS lays the groundwork for several exciting future directions, which stem from the limitations of our current implementation. One area of exploration involves expanding the gesture set to incorporate more intuitive interactions or additional utility. The results from the ring gesture elicitation study [13] show end

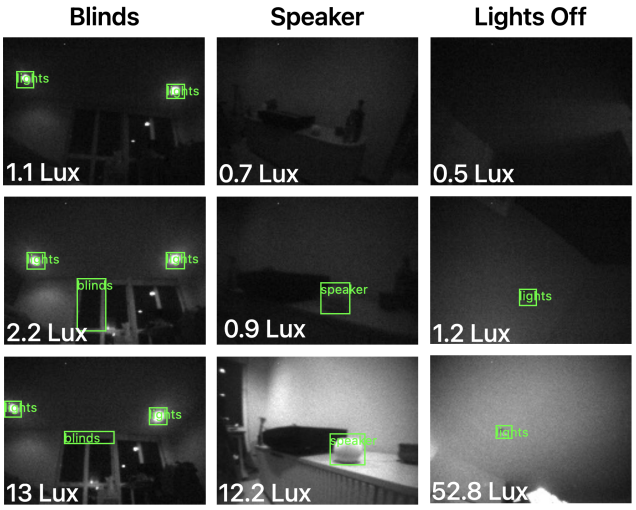

Figure 17: Low-light testing examples. Rows ordered in increasing illuminance. 50% brightness set by our lighting system is approximately 12 lux, 20%-2 lux, and 10%-1 lux. Failures for speaker and lights occurred when our lighting system was turned to 0%, while blinds failed at 10%. All examples were collected at night.

users' preferences for an extensive referent gesture set performed with a ring to control various devices within a smart home. Many of these controls can be mapped to IRIS's simple gesture set, but additional gestures could enable IRIS to scale for devices that offer more functionality. For instance, a press-hold-and-drag gesture could enable users to precisely control the stopping point for blinds. Similarly, sliding or flicking vertically and horizontally could be used to modulate light brightness or speaker volume, offering an alternative to rotation akin to a physical dial. Adding redundancy through additional gestures would be beneficial for users to pick and choose gestures that feel most natural (i.e. IRIS could offer several gestures for controlling volume). Finally, GestuRING [44] presents hundreds of potential gestures for wearable rings, and in-air or on-body input could be utilized for controlling wearable devices, like headphones.

Detecting small objects across far distances (Fig. 16) and/or unseen devices remains a challenge for our current implementation. In IRIS, users could take reference images of a desired device from farther distances, and the system would learn to utilize information from the scene to control the object. For unseen classes in our YOLO dataset, users could potentially take reference images of the unseen device and associate them with the device's corresponding UUID. While Fig. 7c shows the technically feasible of this for HVACs with our existing system, we did not test this extensively and leave this exploration to future work.

Expanding IRIS's hardware presents another promising avenue for future research. Integrating a capacitive touch surface in place of, or alongside, the current button could enable users to navigate digital screens when pointing IRIS at the device. This could enable functionalities like navigating a television menu. Additionally, swiping across the touch surface could provide another alternative to rotations for scenarios requiring granular control. Finally, integrating a digital microphone could lead to a multimodal interface, and provide users the additional option of voice input.

Running the neural network on the mobile phone affects its battery life. Offloading inference from the mobile device to the smart home hub is another promising avenue for improvement. This approach would benefit battery life by minimizing the computational demands placed on the user's phone. Alternatively, embedding the inference on the ring could be a more privacy-preserving approach, alleviating the need to stream images wirelessly. With the recent advances in accelerator hardware, this could be a possibility in the near future.

Since IRIS has limited use cases outside the home, incentivizing users to wear the ring all day is of notable importance. Integrating IRIS's features with existing health-tracking smart rings, such as Oura, Ultrahuman, or the Samsung Galaxy Ring [22, 29, 42], would provide users a reason to wear the ring all day. IRIS would then augment existing smart rings by providing smart home control in addition to tracking daily health metrics. Another application would be to control on-the-go accessories like Bluetooth speakers and headphones. An accessibility application of IRIS could be for individuals experiencing speech difficulties due to conditions like stuttering, apraxia, or dysarthria. IRIS could offer a valuable alternative to the conventional voice assistant for the speech-impaired population. Exploring this however is not in the scope of this paper.

6 CONCLUSION

We presented IRIS, the first wireless ring form-factor system for vision-based smart home interaction. To achieve this, we made multiple technical contributions, including a SWaP-optimized wireless hardware design that integrates a camera within a low-power, wireless ring form factor. Additionally, we achieved instance-level classification that leverages contextual scene semantics. We do this through a combination of YOLO for object detection and CODA for centered object selection. This significantly reduced the search space for the DINOV2 model, optimizing for runtime latency while not sacrificing instance classification performance. Finally, system-level optimizations ensured real-time responsiveness and extended battery life to 16-24 hours. Evaluations from our user study and experiments demonstrate IRIS's effectiveness and user preference over traditional voice control interfaces. We believe that this work represents an important step that can influence the landscape of ring-based human-computer interaction.

7 AUTHOR CONTRIBUTIONS

Manuscript Preparation: MK and SG led; AG and BV led specific sections; YL and EG contributed. **System conceptualization:** MK and SG conceptualized the system. **Hardware and Firmware Design and Development:** MK led design and architecture; MK and AG jointly contributed to development. **Neural Network Design and Development:** MK and BV led design and architecture; MK led YOLO development; BV led DinoV2 development and EG contributed. **Data Collection:** MK, AG, YL, EG, and AB jointly contributed. **Mobile Application Development:** MK and YL led; AG and EG contributed. **Interaction Design:** MK led overall design. **User Study Design and Analysis:** MK and AG jointly led and contributed. **System Evaluation:** MK led overall evaluation; AG led analysis from specific sections.

ACKNOWLEDGMENTS

The researchers were partly supported by the Moore Inventor Fellow award #10617, NSF Graduate fellowship and a Thomas J. Cable Endowed Professorship. This work was facilitated through the use of computational, storage, and networking infrastructure provided by the HYAK Consortium at the University of Washington.

REFERENCES

[1] Amr Alanwar, Moustafa Alzantot, Bo-Jhang Ho, Paul Martin, and Mani Srivastava. 2016. SeleCon: Scalable IoT Device Selection and Control Using Hand Gestures. In *Proceedings of the 10th ACM Conference on Embedded Systems for Energy-Efficient Buildings*, Vol. 2017. IoTDI 2017 (2017), Log Angeles, CA, USA, 107–114. https://doi.org/10.1145/3054977.3054981 pmid:29683151

[2] Amazon. 2024. *Amazon Alexa Voice AI | Alexa Developer Official Site.* Amazon Alexa. https://developer.amazon.com/en-US/alexa

[3] Apple. 2024. HomePod. https://www.apple.com/homepod/.

[4] Roger Boldu, Alexandru Dancu, Denys J.C. Matthies, Thisum Buddhika, Shamane Siriwardhana, and Suranga Nanayakkara. 2018. FingerReader2.0: Designing and Evaluating a Wearable Finger-Worn Camera to Assist People with Visual Impairments while Shopping. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3, Article 94 (sep 2018), 19 pages. https://doi.org/10.1145/3264904

[5] Liwei Chan, Yi-Ling Chen, Chi-Hao Hsieh, Rong-Hao Liang, and Bing-Yu Chen. 2015. CyclopsRing: Enabling Whole-Hand and Context-Aware Interactions Through a Fisheye Ring. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 549–556. https://doi.org/10.1145/2807442.2807450

[6] Kaifei Chen, Jonathan Fürst, John Kolb, Hyung-Sin Kim, Xin Jin, David E. Culler, and Randy H. Katz. 2018. SnapLink: Fast and Accurate Vision-Based Appliance Control in Large Commercial Buildings. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 129 (jan 2018), 27 pages. https://doi.org/10.1145/3161173

[7] Rajkumar Darbar, Mainak Choudhury, and Vikalp Mullick. 2019. RingIoT: A Smart Ring Controlling Things in Physical Spaces. , 2–9 pages.

[8] Adrian A. de Freitas, Michael Nebeling, Xiang 'Anthony' Chen, Junrui Yang, Akshaye Shreenithi Kirupa Karthikeyan Ranithangam, and Anind K. Dey. 2016. Snap-To-It: A User-Inspired Platform for Opportunistic Device Interactions. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 5909–5920. https://doi.org/10.1145/2858036.2858177

[9] Retail Dive. 2024. 27% Increase in Smart Home Adoption Since 2020: YouGov Report. https://www2.deloitte.com/us/en/insights/industry/telecommunications/connectivity-mobile-trends-survey/2023/smart-home-industry-adoption-trend.html

[10] Daily Dot. 2022. Ring Zero: The Smart Logbar That Could Change How We Interact With Tech. https://www.dailydot.com/debug/ring-zero-smart-logbar-sxsw/

[11] Yasuhiro Endo, Zheng Wang, J Bradley Chen, and Margo I Seltzer. 1996. Using latency to evaluate interactive system performance. *ACM SIGOPS Operating Systems Review* 30, si (1996), 185–199.

[12] M. Fiala. 2005. ARTag, a fiducial marker system using digital techniques. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, Vol. 2. IEEE, San Diego, CA, USA, 590–596 vol. 2. https://doi.org/10.1109/CVPR.2005.74

[13] Bogdan-Florin Gheran, Jean Vanderdonckt, and Radu-Daniel Vatavu. 2018. Gestures for Smart Rings: Empirical Results, Insights, and Design Implications. In *Proceedings of the 2018 Designing Interactive Systems Conference* (Hong Kong, China) *(DIS '18)*. Association for Computing Machinery, New York, NY, USA, 623–635. https://doi.org/10.1145/3196709.3196741

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, Redmond, WA, USA, 770–778.

[15] Vikram Iyer, Ali Najafi, Johannes James, Sawyer Fuller, and Shyamnath Gollakota. 2020. Wireless steerable vision for live insects and insect-scale robots. *Science robotics* 5, 44 (2020), eabb0839. https://doi.org/10.1126/scirobotics.abb0839

[16] Shilpi Jain, Sriparna Basu, Arghya Ray, and Ronnie Das. 2023. Impact of irritation and negative emotions on the performance of voice assistants: Netting dissatisfied customers' perspectives. *International Journal of Information Management* 72 (2023), 102662. https://doi.org/10.1016/j.ijinfomgt.2023.102662

[17] Shu-Jun Ji, Qing-Hua Ling, and Fei Han. 2023. An Improved Algorithm for Small Object Detection Based on YOLO v4 and Multi-scale Contextual Information. *Computers and Electrical Engineering* 105 (2023), 108490. https://doi.org/10.1016/j.compeleceng.2022.108490

[18] Lei Jing, Zixue Cheng, Yinghui Zhou, Junbo Wang, and Tongjun Huang. 2013. Magic Ring: a self-contained gesture input device on finger. In *Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia* (Luleå, Sweden) *(MUM '13)*. Association for Computing Machinery, New York, NY, USA, Article 39, 4 pages. https://doi.org/10.1145/2541831.2541875

[19] Runchang Kang, Anhong Guo, Gierad Laput, Yang Li, and Xiang 'Anthony' Chen. 2019. Minuet: Multimodal Interaction with an Internet of Things. In *Symposium on Spatial User Interaction* (New Orleans, LA, USA) *(SUI '19)*. Association for Computing Machinery, New York, NY, USA, Article 2, 10 pages. https://doi.org/10.1145/3357251.3357581

[20] Xiaoyu Li, Shuqin Zeng, Yanwei Zhang, Ping Wan, and Jun Wang. 2012. Analysis and processing of pixel binning for color image sensor. *EURASIP Journal on Advances in Signal Processing* 2012, 1 (2012), 81. https://doi.org/10.1186/1687-6180-2012-81

[21] Brian D. Mayton, Nan Zhao, Matt Aldrich, Nicholas Gillian, and Joseph A. Paradiso. 2013. WristQue: A personal sensor wristband. In *2013 IEEE International Conference on Body Sensor Networks* (Cambridge, MA, USA). IEEE, Cambridge, MA, USA, 1–6. https://doi.org/10.1109/BSN.2013.6575483

[22] Jay McGregor. 2024. *Samsung Galaxy Ring: Release Date, Price, Design, Features.* Forbes. https://www.forbes.com/sites/jaymcgregor/2024/03/11/samsung-galaxy-ring-release-date-price-design-features/?sh=308cc8f513bf

[23] Kento Miyaoku, Anthony Tang, and Sidney Fels. 2007. C-Band: A Flexible Ring Tag System for Camera-Based User Interface. In *Virtual Reality*, Randall Shumaker (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 320–328.

[24] Suranga Nanayakkara, Roy Shilkrot, Kian Peen Yeo, and Pattie Maes. 2013. EyeRing: a finger-worn input device for seamless interactions with our surroundings. In *Proceedings of the 4th Augmented Human International Conference* (Stuttgart, Germany) *(AH '13)*. Association for Computing Machinery, New York, NY, USA, 13–20. https://doi.org/10.1145/2459236.2459240

[25] Jared Newman. 2022. The Smart Home Is Flailing as a Concept Because It Sucks. https://www.fastcompany.com/90660570/the-smart-home-is-flailing-as-a-concept-because-it-sucks

[26] Jakob Nielsen. 1993. Smart Home Statistics. https://www.nngroup.com/articles/response-times-3-important-limits/.

[27] Oberlo. 2024. Smart Home Statistics. https://www.oberlo.com/statistics/smart-home-market.

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. 2024. DINOv2: Learning Robust Visual Features without Supervision. arXiv:2304.07193 [cs.CV]

[29] Oura. 2024. Oura Ring. https://ouraring.com/. Accessed: March 31, 2024.

[30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020 [cs.CV]

[31] 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 (CVPR)*. IEEE, Seattle, WA, USA, 779–788. https://doi.org/10.1109/CVPR.2016.91

[32] Leon Reicherts, Yvonne Rogers, Licia Capra, Ethan Wood, Tu Dinh Duong, and Neil Sebire. 2022. It's Good to Talk: A Comparison of Using Voice Versus Screen-Based Interactions for Agent-Assisted Tasks. *ACM Trans. Comput.-Hum. Interact.* 29, 3, Article 25 (jan 2022), 41 pages. https://doi.org/10.1145/3484221

[33] Ana Rodrigues, Rita Santos, Jorge Abreu, Pedro Beça, Pedro Almeida, and Sílvia Fernandes. 2019. Analyzing the performance of ASR systems: The effects of noise, distance to the device, age and gender. In *Proceedings of the XX International Conference on Human Computer Interaction* (Donostia, Gipuzkoa, Spain) *(Interacción '19)*. Association for Computing Machinery, New York, NY, USA, Article 8, 8 pages. https://doi.org/10.1145/3335595.3335635

[34] Mia Sapienza. 2022. Are You Still Relying on Your Phone to Control Your Home? https://www.brilliant.tech/blogs/news/are-you-still-relying-on-your-phone-to-control-your-home

[35] Nordic Semiconductor. 2022. *Things You Should Know About Bluetooth Range.* Nordic Semiconductor. https://blog.nordicsemi.com/getconnected/things-you-should-know-about-bluetooth-range

[36] Roy Shilkrot, Jochen Huber, Wong Meng Ee, Pattie Maes, and Suranga Chandima Nanayakkara. 2015. FingerReader: A Wearable Device to Explore Printed Text on the Go. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2363–2372. https://doi.org/10.1145/2702123.2702421

[37] Lee Stearns, Uran Oh, Leah Findlater, and Jon E. Froehlich. 2018. TouchCam: Realtime Recognition of Location-Specific On-Body Gestures to Support Users with Visual Impairments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 4, Article 164 (jan 2018), 23 pages. https://doi.org/10.1145/3161416

[38] George Stetten, Roberta Klatzky, Brock Nichol, John Galeotti, Kenneth Rockot, Kimberly Zawrotny, David Weiser, Nathan Sendgikoski, and Samantha Horvath.

2007. Fingersight: Fingertip Visual Haptic Sensing and Control. In *2007 IEEE International Workshop on Haptic, Audio and Visual Environments and Games*. IEEE, Ottawa, ON, Canada, 80–83. https://doi.org/10.1109/HAVE.2007.4371592

[39] Google Store. 2024. *How to Set Up a Smart Home*. Google. Accessed: March 31, 2024.

[40] Mingxing Tan and Quoc Le. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 97)*, Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, Mountain View, CA, USA, 6105–6114. https://proceedings.mlr.press/v97/tan19a.html

[41] Punch Through. 2022. *Maximizing BLE Throughput on iOS and Android*. PunchThrough. https://punchthrough.com/maximizing-ble-throughput-on-ios-and-android/#:~:text=It%20is%20important%20to%20know,per%20connection%20event%20in%20Android

[42] Ultrahuman. 2024. Ultrahuman. https://www.ultrahuman.com//. Accessed: March 31, 2024.

[43] Ultralytics. 2024. YOLOv8. https://github.com/ultralytics/yolov8. Accessed: 2024-03-31.

[44] Radu-Daniel Vatavu and Laura-Bianca Bilius. 2021. GestuRING: A Web-based Tool for Designing Gesture Input with Rings, Ring-Like, and Ring-Ready Devices. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 710–723. https://doi.org/10.1145/3472749.3474780

[45] Bandhav Veluri, Collin Pernu, Ali Saffari, Joshua Smith, Michael Taylor, and Shyamnath Gollakota. 2023. *NeuriCam: Key-Frame Video Super-Resolution and Colorization for IoT Cameras*. Association for Computing Machinery, New York, NY, USA, Chapter 25, 1–17. https://doi.org/10.1145/3570361.3592523

[46] P. K.A. Wollner, P. M. Langdon, T. Goldhaber, I. M. Hosking, A. Mieczakowski, and P. J. Clarkson. 2012. Evaluation of setup procedures on mobile devices based on users' initial experience. In *NordDesign 2012 - Proceedings of the 9th NordDesign Conference*, Poul Kyvsgaard Hansen, John Rasmussen, Kaj A. Jorgensen, and Christian Tollestrup (Eds.). Center for Industrial Production, Aalborg University and Design Society, University of Strathclyde, Aalborg, Denmark, 1–8. 9th NordDesign Conference, NordDesign 2012 ; Conference date: 22-08-2012 Through 24-08-2012.

[47] Yoonjong Yoo, Jaehyun Im, and Joonki Paik. 2015. Low-Light Image Enhancement Using Adaptive Digital Pixel Binning. *Sensors* 15, 7 (2015), 14917–14931. https://doi.org/10.3390/s150714917

[48] Sang Ho Yoon, Yunbo Zhang, Ke Huo, and Karthik Ramani. 2016. TRing: Instant and Customizable Interactions with Objects Using an Embedded Magnet and a Finger-Worn Device. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 169–181. https://doi.org/10.1145/2984511.2984529

[49] Tengxiang Zhang, Xin Zeng, Yinshuai Zhang, Ke Sun, Yuntao Wang, and Yiqiang Chen. 2020. ThermalRing: Gesture and Tag Inputs Enabled by a Thermal Imaging Smart Ring. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376323