Semantic Scene Graphs for Creating a Localization-Ready Internet of Things

Jan Kolberg, Michael Pabst, Verena Biener, Shohei Mori, and Dieter Schmalstieg

Abstract—Controlling devices connected to the Internet of Things often requires juggling multiple smartphone apps or physical remote controls, creating a fragmented user experience. Augmented Reality (AR) can afford superior control by automatically presenting virtual user interfaces that are spatially aligned with networked devices. However, before such user interfaces can be delivered, physical devices must be localized in the environment. This paper introduces LORIOT (LOcalization-Ready Internet Of Things), a novel end-to-end system that uses a semantic scene graph and a large language model to map the identities of the networked devices to physical objects, given a pre-filtered set of IoT-device candidate nodes. A declarative UI specification enables *automatic generation of device control panels* for AR and non-AR clients. We evaluate the mapping component on a controlled synthetic-room benchmark of 100 randomly generated rooms. Using device network metadata alone, we achieve a baseline macro-averaged F1 score of 0.80 for digital→physical associations. When device metadata is enriched with physical attributes (mounting location, materials, color, and size), performance improves to 0.88. Moreover, we evaluate the benefit of spatially registered AR control in a within-subject user study ($N{=}20$), comparing in-situ AR panels against conventional non-AR control with smartphone apps or physical remote controls. AR yields significantly faster task completion, lower mental demand, and higher usability.

Index Terms—Internet of Things, Augmented Reality, Device Localization, Semantic Scene Graphs, Human-Computer Interaction, User Interface Generation.

1 Introduction

Controlling devices connected to the Internet of Things (IoT) can be frustrating, if users are forced to juggle multiple manufacturer-specific smartphone applications or search for dedicated physical remotes. Augmented Reality (AR) provides more immediate control by presenting virtual user interfaces spatially registered with networked devices. However, before such AR user interfaces can be delivered, devices must be located in the environment. The system must know the precise physical pose (six degrees of freedom for combined position and orientation) of each device. Localization remains a critical challenge for AR-style interfaces [28], [36], [43].

Existing solutions only partially address this problem. Some devices can make themselves known with radio signals [18], [29], or they can carry QR codes or RFID tags for detection on demand [4], [13], [21], [33], [37]. Experimental mapping applications can detect blinking lamps or audio signals from loudspeakers [38]. If all else fails, stationary devices can be manually entered on a map. All of these methods are cumbersome and do not scale well to large environments or daily use. Furthermore, even if the location of a device is known, connecting the physical device to its digital identity is still far from trivial. Remote operation over the network is often vendor-specific and requires a fair amount of engineering effort to establish universal remote control over all networked devices.

In this paper, we introduce LORIOT, the LOcalization-Ready Internet Of Things. The purpose of LORIOT is to automate the localization of networked devices and establish a mapping between physical devices and their digital identity. We achieve this by first using an RGB-D scan of the environment to construct a semantic scene graph (SSG). The SSG is composed of objects with both geometry and semantic attributes. We then use a large language model (LLM) for device mapping. The LLM reasons over the SSG together with network-side device metadata to associate each physical object observation with the correct digital device identity. Finally, we use the LLM one more time to derive a declarative user interface (UI) description from the device metadata. This description is combined with the mapped physical poses to automatically generate a spatially registered remote control interface in AR. In our current prototype, we manually pre-filter the SSG to the subset of nodes corresponding to IoT devices before running the LLM-based device mapping.

In summary, our contributions are the following:

- An end-to-end system makes IoT deployments localization-ready by constructing a semantic scene graph from an RGB-D scan, maps network identities to physical objects via LLM-based reasoning, and enables in-situ AR control through spatially registered interfaces.
- An LLM-based device mapping method uses semantic scene graph descriptions together with network-side device metadata to produce confidence-ranked device–object associations and to support resolving remaining ambiguities.
- A declarative user interface specification and generation pipeline allows an LLM to synthesize device-agnostic control panels from device metadata and instantiate them as spatially registered AR interfaces (or

- *J. Kolberg, M. Pabst, V. Biener, S. Mori, and D. Schmalstieg are with VISUS, University of Stuttgart, Germany. Email {first.last}@visus.uni-stuttgart.de.*
- *D. Schmalstieg is also with Graz University of Technology, Austria.*
- *J. Kolberg is the corresponding author.*

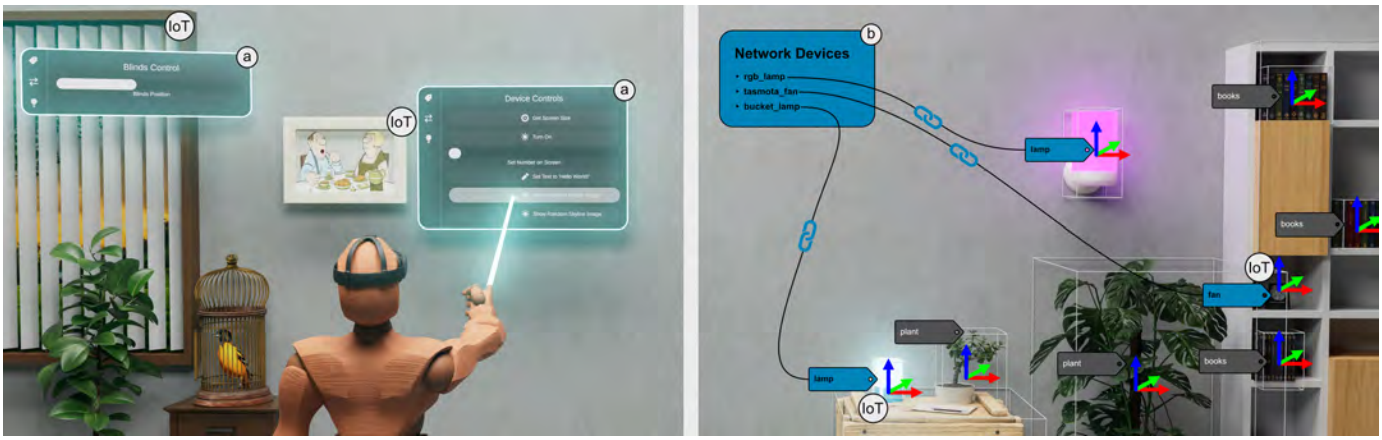

Fig. 1. Conceptual illustration of Device localization and control in LORIOT. **Left: LLM-generated AR control panels:** Given a device's network metadata (available commands and parameters), an LLM produces a declarative JSON UI description (widgets + command bindings), which the AR client instantiates as an in-situ panel (a) next to the physical device (IoT). Additional examples of generated UIs for different devices are shown in Figures 4 and 5. **Right: Mapping between physical object and network identity:** Panel (b) visualizes the system's predicted associations between network-side device identities and SSG object nodes. When multiple identical device instances exist, LORIOT resolves remaining ambiguities via a separate situated confirmation/correction step (Sec. 3.1).

conventional 2D interfaces).

Two evaluations demonstrate our achievements: First, a controlled synthetic benchmark measures mapping success. Second, a within-subject user study (N=20) compares AR vs. non-AR control of IoT devices, showing faster performance and greater subjective usability for AR.

The purpose of LORIOT is to demonstrate how AR can be used to interact with smart environments at scale. By supporting the critical steps of device mapping and UI generation, it provides a key enabling technology for universal remote control in AR. Specifically, we show how an LLM can help with organizing one's growing menagerie of devices. Our workflow is friendly to non-technology-savvy users and drastically reduces the time required for setting up environments with many networked devices.

2 Related Work

Our work brings together multiple topics from the domains of AR and IoT. We organize the related work into three groups: device control with AR, device localization, and construction of user interfaces for IoT.

2.1 Controlling devices in AR

Integration of device control in AR environments has been explored with various technical approaches and interaction styles [1], [4], [12], [13]. Prior work usually focuses on relatively simple device control scenarios, often limited to a single device type or a specific use case [4], [13], [20], [40]. Current consumer-grade systems, such as the Apple Vision Pro, already demonstrate compelling smart home control in AR[1]. However, the setup for these applications is manual and often cumbersome.

Beyond device localization, prior work on AR IoT control mainly focuses on interaction techniques for issuing commands (e.g., gesture-based input) and on communication layers that connect AR clients to heterogeneous IoT protocols.

Hand gesture recognition has emerged as another intuitive method for remote control in AR [40]. Jo and Kim [20] implemented hand gesture-based control for smart lighting systems, allowing users to perform gestures such as pointing and pinching to manipulate device states.

The communication layer between AR applications and networked devices has been addressed through various implementations of protocols. Fleck et al. [11], [12] developed a Unity tool that integrates MQTT communication, allowing real-time bidirectional communication between AR applications and IoT networks. Dias et al. [9] present another framework for device management via MQTT in AR environments.

While these methods demonstrate various ways of interaction with devices in AR, they often presuppose that the pose of the device is already known or rely on manual setup, such as placing markers [13], [19]. Our work addresses this fundamental limitation by providing an integrated localization pipeline. By leveraging an SSG, LORIOT removes the need for manual placement or on-device markers, enabling nearly instant control over a large number of heterogeneous devices, a critical step for going beyond a reality where only one device is controlled at a time.

2.2 Device localization

Prior work on device localization for IoT systems has explored various methods to bridge the gap between a device's network identity and its physical position. Radio-based techniques utilize wireless signal properties such as ZigBee signal strength [29] or UWB radio [18] to estimate device locations; however, this approach lacks precision and is only effective for a small number of devices that are sufficiently distant from one another. A widely used class of solutions relies on visual fiducials (e.g., QR-code-like tags) attached on or near devices. By detecting the marker, an AR client can directly recover device identity and pose for

1. https://support.apple.com/en-gb/guide/apple-vision-pro/dev26039f68/visionos

spatially registering overlays [3], [19], [21], [26], [41]. While effective, these techniques shift the burden to deployment and maintenance: Users must attach markers, keep them visible, and tolerate potential occlusions or aesthetic concerns, which limits scalability in heterogeneous real-world environments.

An alternative approach relies on physically observing state changes. Schenkluhn et al. [38] propose to identify devices by sequentially activating their output to allow identification of their physical location. This strategy is effective for devices with easily perceptible state changes, like lamps, but leaves out input-only devices (e.g., buttons) or devices with less obvious state changes, such as height-adjustable desks or motorized window blinds.

All of these methods are effective, but typically limited in scope, dependent on manual intervention, or not easily scalable to large, heterogeneous environments. To overcome these limitations, we propose to rely on semantics. We leverage an LLM to reason about the information contained in an SSG and the network metadata to perform device localization in a robust and generalizable way. LORIOT maps a wide variety of devices without relying on auxiliary channels such as radio signal strength, observations of physical markers, or observable state changes. Instead, our work builds on recent advances in 3D scene understanding [8], [23], [32]. The technology underlying SSG is rapidly advancing because of its importance for robotics and artificial intelligence [2], [15], [22], [35].

2.3 User interfaces for the Internet of Things

Interoperability is a continuing concern in the IoT community. Several universal IoT control systems have been developed, such as the W3C Web of Things and OCF IoTivity, which use a JSON schema to control devices from different manufacturers [27], [42]. Separate efforts have focused on the generation of user interfaces, ranging from prefabricated components to dynamic user interfaces generated by an LLM [6], [7], [14], [17], [25].

A common approach is to map structured schemas or declarative specifications to interactive forms or dashboards, which can reduce manual UI engineering effort and support UI adaptation as device capabilities evolve [7], [14]. In XR settings, Espinal et al. [10] propose a runtime that interprets JSON-based descriptions to visualize and interact with IoT device data on head-mounted displays. Complementary authoring approaches aim to make such interfaces and policies editable by users, e.g., via interactive tools for creating context-aware behavior in XR [30].

Beyond schema-driven generation, recent work explores using large language models to generate (or assist in generating) UI structures directly from natural-language intent or task descriptions [17]. Because LLM outputs can be generic or misaligned with user preferences, preference libraries have been proposed to guide widget selection toward more usable designs [25]. Research on generative and malleable user interfaces further highlights the value of keeping an explicit intermediate representation so that end-users can iteratively refine generated interfaces instead of receiving one-off code [6].

While prior work has addressed universal control and dynamic user interface generation as separate challenges, our work unifies these concepts. We introduce a single, cohesive JSON specification that serves a dual purpose: It provides a manufacturer-independent abstraction for device control and simultaneously contains the semantic information necessary for an LLM to automatically generate corresponding interactive user interfaces in AR. By grounding UI generation in a compact declarative representation, our approach supports deploying consistent generated interfaces across different clients (AR and smartphone) while keeping back-end control and routing unified.

3 System overview

We run LORIOT as a distributed system (Figure 2), which consists of three main components communicating over a local Wifi network: the device server, the IoT devices, and two types of clients: the AR client and the smartphone client (the latter is used for comparison with AR).

The server contains most of the system logic. It runs on a popular software stack consisting of Node-RED[2] and Mosquitto[3]. Node-RED is an automation framework built with NodeJS[4]. The server formats its data resources as JSON and persistently stores them in CouchDB[5]. Mosqitto is a broker for MQTT messages. MQTT[6] (Message Queuing Telemetry Transport) is a lightweight messaging protocol widely used in IoT networks for publish-subscribe communication.

First, the server (using an NVIDIA RTX 4090 GPU) creates the SSG. At runtime, the server invokes an LLM specialized for logical reasoning to compute the device mapping and generate the user interfaces for the devices. As LLM, we use GPT5.4, which runs in the cloud and is transparently invoked by the server using the OpenAI remote call interface. Second, the device mapping (Section 3.1) is determined. Third, the user interfaces (Section 3.2) are generated. We also prepare the spatial registration of the user interfaces (Section 3.4).

Furthermore, the server routes all commands triggered by the user to the devices. The MQTT protocol (Figure 3) uses a publish-subscribe architecture with two channels, one to send messages to devices and one to receive messages from devices. Timestamps are used to uniquely identify messages, making it easy to orchestrate request-reply sequences.

The routing ability of our server is conceptually similar to the function of home automation appliances such as Matter[7], which can talk to devices using a variety of network protocols and standards. However, our server differs from existing home automation appliances in two important ways. First, it accepts control input from AR clients. Second, our server derives its routing tables using an LLM instead of relying on user input on the console.

As AR client, we use an application built with Unity running on a Meta Quest 3 headset. The client connects to

2. https://nodered.org
3. https://mosquitto.org
4. https://nodejs.org
5. https://couchdb.apache.org/
6. https://mqtt.org
7. https://project-chip.github.io/connectedhomeip-doc/

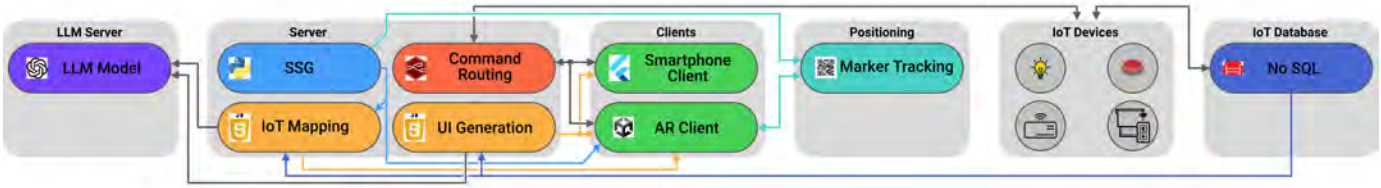

Fig. 2. Network layout and client/server architecture of LORIOT—the server is responsible for the creation of the device mapping and the user interfaces. Multiple clients can connect to the server and receive user interfaces that enable the user to interact with the chosen devices.

the server to fetch the automatically generated user interfaces to remotely control the IoT devices. The smartphone client displays the same user interface in an Android app, but purely in 2D without any spatial registration or AR view.

We assume that all IoT devices are connected and configured in the local WiFi network. Improving the initial setup of the devices themselves is beyond the scope of our work.

3.1 Device mapping

To perform device localization and matching, we use a SSG generated using a modified version of *ConceptGraphs* [15]. ConceptGraphs is a state-of-the-art framework for SSG generation, which accepts RGB-D stream as input and generates the nodes $\mathcal{N}_S$ and edges $\mathcal{E}_S$ of an SSG, as well as rudimentary 3D geometry for each detected object. Each original node contains a 3D bounding box, an open-vocabulary language embedding, and a detailed object caption. To improve the subsequent mapping process, we extended the feature extraction of ConceptGraphs to include additional object describing attributes, such as object color, to assist the reasoning process in disambiguating similar objects (Listing 1).

Listing 1. Sample Node Data

```
{
    obj_id: 219,
    object_tag: lamp,
    object_caption: A small, modern black desk lamp
        with a conical or fabric shade, round base,
        and adjustable arm, positioned on a desk.,
    bbox_extent: [0.39, 0.17, 0.16],
    bbox_center: [-1.95, 0.92, -0.26],
    bbox_volume: 0.01,
    object_tags: [
        small, modern, black, desk lamp,
        conical shade, fabric shade, round base,
        adjustable arm // NEW
    ],
    bbox_rot: [-0.75, 0.62, 0.22, -2.88],
    color: [0.28, 0.25, 0.22] // NEW
}
```

This enhanced SSG contains pose information for all available devices. Following the automatic SSG generation step, we perform a manual SSG refinement step to further improve and clean up the graph. This step is necessary because the SSG can contain inaccurate information, including duplicate object detections and incorrect labels. However, manually labeling large-scale environments containing thousands of objects without automation is time-consuming and exhausting. Therefore, our two-step pipeline reduces the workload over a purely manual process. The description of physical devices is collected in a subset of nodes $\mathcal{N} \subseteq \mathcal{N}_S$. In our current prototype and evaluations, we obtain $\mathcal{N}$ by manually selecting the SSG nodes that correspond to IoT devices (i.e., connected devices vs. visually similar non-connected objects). The LLM-based localization then maps network identities to this candidate set $\mathcal{N}$ as described next. Automating this pre-filtering step is left for future work.

Our next goal is to use the LLM to find a bijective function $\psi : \mathcal{M} \to \mathcal{N}$ that maps the metadata of each networked device $M \in \mathcal{M}$ to each node $N \in \mathcal{N}$. For this result, we select one $N \in \mathcal{N}$ and forward its description in the LLM context window together with the descriptions of all metadata records $M \in \mathcal{M}$. After determining an initial mapping ψ, we repeat the process in the opposite direction to obtain ψ'.

In realistic deployments, an instance-level bijection between devices and objects is often not identifiable from semantics alone: If multiple identical device instances exist (e.g., several identical lamps), the available metadata and the scene graph descriptions may be insufficient to determine which specific network identity corresponds to which specific physical instance.

We therefore distinguish two levels of mapping. **(1) Instance-level association:** LORIOT computes two directional association relations with confidence scores. In the *digital→physical* direction, the LLM predicts a ranked set of candidate objects $N \in \mathcal{N}$ for each metadata record $M \in \mathcal{M}$ (Listing 2). In the *physical→digital* direction, the LLM predicts a ranked set of candidate device identities $M \in \mathcal{M}$ for each object $N \in \mathcal{N}$. We denote these (generally non-invertible) relations as $\psi_{\mathrm{d}} \subseteq \mathcal{M} \times \mathcal{N}$ and $\psi_{\mathrm{p}} \subseteq \mathcal{N} \times \mathcal{M}$. **(2) Type-level bijection (equivalence classes):** If devices are grouped into *device types* (e.g., "lamp") and objects into corresponding *object types*, then a bijection is well-defined *between types* even when individual instances are ambiguous. This notion is used by our synthetic benchmark ground truth (Section 4.2).

Listing 2. Sample Mapping Result

```
{ relevant_objects:
  [
    {
      obj_id: 1111,
      mapping_accuracy: 0.8
    },
    {
      obj_id: 219,
      mapping_accuracy: 0.8
    }
  ],
  query_achievable: true,
  explanation: The device is a '
      philipps_hue_go_table_lamp', which is a smart
      lamp/light. Among the objects provided, obj_id
       1111 and obj_id 219 are both small, modern
      desk lamps with descriptions and tags
      indicating they are lamps (desk, modern, black
```

```
      , with shades).These best fit the function and
       appearance of a smart lamp, even though there
       is no explicit mention of color-changing
      features or smart technology in the physical
      descriptions. Their size and location (desk)
      also fit the likely placement of a Hue Go lamp
      . No other objects (such as ceiling lights or
      power outlets) match the function of a smart
      lamp as closely.,
}
```

The final mapped data (Listing 2) contains a list of relevant objects (a list of nodes), a predicate indicating whether at least one match could be found, and an explanation for the mapping. The need to generate the predicate and the explanation forces the LLM to improve reasoning by developing a step-by-step argument [24], [31]. As a byproduct of generating ψ, we also collect a mapping accuracy $\rho \in [0, 1]$ that rates each mapping from networked device to semantic description.

In practice, it must be expected that the two mappings do not necessarily form an instance-level bijective mapping, i.e., $\psi \neq (\psi')^{-1}$. In particular, when multiple identical devices are present in the environment, the LLM cannot decide which network identifier corresponds to which device instance, even when the type-level bijection is perfectly correct. We resolve any ambiguities by letting the user decide about any remaining duplications using a situated AR interface. We use ρ to generate ranked suggestions that the user must confirm or override. In the results section, we will demonstrate how the number of explicit interactions needed to resolve ambiguous assignments is only a fraction of the interactions needed to manually classify all devices from scratch.

3.2 User interface generation

For user interface generation, we start with the assumption that each device is described by a metadata record (Listing 3) that explains its operation. A metadata record contains a device name, its network (IP) address, and a list of commands that can be used to interact with the device. We store this information in a generic JSON format that is independent of any vendor or automation standard.

Listing 3. Sample Network Metadata

```
{
  name: nano_leaf,
  IPAddress: 192.168.0.105,
  Port: 16021,
  commands: [
    {
      description: Changes the state of the device,
      endpoint: /api/v1/api_key/state,
      request: PUT,
      request_body: {on: {value: true}},
      response: 204 No Content
    },
    {
      description: Retrieves state of the device,
      endpoint: /api/v1/api_key/state/on,
      request: GET,
      response: 200 OK,
      response_body: {value: true} OR {value: false}
    },
  ]
}
```

TABLE 1
Heuristic linkage from device metadata records to declarative UI widgets.

Metadata pattern (per command)	Generated widget
Boolean state change (e.g., `PUT` with {`on:{value:true/false}`})	`toggleButton` (On/Off)
Stateless action (e.g., `POST/PUT` without user parameter)	`button`
Query/read-back (e.g., `GET` returning a response)	`button` with `showResponseText`
Numeric parameter (e.g., brightness/hue)	`slider` with `$value` placeholder

The commands understood by the device are expressed as HTTP (GET, POST, PUT) or MQTT requests. Description, endpoint, request, and response are mandatory fields per command. We manually collected the command metadata for all devices. However, it would be straightforward to extract this information automatically from product databases of networked devices that support any of the popular automation standards, such as Matter, Nest, ZigBee, etc. For our experiments, we avoided dealing with vendor-lock-in situations by only using devices that are open-source or for which public API documentation exists.

The JSON-formatted metadata shown in Listing 3 is used to automatically generate corresponding UI components represented as another JSON (Listing 5) through an LLM. This representation can then be used in two scenarios. The first user interface runs on an AR client and is spatially registered with physical devices. The second user interface runs on a smartphone as an Android application and does not use any spatial registration or AR. This interface is intended for a comparison of the user experience between AR and non-AR, detailed in Section 5.

For the AR interface, we define templates (Unity prefabs) for the AR client that represent common widgets such as push buttons, toggle buttons, sliders, texts, and spacers. The LLM is instructed to assemble a 2D user interface panel that has a matching widget for every attribute and command enumerated in a device's metadata record. This transformation step is easily prepared by passing the JSON metadata of the target device to the LLM prompt. The returned result is a declarative JSON which represents the interactive UI for the corresponding device as a blueprint. In practice, the LLM follows simple, prompt-specified heuristics such as those summarized in Table 1.

For example, the metadata record in Listing 3 contains a state-changing command (`PUT` to `/state` with an `on.value` boolean), which is translated into the `toggleButton` widget with separate activating/deactivating commands in Listing 5. The metadata also contains a state-query command (`GET` to `/state/on`), which is parsed into a `button` widget configured to display the response text. Numeric parameters are handled analogously via sliders by inserting a `$value` placeholder into the request body (Listing 4), which is later substituted at runtime by the Unity widget prefab.

To enforce a valid JSON syntax, the structured output of the LLM is utilized. The few-shot examples in the prompt may contain placeholders that serve as parameters in the

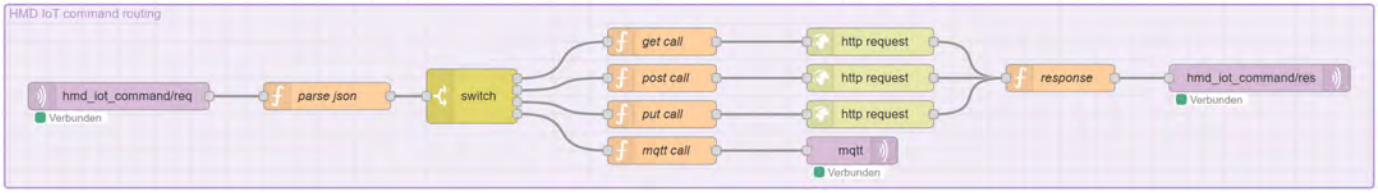

Fig. 3. Node-RED command routing example. The entire graph represents receiving commands from various clients and routing them to the appropriate devices using different protocols.

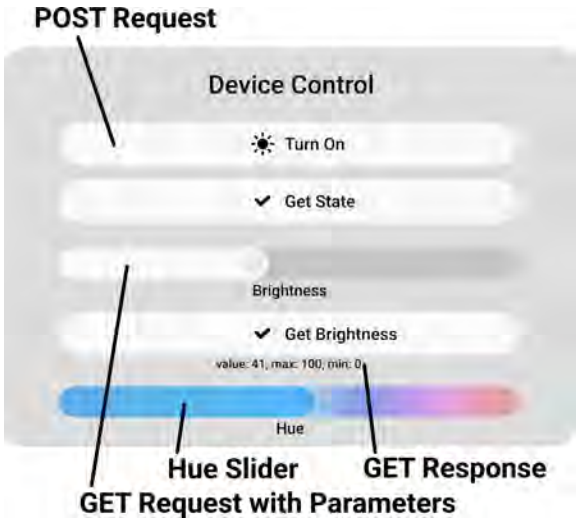

Fig. 4. Example of an automatically generated user interface for a lamp.

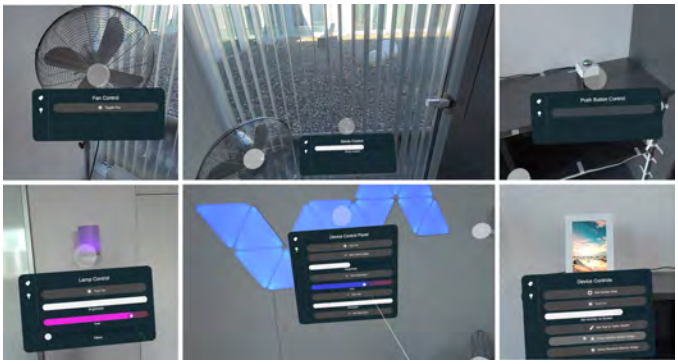

Fig. 5. Automatically generated user interfaces for various devices.

commands. For example, a toggle button distinguishes two states (active, inactive), while a slider must send its current value using a "set value" command. The LLM model must correctly insert the placeholder into the given command data so that the request body is valid. The Unity prefab is programmed to insert the actual value into the command at the placeholder position. Providing the few-shot sample from Listing 4 is enough to make this system work. Figure 4 shows the resulting user interface from the shortened Listing 5.

Of course, the approach described above is still limited to simple commands that can be used independently. The automatically generated user interface is not able to represent complex control procedures that would require a finite state machine or a similarly powerful user interface formalism.

Listing 4. Custom slider widget prompt

```
The request body should contain $value as a
    placeholder for the slider value. e.g. {hue:
    $value}
```

Listing 5. Sample user interface representation

```
[
    { text: { text: Device Control }},
    { spacer: { height: 20 }},
    { toggleButton: {
            buttonName: Turn On,
            deactivatingButtonName: Turn Off,
            iconType: Sun,
            activatingCommand: { ... },
            deactivatingCommand: { ... }
    }},
    { spacer: {height: 20}},
    { button: {
            buttonName: Get State,
            iconType: Check,
            command: { ... },
            showResponseText: true
    }},
    { spacer: {height: 20}},
    { slider: {
            label: Brightness,
            minValue: 0,
            maxValue: 100,
            wholeNumbers: true,
            command: {
                description: Change brightness,
                endpoint: /api/v1/api_key/state,
                request: PUT,
                request_body: {brightness:{value:
                    $value}},
                response: 204 No Content
            }
    }}
]
```

When the AR client receives the JSON-formatted user interface specification, it uses a JSON parser to extract the data items for each device. It instantiates a corresponding widget for each item and fills it with the extracted data values. This procedure is purely data-driven, and new physical environments do not require any code modifications of the Unity application (i.e., the C# code).

The Unity application also contains the network communication code to talk to IoT devices via MQTT messages, which is used by the widget prefabs. All network traffic is sent to the server, which uses Node-RED to determine any final command routing to the IoT devices (Figure 3). Routing is necessary because our AR clients do not support network standards such as ZigBee natively.

Commands that return a response display it for a default period of five seconds below the widget. If the response is a JSON object, LORIOT formats the data in a more readable way, as can be seen in Figure 4; otherwise, the string will

be directly displayed. We also support the inheritance of behaviors to generate more sophisticated variants of basic widgets. For example, a hue slider works like a basic slider, but employs a more sophisticated visual representation. The LLM is specifically prompted to prefer the sophisticated widget variant, if one exists. For example, if the command controls the hue of a lamp, the LLM should prefer the hue slider over a normal slider.

Certain UI components like toggle buttons or sliders have internal state (e.g., the current slider value). This state needs to be initialized to match the actual device state. We address this requirement by prompting the LLM to generate a query command to initialize such stateful UI components.

The user interface client running on the smartphone is written in Dart and Flutter. It receives the same JSON data as the AR client, but converts it into a conventional 2D user interface without any spatial registration.

3.3 Natural language control

In addition to gesture-operated user interfaces, our system supports voice-based interaction for hands-free device control. This is particularly useful in an AR context, where the user's hands might be occupied with other tasks. The interaction flow is initiated by the user speaking a command, which is captured by the microphone of the Meta Quest 3.

The captured audio is processed by OpenAI's Whisper model for speech-to-text transcription. The resulting text is sent to a LangChain[8] agent running on our server. This agent is designed to interpret natural language commands.

To process a command, the agent is provided with two pieces of information: the user's transcribed request and the full set of API specifications for all IoT devices, which it retrieves from the CouchDB database. By combining the user's intent with the knowledge of available device capabilities, the agent can reason about the request and identify the appropriate device and command.

Finally, the agent constructs a precise, executable control request. The response is formatted as a JSON object that can be directly processed by our server's command routing logic, as previously described. The server dispatches the command to the corresponding IoT device via MQTT. Linking voice commands to device actions provides a seamless and intuitive experience within the smart environment.

3.4 Spatial registration

For spatial registration of user interfaces to physical devices in AR, it is necessary to obtain the pose of each device in the current tracking coordinate system of the AR client. The AR client relies on a hardware-accelerated 3D localization (*ego-tracking*) which delivers the current pose.

The objects in the SSG are created by first constructing a 3D point cloud from an RGB-D image stream (*3D scanning*) and then segmenting the point cloud into distinct objects. We fit axis-aligned 3D bounding boxes as proxy geometry for each object, because this choice ensures that only partially reconstructed objects can still be assigned recognizable bounding geometry. In general, the coordinate systems used for the bounding boxes of objects in the SSG and the coordinate system used by the ego-tracking of the AR client are not aligned. Therefore, spatial registration must consider two related problems:

(1) *Relocalization of the AR client*: The ego-tracking of the AR client initializes the coordinate system to the pose of the first sensor reading. The coordinate system is relative to the pose recorded at the moment when the AR client was started, which is different in each session. To enable spatial constancy across sessions, some AR libraries provide relocalization based on spatial anchors, which contain an index of visual features in the environment. However, anchors are platform-specific and can be cumbersome to use.

(2) *Registration of the coordinate systems of the SSG and the AR client*: The SSG is derived from a 3D scan, and we must determine the rigid transformation from the 3D scan to the current ego-tracking. This process requires knowing the pose of feature points or objects in both representations.

Our framework addresses both problems at once by placing a single fiducial marker. We performed the 3D scanning for the prior SSG creation with the Record3D[9] app on an iPad Pro (11-inch, third generation), while we used a Meta Quest 3 as an AR client for the subsequent user study. The transformation implied by the marker lets us switch from the SSG coordinate system to the AR client coordinate system. This simple approach is sufficient for our purposes, since our focus lies on generating user interfaces via semantic reasoning.

4 Mapping evaluation

This section evaluates the technical core of LORIOT: the LLM-based mapping between network-side device identities and SSG nodes in the SSG. Our goal is to quantify *how reliably the mapping works at scale* under controlled conditions, independently of the AR user interface and human factors (which we study separately in Section 5). We ask the following questions: **(Q1)** How accurate is the device mapping in either direction (*digital→SSG* vs. *SSG→digital*) when many heterogeneous devices are present? **(Q2)** Do the model's self-reported confidence scores align with mapping correctness, i.e., can confidence-weighted metrics serve as an indicator of reliability? We answer these questions using synthetically generated rooms to enable large-scale, repeatable experiments with known ground truth and to vary the number and composition of devices in a way that would be impractical with physical rooms. We report on our experiment with physical rooms and devices in Section 5.

4.1 Room generator

To test the scalability of LORIOT, we implemented a room generator using Unity, which randomly populates an arbitrarily sized room with devices. These synthetically generated rooms can then be used to test the performance of our IoT mapping approach. Thus, the mapping evaluation is a controlled *synthetic-placement* benchmark: we vary the number and arrangement of devices while keeping the underlying device descriptions consistent with those observed in the real office scan.

8. https://github.com/langchain-ai/langchain

9. https://record3d.app/

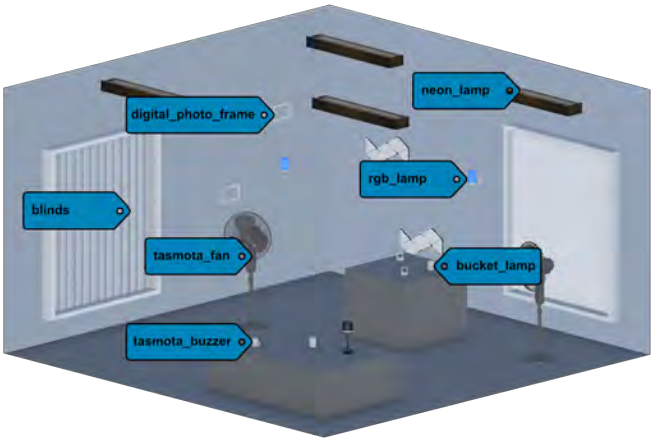

Fig. 6. Output of the room generator with 20 devices placed at random positions.

We directly generate SSG [15] node data based on randomly placed synthetic devices. Figure 6 shows a visualization of a room generated with the devices placed. For each device type, we provide SSG data (caption, object-tag, descriptions, etc.) which are copied from previous real world scans of our office and additionally a tag expressing its potential placements (wall, floor, ceiling or table). This tag is only used in the synthetic room generation process to ensure that devices are placed in appropriate locations within the room. Based on our experience of creating multiple scans of the same environment, feature extraction for a certain device type is always similar across scans. In order to make the SSG as similar as possible to an actual real world scan, we initialize device nodes with data from a scan of the physical counterparts.

We can generate a room of arbitrary size and with an arbitrary number of devices. Decorations are omitted, since our algorithm operates only on device nodes, after non-device nodes have been filtered out. The generator ensures that the bounding boxes of the devices do not overlap or intersect the room boundaries. To provide enough placement opportunities, we generate randomly sized desks (cubes in Figure 6) on which smaller devices are placed. The actual position and rotation of the final SSG node data is determined by the positions and orientations randomly assigned to the synthetic devices.

The currently supported device types are: *blinds controller*, *ceiling panel*, *photo frame*, *lamp*, *nanoleaf*, *neon tube*, *philips hue lamp*, *tasmota lamp*, *tasmota fan*, *tasmota button*, and *tasmota buzzer*.

We mimic real-world devices in our test office, which have been used in the user study described in Section 5. Note the addition of *neon tube* and *ceiling panel*. The *neon tube* exists in the test office, but it is not an IoT device. The *ceiling panel* was introduced to increase the variety of ceiling-mounted devices.

The SSG information for all other physical devices has been copied from the actual SSG data from the test office. The device metadata was also copied. If multiple instances of one device type were generated, the device names were suffixed with a number (e.g., *lamp 1*, *lamp 2*).

4.2 Evaluation method

The performance of the system was evaluated in 100 randomly generated rooms, each containing 20 random devices, providing a large-scale assessment of the system's scalability and robustness. The SSG data of the synthetic devices was taken from actual SSG data of the underlying physical room. For each synthetic room, the predicted associations were compared to ground truth (correct mapping) to determine key performance metrics. As LORIOT is not able to distinguish digital identities of identical device types, the ground truth always contains all instances of a given device type. For example, if there are two lamps in the room, the ground truth for each lamp SSG node will include both the network identities *lamp 1* and *lamp 2*. The same applies to the other mapping direction: Each network lamp device will include both lamp SSG nodes in its ground truth. Note that the order of devices in the generated mapping data is not of interest. We only evaluated if the correct mapping was found. We used the following classifications:

- *True positives* (TP): An association that is correctly predicted by the system also exists in the ground truth.
- *False positives* (FP): An association that is predicted by the system does not exist in the ground truth.
- *False negatives* (FN): An association that exists in the ground truth is not predicted by the system.

Based on these classifications, we calculated four metrics:

- *Precision (P)* measures the accuracy of positive predictions, calculated as $P = \frac{TP}{TP+FP}$.
- *Recall (R)* measures the system's ability to identify all relevant mappings, calculated as $R = \frac{TP}{TP+FN}$.
- *F1 Score ($F1$)* is the harmonic mean of Precision and Recall, providing a balanced measure of mapping performance, calculated as $F1 = 2 \cdot \frac{P \cdot R}{P+R}$.
- *The Intersection over Union (IoU)*, also known as the Jaccard index, measures the overlap between prediction and ground truth, calculated as $IoU = \frac{TP}{TP+FP+FN}$.

We evaluated the scores with macro- and micro-averaging to take the frequency at which the devices appear into account.

Furthermore, the comparison between weighted and unweighted metrics provides valuable insight into the confidence ρ of the system in its predictions. Unweighted metrics treat every prediction or each type of device equally, whether the system was confident in its response or not. A correct prediction with 100% confidence has the same impact as a correct prediction with 51% confidence. This approach provides a straightforward view of the system's raw accuracy. In contrast, the weighted metrics factor in the confidence of the system for each prediction. In this way, not only the mapping prediction itself, but also the prediction rating of the LLM can be evaluated.

This feature is achieved by replacing the standard counts of true positives, false positives, and false negatives with the sum of their associated confidence scores. For example,

TABLE 2
Overall success rate (**U**n**W**eighted and **W**eighted)

Metric Type	digital→SSG				SSG→digital			
	P	**R**	**F1**	**(S)IoU**	**P**	**R**	**F1**	**(S)IoU**
Macro (UW)	0.72	0.99	0.80	0.72	0.10	0.16	0.12	0.10
Macro (W)	0.76	0.89	0.79	0.70	0.12	0.13	0.12	0.10
Micro (UW)	0.63	0.99	0.77	0.63	0.12	0.07	0.09	0.05
Micro (W)	0.68	0.89	0.77	0.63	0.14	0.06	0.08	0.04

with $C_i \in [0,1]$ denoting confidence in association i, the weighted precision is calculated as

$$P_{\mathrm{w}} = \frac{\sum_i^{\mathrm{TP}} C_i}{\sum_i^{\mathrm{TP}} C_i + \sum_j^{\mathrm{FP}} Cj}.$$

Weighted version of the other metrics (Recall, F1-Score, and Soft-IoU) are calculated analogously.

4.3 Overall success rate

The global evaluation (Table 2) indicates a strong difference in performance between the two mapping directions. LORIOT demonstrates strong capabilities in mapping network identities to SSG nodes, with a macro-averaged unweighted F1 score of 0.80 using baseline device metadata. This score indicates a solid level of accuracy and completeness in its predictions.

In contrast, LORIOT struggles with mapping from SSG nodes to network identities. The macro-averaged unweighted F1 score for this association is significantly lower, at just 0.12. This discrepancy highlights the difficulty of accurately mapping objects back to their corresponding devices, a key area for future improvement.

This result must be expected given that the API abstractions and names are not descriptive enough to identify the corresponding devices. Additional contextual information, such as the physical characteristics of objects and their potential positions in the environment, would probably improve the accuracy of the mapping ρ. We explore this hypothesis in an extended evaluation in the following Section 4.4.

4.4 Extended evaluation with enriched device metadata

We assumed that the results could be improved with richer metdata for each IoT device. Therefore, we repeated the evaluation to include enriched device descriptions. Beyond the baseline network metadata (device name, API commands), we augmented each device record with *physical context attributes*: (1) mounting location (e.g., "wall-mounted," "ceiling-mounted," "on table"), (2) primary materials (e.g., plastic, metal, glass), (3) dominant color, and (4) the dimensions of the device. These attributes reflect information that IoT manufacturers could reasonably include in product metadata or device profiles. We re-ran the identical evaluation procedure on the same 100 randomly generated rooms using the enriched metadata.

With enriched device metadata, the macro-averaged unweighted F1 score improved substantially from 0.80 to **0.88**, representing an 10% absolute improvement. Weighted F1 also improved from 0.79 to **0.89**, indicating that the LLM not only produces more correct predictions, but also expresses higher confidence in them. This improvement is observed across device types. For example, *lamp* performance improved from 0.42 to 0.85 (F1 improvement of +43 percentage points), *tasmota lamp*, from 0.47 to 0.65, and *ceiling panel*, from 0.73 to 0.81.

These results confirm that enriching network-side device metadata with physical context attributes substantially improves both mapping directions and overall confidence. In practical deployments, such metadata could be derived from IoT manufacturer datasheets or added incrementally as part of standardized device profiles.

4.5 Bijective mapping scores

While the two directional relations ψ_{d} and ψ_{p} are informative, they do not directly imply an instance-level bijection in the presence of identical devices. To report a single, presentable score for a *bijective* mapping, we therefore evaluate an additional metric.

We compute a bijection between *device types* and *object types* by collapsing identical instances into equivalence classes. This score measures whether the system can correctly match *types* even when individual instances are indistinguishable. We partition devices into *device types* by stripping instance suffixes (e.g., `lamp 1`, `lamp 2` $\mapsto$ `lamp`), and analogously group physical objects into *object types*. For a given type t, let $\mathcal{M}_t$ be the set of device instances of type t, and $\mathcal{N}_t$, the set of ground-truth object instances of the same type. From the digital→physical relation ψ_{d}, we keep the top-k candidate objects (we use k=3) per device and build a bipartite graph between $\mathcal{M}_t$ and $\mathcal{N}_t$. We then compute a *maximum-cardinality matching* $S_t \subseteq \mathcal{M}_t \times \mathcal{N}_t$ (one-to-one) and define

$$P_t = \frac{|S_t|}{|\mathcal{M}_t|}, \qquad R_t = \frac{|S_t|}{|\mathcal{N}_t|}, \qquad F1_t = \frac{2P_tR_t}{P_t + R_t}.$$

Macro-averaging reports the mean of $P_t, R_t, F1_t$ across types and rooms; micro-averaging sums $|S_t|$, $|\mathcal{M}_t|$ and $|\mathcal{N}_t|$ across all types/rooms before computing $P, R, F1$. To reflect confidence, we additionally compute a *maximum-weight* matching using edge weights given by the mapping accuracy of the LLM and report the average matched weight per device.

In the baseline evaluation, we obtain a macro-averaged $F1$ of **0.82** (macro precision/recall: 0.82/0.82). Aggregated across all rooms, the micro-averaged score is $F1$=**0.82** (1640 correctly matched pairs out of 1999), with a confidence-weighted average of **0.75** per device.

In the extended evaluation with enriched metadata, the macro-averaged F1 improved from 0.82 to **0.91**. Micro-averaged F1 across all 100 rooms improved from 0.82 to **0.91**.

4.6 Analysis by device type

While the overall performance is strong for mapping network identities to SSG nodes, a detailed breakdown by device type (Table 3) reveals significant variations.

LORIOT achieved near-perfect unweighted F1 scores of 1.00 for several device types, including *blinds controller*,

TABLE 3
Performance metrics by device type (macro-averaged), comparing unweighted (left) and weighted (right) results.

Device Type	Unweighted Macro-Avg				Weighted Macro-Avg			
	P	R	F1	IoU	P	R	F1	SIoU
blinds controller	1.00	1.00	1.00	1.00	1.00	0.97	0.98	0.97
ceiling panel	1.00	1.00	1.00	1.00	1.00	0.98	0.99	0.98
photo frame	1.00	1.00	1.00	1.00	1.00	0.96	0.98	0.96
lamp	0.39	0.62	0.45	0.38	0.39	0.56	0.44	0.37
nano leaf	1.00	1.00	1.00	1.00	1.00	0.95	0.97	0.95
neon tube	1.00	1.00	1.00	1.00	1.00	0.96	0.98	0.96
philips hue lamp	0.42	0.46	0.43	0.40	0.42	0.37	0.39	0.33
tasmota lamp	0.06	0.06	0.06	0.06	0.06	0.04	0.05	0.04
tasmota buzzer	0.95	1.00	0.97	0.95	0.96	0.95	0.95	0.91
tasmota fan	1.00	0.89	0.93	0.89	1.00	0.83	0.89	0.83
tasmota button	0.58	0.68	0.61	0.58	0.55	0.48	0.50	0.41

ceiling panel, *photo frame*, *nano leaf*, and *neon tube*. These results are from the extended evaluation with enriched device metadata. In the baseline evaluation using network metadata alone, performance on ceiling-mounted devices was lower. This indicates that, for devices with consistent visual features and sufficient descriptive metadata, the system achieves high accuracy. The objects should be visually distinguishable by their capabilities. For example, the 3D-printed *tasmota lamp* were detected as *cups* by the SSG, and the small size of the *tasmota button* relative to its larger corpus challenged the SSG to correctly classify it as a button. Instead, it was classified as a *tissue box*. The corresponding weighted F1 scores for these devices with a high unweighted score remain very high, with a minimal drop, suggesting that the system is also very confident in these predictions.

For other devices, LORIOT demonstrated robust performance. In the baseline evaluation, *tasmota buzzer* and *tasmota fan* achieved unweighted F1 scores of 0.86 and 0.99.

We noted a drop in performance for a few devices, which can be interpreted as an avenue for future research. The difficult devices in the baseline include *lamp* (0.42), *philips hue lamp* 0.57, and *tasmota push button* (0.79). The lower scores for these devices can be attributed to sparse SSG data or insufficient network metadata.

Device types that were already performing well in the baseline (e.g., *blinds controller*, *photo frame*, *tasmota fan*) see only marginal improvements in the extended evaluation, while poorly-performing types benefit substantially, demonstrating that enriched metadata particularly helps disambiguate visually similar or under-described devices.

4.7 Confidence and weighted metrics

The comparison between unweighted and weighted metrics provides valuable insight into the confidence of the system in its predictions. In a perfect scenario, these metrics would be nearly identical. However, our results consistently show that the weighted F1 scores are lower than their unweighted counterparts in the baseline evaluation. The observed drop in scores (e.g., the global unweighted F1 score for device-to-object metrics is 0.80, while the corresponding weighted score is 0.79) suggests that LORIOT is frequently accurate, but tends to make many correct predictions with low confidence. The extended evaluation with enriched metadata shows more aligned weighted and unweighted scores, suggesting that better metadata leads to both higher accuracy and higher confidence in predictions.

5 User Study

The technical evaluation in the previous section establishes how well our technical core (automatic localization of devices with LLM) works, but does not reveal whether human users actually prefer the automatically generated AR user interfaces to conventional controls (usually smartphone apps or infrared remote controls). Therefore, we conducted a user study that compares the AR and non-AR conditions. This study was approved by the ethics committee of our institution.

Our study was designed as a within-subjects experiment with two conditions: One with AR interaction (using a Meta Quest 3) and one without AR interaction (non-AR). Participants were asked to complete a set of tasks in both conditions, and their performance was measured in terms of success and task completion time. After each condition, participants were asked to complete the NASA TLX [16] and the System Usability Scale (SUS) [5] questionnaires to evaluate the usability of the system. We filled our office with 26 devices and divided them into two groups D_1 and D_2 of 13 devices each. To prevent biases and learning effects, we counterbalanced the order of conditions among participants in four groups:
1: AR/D_1 $\rightarrow$ non-AR/D_2, 2: AR/D_2 $\rightarrow$ non-AR/D_1,
3: non-AR/D_1 $\rightarrow$ AR/D_2, 4: non-AR/D_2 $\rightarrow$ AR/D_1.

Both groups of devices were placed in the same room (Figure 7). However, participants could only interact with one group at a time. All devices were easily accessible and visible to participants. However, participants were not introduced to the devices prior to the event.

Device Group D_1:
- philips hue light strip
- desktop fan ($\times 2$)
- table controller ($\times 2$)
- rgb lamp ($\times 3$)
- lamp ($\times 5$)

Device Group D_2:
- nanoleaf
- tasmota lamp ($\times 2$)
- tasmota fan large ($\times 2$)
- photo frame ($\times 3$)
- blinds controller ($\times 2$)
- philips hue lamp ($\times 3$)

With these device groups in place, the study aimed to evaluate two research questions:

- RQ1: Does the additional location information of devices in AR improve the task completion time compared to a condition without AR, which lacks location information?
- RQ2: How does AR influence the perceived quality of interaction compare to a conventional approach with multiple smartphone applications and physical remotes?

To evaluate RQ1, we ensured that we provided multiple instances of the same device in the office, so that participants could experience the difference between the known locations of the devices that use AR and no location information in the non-AR condition.

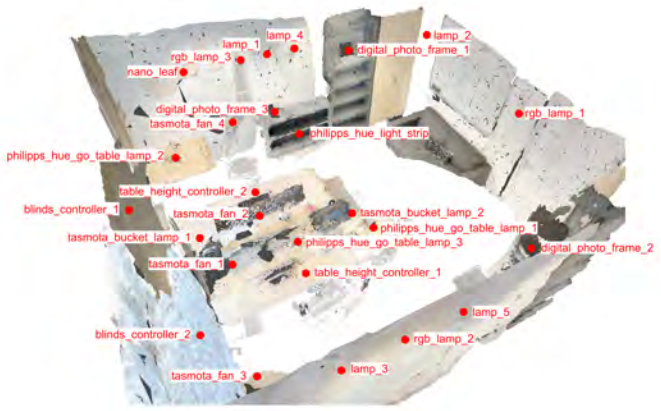

Fig. 7. Test office with all 26 device placements.

TABLE 4
Wilcoxon signed-rank test results comparing AR and non-AR conditions for NASA TLX and SUS questionnaires.

Metric	p-value	Effect Size
NASA TLX (21-point scale)		
Mental Demand	0.023	$r = -0.60$
Effort	0.011	$r = -0.68$
Frustration	0.048	$r = -0.61$
Physical Demand	0.162	$r = -0.38$
Temporal Demand	0.207	$r = -0.39$
Performance	0.819	$r = 0.08$
System Usability Scale (SUS)		
SUS Score	0.009	$r = 0.71$

5.1 Participants

We recruited 20 participants (15 male, 5 female) for the study, with an average age of 26.95 years ($\sigma = 4.38$). On average, they rated their previous experience on a 5-point Likert scale with 2.9 ($\sigma = 1.29$) for AR, and, 2.6 ($\sigma = 1.10$) for IoT. All participants provided their informed consent prior to participating in the study. Participants were compensated with 15 €.

5.2 Tasks

Given the device groups D_1 and D_2, the participants were asked to complete a set of tasks under both conditions that correspond to the devices in the groups. The tasks were always presented in the same order.

Tasks for D_1:

- Color the Philips Hue light strip red
- Activate one specific desktop fan
- Increase the height of one specific table
- Make specific RGB lamp blink slowly in purple
- Activate two specific wall mounted lamps

Tasks for D_2:

- Color the Nanoleaf lamp green
- Turn on specific bucket lamp
- Show beach image on specific digital photo frame and skyline on other one
- Open the blinds of one specific window

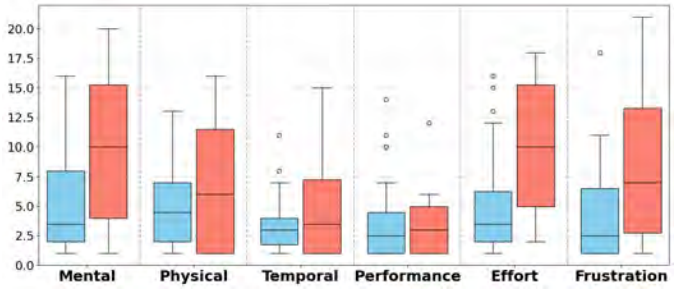

Fig. 8. NASA TLX (1=very low, 21=very high; for performance 1=perfect, 21=failure). AR, non-AR.

- Colorize a specific Philips Hue Go Table lamp in orange and another one in blue

To avoid bias, participants were not introduced to the devices in advance. In addition, tasks were communicated so that participants did not receive valuable information about the devices. Pointing gestures were used to tell participants which device to interact with: "Please turn on the lamp over there (pointing gesture)" or "Please close the blinds on the left window".

For the non-AR condition, participants were provided with a smartphone on which three applications were installed—one for the Philips Hue devices, one for the Nanoleaf lamp, and one custom-built for all the other devices. The custom smartphone application reused the user interface automatically generated from Listing 5 to control the devices. In addition to the apps, three physical infrared remotes were provided, which controlled the RGB lamps. This setup was chosen to reflect a realistic scenario. The current state of IoT device controls often requires one to use multiple apps and remotes.

Participants were informed that there was always exactly only one way to interact with a device, e.g., the Nanoleaf lamp can only be controlled via the Nanoleaf app and not additionally via the infrared remotes. No further assistance was provided.

However, in the AR condition, white spheres were displayed right next to the devices. These spheres could be selected by the controller to display the AR user interface for the corresponding device. In these panels, participants could control the associated device using the user interface generated from Listing 5.

5.3 Results

All participants were able to successfully complete all the tasks in a reasonable amount of time.

Questionnaires: We analyzed scores from NASA TLX [16], which used a 21-point scale, and SUS [5] using a Wilcoxon signed-rank test to compare AR and non-AR conditions. The test results showed that AR resulted in significantly lower (Table 4) mental demand, effort and frustration than non-AR. No significant differences were found for physical demand, temporal demand, and performance. The mean values for NASA TLX are displayed in Figure 8. Usability was rated significantly higher for AR ($\mu = 84.3$, $\sigma = 13.1$) than for non-AR ($\mu = 62.9$, $\sigma = 26.0$).

We analyzed the preference answers using a binomial test for significant deviations from a balanced choice (50%)

of AR and non-AR. The results show that AR was significantly preferred in terms of efficiency ($p < 0.001, g = 0.4$), intuitiveness ($p = 0.41, g = 0.25$), and fun ($p < 0.001, g = 0.4$). However, for overall preference, the test results were not significant, although the majority of participants still preferred AR (Table 5).

Performance: We recorded the task completion time for each subtask and averaged it for each participant. As this data was not normally distributed, we used the Wilcoxon signed-rank test to compare AR with non-AR. The test indicates that the participants were significantly ($p = 0.003, r_{rb} = -0.73$) faster with AR ($\mu = 39.2s$, $\sigma = 21.1$) than with non-AR ($\mu = 68.2s$, $\sigma = 26.9$).

A t-test confirmed that there was no significant ($p = 0.006, d = 0.69$) difference between the two groups of devices, which were counterbalanced so that we assigned exactly the same number of participants to each combination of AR/non-AR and device groups.

Qualitative feedback: The feedback of the participants provided rich insight into the user experience, particularly highlighting the challenges of the non-AR condition and the benefits of the AR system. Several common themes emerged from the interviews.

A major point of frustration for participants in the non-AR condition was the decentralized and disparate nature of the controls. The sheer number of apps and physical remotes made it difficult to identify which control corresponded to which device. Multiple participants (e.g., P2, P4, P6, P11) struggled with this initial discovery task. Participant P11 explicitly stated that the most frustrating part was having "three different remote controls and three different apps," requiring them to "try, like, every different remote and app to find which one it is for." This sentiment was echoed by participant P7, who commented that they had to "test it again and again in order to find out which device should be controlled by which control or which app." Participant P19 described the non-AR setup as "more complex," noting that finding the right app and learning the different interfaces would be "terrible" in an unfamiliar environment.

A related difficulty was the spatial ambiguity of the devices and their controls. Participants struggled to visually connect the device in the app interface to its physical location. This difficulty was highlighted by participant P2, who said that they were switching a device on and off and "did not realize that it was behind me." Participant P11 also pointed out that, without labels, they did not know which device was "fan 1, fan 2, lamp 1, lamp 2, lamp 3" in the app interface, further complicating the task of finding the correct device to control.

In contrast, the AR condition received overwhelming positive feedback. Participant P10 found that, with AR, they "just found the object" and could "change it" immediately, contrasting this experience with the non-AR condition where they had to "search for everything and had to memorize the objects." Participant P18 summarized the general sentiment, stating, "With AR, it was great. I did not have any difficulty."

Limitations (task ordering).: While we counterbalanced the order of interaction *conditions* (AR vs. non-AR) in our within-subject design, the *order of tasks* was kept fixed for all participants. However, we believe that the impact of a potential ordering bias is minimal. First, the tasks were designed to be relatively simple and quick to complete, which minimizes fatigue. Second, the counterbalancing of conditions ensures that any learning effects are distributed across both AR and non-AR conditions, reducing their impact on the overall results. Third, the significant differences observed in task completion times and user preferences suggest that the benefits of AR are robust and cannot be explained by task ordering alone.

TABLE 5
Number of participants who preferred AR or non-AR when asked the following questions.

Question	AR	non-AR
Which condition did you find more efficient to use?	18	2
Which condition did you find more intuitive to use?	15	5
Which condition did you find most fun to use?	18	2
Which condition did you prefer overall? AR or non-AR?	14	6

6 Discussion

Our findings demonstrate the effectiveness of LORIOT in streamlining IoT device localization within AR environments, but they also highlight critical areas for future research and incremental advancements that build directly upon our foundational framework.

One current limitation lies in the initial SSG generation. While LORIOT effectively leverages the SSG generation of ConceptGraphs, the raw output still requires some manual preprocessing, such as filtering duplicate objects or explicitly identifying potential IoT candidates. Recent SSG pipelines that were published after the experiments reported in this paper have already achieved a much higher degree of automation and faster processing times.

Our evaluation revealed an asymmetry in mapping performance. LORIOT accurately maps network identities to physical objects, but struggles with the reverse mapping. Therefore, we also computed a *type-aware* bijection (type-level) with $F1{=}0.82$, which demonstrates that mutually agreed assignments are highly reliable, but identical instances remain underconstrained without richer metadata.

Our extended evaluation confirmed that Enriching metadata of networked devices with details about their physical appearance—such as object size, material, or general location (e.g., on a wall or table)—improves mapping in both directions. The digital→physical F1 score was increased from 0.80 to 0.88, and the bijective F1, from 0.82 to 0.91. This improvement is particularly pronounced for previously low-performing device types, demonstrating that metadata enrichment is essential for automated device localization. We expect that, as IoT manufacturers provide more detailed metadata, mapping will naturally improve.

A remaining challenge for full automation is the disambiguation of multiple identical IoT objects. LORIOT currently addresses this issue by transforming a complex, open-ended search into a structured, guided user intervention, presenting ranked candidates with confidence scores. This approach significantly reduces the manual effort from identifying all possible devices to selecting among a few highly-ranked suggestions. It is considerably easier to map small

groups of 1–5 devices than it is to map 26 devices at once. Future work can incrementally build upon this foundation by integrating LORIOT with complementary localization techniques. For instance, combining our semantic mapping with radio-based localization [29], sequential device activation [38] or ODIF/BLEARVIS approach [39], which fuses computer vision with radio signal analysis, could automate the final disambiguation step for identical devices, further reducing human involvement.

Our user study showed that participants, including those without AR experience, found the user interfaces generated by the LLM easy and pleasant to use. The overall effectiveness of our approach, reflected in faster task times (RQ1) and higher user preference over decentralized methods (RQ2), was further reinforced by the usability of the automatically generated user interfaces. We have extended our JSON UI syntax to allow for stateful UI components to retrieve their current state. However, this currently requires concrete retrieval requests. If a lamp returns its current color as RGB and not as required as a hue value, then additional calculations of the retrieved value are required. One could ask the LLM to additionally write C# or Dart code to make this parsing work, but it would contradict our forced structured output driven JSON approach. So this is a potential field for future research. It is important to note, however, that all our automatically generated user interfaces worked correctly without any manual adjustments.

Finally, our use of a QR code marker to align virtual content with the real world was simple and reliable. While not feasible for large-scale deployment, it serves as a practical solution for our current system. A more sophisticated solution would use a cross-platform anchor mechanism, such as BOWA [34], which uses the coarse scene structure itself as an anchor. Integrating such a platform-independent method represents a clear next step for future work.

7 Conclusions

Interacting with the growing ecosystem of devices is often fragmented, relying on a patchwork of manufacturer-specific apps and remotes. In this paper, we argue that AR presents a more intuitive alternative, but is predicated on solving the critical challenge of device localization. We introduced LORIOT, a novel system that addresses this bottleneck by leveraging SSG and LLM to automatically map IoT devices to their corresponding physical objects. A key component of LORIOT is a declarative user interface specification that facilitates both manufacturer-independent device control and the automatic generation of interactive UI panels within AR.

Our primary contribution is the design and implementation of the first end-to-end system that integrates these components into a single pipeline, which automates the crucial steps of device mapping and UI generation.

Our evaluations demonstrated the effectiveness of this approach. In a large-scale evaluation across 100 synthetically generated rooms, the mapping from network identity to physical object achieved a F1 score of 0.80 (using standard device metadata) and 0.88 (using enriched metadata that includes physical context attributes), confirming the ability of LORIOT to accurately identify device locations based on semantic and network information. These results validate that practical improvements in device metadata enhances localization performance. Furthermore, a user study with 20 participants showed that the AR-based interaction system was significantly faster, less mentally demanding, and perceived as more usable and enjoyable than traditional, fragmented control methods involving multiple applications and remotes.

Although challenges remain, particularly in the reverse mapping from physical objects to network identities and in fully automating the handling of identical device instances, LORIOT represents a substantial advancement. Our extended evaluation demonstrates that addressing metadata sparsity further improves performance, pointing to a practical path forward for real-world deployments. By providing a robust and scalable end-to-end pipeline for device localization and interaction, LORIOT lays the groundwork for the practical deployment of a sophisticated AR interface in complex, real-world settings. This research paves the way for future environments in which interacting with the digital and physical worlds is a seamless and intuitive experience.

Acknowledgments

This work was supported by the Alexander von Humboldt Foundation, funded by the German Federal Ministry of Research, Technology and Space, the German Research Foundation *DFG* under Germany's Excellence Strategy EXC 2120/2 (390831618), and the Austrian Science Funds *FWF* (I5912).

References

[1] T. Andrade and D. Bastos. Extended Reality in IoT scenarios: Concepts, Applications and Future Trends. In *2019 5th Experiment International Conference (Exp.at'19)*, pp. 107–112, June 2019. doi: 10.1109/EXPAT.2019.8876559

[2] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese. 3d scene graph: A structure for unified semantics, 3d space, and camera. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pp. 5664–5673, 2019.

[3] A. Blaga, C. Militaru, A.-D. Mezei, and L. Tamas. Augmented reality integration into MES for connected workers. *Robotics and Computer-Integrated Manufacturing*, 68:102057, Apr. 2021. doi: 10.1016/j.rcim.2020.102057

[4] Ó. Blanco-Novoa, P. Fraga-Lamas, M. A. Vilar-Montesinos, and T. M. Fernández-Caramés. Creating the Internet of Augmented Things: An Open-Source Framework to Make IoT Devices and Augmented and Mixed Reality Systems Talk to Each Other. *Sensors*, 20(11):3328, Jan. 2020. doi: 10.3390/s20113328

[5] J. Brooke et al. SUS—a quick and dirty usability scale. *Usability Evaluation in Industry*, 189(194):4–7, 1996. doi: 10.1201/9781498710411-35

[6] Y. Cao, P. Jiang, and H. Xia. Generative and Malleable User Interfaces with Generative and Evolving Task-Driven Data Model, Mar. 2025. doi: 10.48550/arXiv.2503.04084

[7] I. Chaerony Siffa, J. Schäfer, and M. M. Becker. Adamant: A JSON schema-based metadata editor for research data management workflows. *F1000Research*, 11:475, July 2022. doi: 10.12688/f1000research.110875.2

[8] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan. Yolo-world: Real-time open-vocabulary object detection, 2024.

[9] J. A. Dias, N. P. Campos, and M. Boavida. Toward an IoT Framework for the New Generation of XR Applications. *IEEE Access*, 13:68726–68752, 2025. doi: 10.1109/ACCESS.2025.3560547

[10] W. Y. A. Espinal, J. Jimenez, and L. Corneo. An eXtended Reality Data Transformation Framework for Internet of Things Devices Integration. In *Proceedings of the 14th International Conference on the Internet of Things*, IoT '24, pp. 10–18. Association for Computing Machinery, New York, NY, USA, Mar. 2025. doi: 10.1145/3703790.3703792

[11] P. Fleck, A. S. Calepso, S. Hubenschmid, M. Sedlmair, and D. Schmalstieg. Ragrug: A toolkit for situated analytics. *IEEE Trans. on Visualization and Computer Graphics (TVCG)*, 29(7):3281–3297, 2023. doi: 10.1109/TVCG.2022.3157058

[12] P. Fleck, D. Schmalstieg, and C. Arth. Creating IoT-ready XR-WebApps with Unity3D. In *The 25th International Conference on 3D Web Technology*, pp. 1–7. ACM, Virtual Event Republic of Korea, Nov. 2020. doi: 10.1145/3424616.3424691

[13] D. Fuentes, L. Correia, N. Costa, A. Reis, J. Barroso, and A. Pereira. SAR.IoT: Secured Augmented Reality for IoT Devices Management. *Sensors*, 21(18):6001, Jan. 2021. doi: 10.3390/s21186001

[14] A. Galizia, G. Zereik, L. Roverelli, E. Danovaro, A. Clematis, and D. D'Agostino. Json-GUI—A module for the dynamic generation of form-based web interfaces. *SoftwareX*, 9:28–34, Jan. 2019. doi: 10.1016/j.softx.2018.11.007

[15] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa, et al. Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning. In *Proc. IEEE Int. Conf. on Robotics & Automation (ICRA)*, pp. 5021–5028. IEEE, 2024.

[16] S. G. Hart and L. E. Staveland. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in psychology*, vol. 52, pp. 139–183. Elsevier, 1988. doi: 10.1016/S0166-4115(08)62386-9

[17] N. Hojo, K. Shinoda, Y. Yamazaki, K. Suzuki, H. Sugiyama, K. Nishida, and K. Saito. GenerativeGUI: Dynamic GUI Generation Leveraging LLMs for Enhanced User Interaction on Chat Interfaces. In *Proceedings of the Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pp. 1–9. ACM, Yokohama Japan, Apr. 2025. doi: 10.1145/3706599.3719743

[18] K. Huo, Y. Cao, S. H. Yoon, Z. Xu, G. Chen, and K. Ramani. Scenariot. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems - CHI '18*, Apr. 2018. doi: 10.1145/3173574.3173793

[19] R. Jaivignesh, R. D. Janarthanan, and V. Gnanalakshmi. Smart Home Automation using Augmented Reality and Internet of Things. *Journal of Physics: Conference Series*, 2325(1):012003, Aug. 2022. doi: 10.1088/1742-6596/2325/1/012003

[20] D. Jo and G. J. Kim. (PDF) AR Enabled IoT for a Smart and Interactive Environment: A Survey and Future Directions. *ResearchGate*, 19, Oct. 2019. doi: 10.3390/s19194330

[21] J. C. Kim, T. H. Laine, and C. Åhlund. Multimodal Interaction Systems Based on Internet of Things and Augmented Reality: A Systematic Literature Review. *Applied Sciences*, 11(4):1738, Jan. 2021. doi: 10.3390/app11041738

[22] U.-H. Kim, J.-M. Park, T.-j. Song, and J.-H. Kim. 3-d scene graph: A sparse and semantic representation of physical environments for intelligent agents. *IEEE Transactions on Cybernetics*, 50(12):4921–4933, 2020. doi: 10.1109/TCYB.2019.2931042

[23] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. In *Proc. Int. Conf. on Computer Vision (ICCV)*, pp. 4015–4026, 2023.

[24] S. Li, J. Chen, Y. Shen, Z. Chen, X. Zhang, Z. Li, H. Wang, J. Qian, B. Peng, Y. Mao, W. Chen, and X. Yan. Explanations from Large Language Models Make Small Reasoners Better, Oct. 2022. doi: 10.48550/arXiv.2210.06726

[25] Y. Liu, M. Sra, and C. Xiao. CrowdGenUI: Enhancing LLM-Based UI Widget Generation with a Crowdsourced Preference Library, Nov. 2024. doi: 10.48550/arXiv.2411.03477

[26] M. Mishakin, V. Andrushchak, J. Gazda, O. Kapshii, O. Karpin, and T. Maksymyuk. XR-Based Immersive User Interface for the IoT and Metaverse Applications. In *2023 IEEE 5th International Conference on Advanced Information and Communication Technologies (AICT)*, pp. 109–112, Nov. 2023. doi: 10.1109/AICT61584.2023.10452669

[27] Open Connectivity Foundation. Iotivity: An open-source framework implementing the ocf secure ip device framework. https://iotivity.org/, 2025. Accessed: 13 August 2025.

[28] X. Pan, G. Huang, Z. Zhang, J. Li, H. Bao, and G. Zhang. Robust collaborative visual-inertial slam for mobile augmented reality. *IEEE Trans. on Visualization and Computer Graphics (TVCG)*, 30(11):7354–7363, Sept. 2024. doi: 10.1109/TVCG.2024.3456152

[29] Y. Park, S. Yun, and K.-H. Kim. When IoT met Augmented Reality: Visualizing the Source of the Wireless Signal in AR View. In *Proceedings of the 17th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '19, pp. 117–129. Association for Computing Machinery, New York, NY, USA, June 2019. doi: 10.1145/3307334.3326079

[30] X. Qian, T. Wang, X. Xu, T. R. Jonker, and K. Todi. Fast-forward reality: Authoring error-free context-aware policies with real-time unit tests in extended reality. In *Proc. ACM Conf. on Human Factors in Computing Systems (CHI)*, CHI '24. Association for Computing Machinery, New York, NY, USA, 2024. doi: 10.1145/3613904.3642158

[31] S. Qiao, Y. Ou, N. Zhang, X. Chen, Y. Yao, S. Deng, C. Tan, F. Huang, and H. Chen. Reasoning with Language Model Prompting: A Survey, Sept. 2023. doi: 10.48550/arXiv.2212.09597

[32] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

[33] Z. Rashid, J. Melià-Seguí, R. Pous, and E. Peig. Using Augmented Reality and Internet of Things to improve accessibility of people with motor disabilities in the context of Smart Cities. *Future Generation Computer Systems*, 76:248–261, Nov. 2017. doi: 10.1016/j.future.2016.11.030

[34] F. Reyes-Aviles, P. Fleck, D. Schmalstieg, and C. Arth. Bag of world anchors for instant large-scale localization. *IEEE Transactions on Visualization and Computer Graphics (Proc. ISMAR)*, 2023. To appear.

[35] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi, A. Gupta, and L. Carlone. Kimera: From slam to spatial perception with 3d dynamic scene graphs. *The International Journal of Robotics Research*, 40(12-14):1510–1546, 2021.

[36] A. Sadeghi-Niaraki and S.-M. Choi. A survey of marker-less tracking and registration techniques for health & environmental applications to augmented reality and ubiquitous geospatial information systems. *Sensors*, 20(10):2997, May 2020. doi: 10.3390/s20102997

[37] D. Sahinel, C. Akpolat, O. Görür, and F. Sivrikaya. Integration of Human Actors in IoT and CPS Landscape. Apr. 2019. doi: 10.1109/WF-IoT.2019.8767276

[38] M. Schenkluhn, M. T. Knierim, F. Kiss, and C. Weinhardt. Connecting Home: Human-Centric Setup Automation in the Augmented Smart Home. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, pp. 1–16. Association for Computing Machinery, New York, NY, USA, May 2024. doi: 10.1145/3613904.3642862

[39] J. Strecker, K. Akhunov, F. Carbone, K. García, K. Bektaş, A. Gomez, S. Mayer, and K. S. Yildirim. MR Object Identification and Interaction: Fusing Object Situation Information from Heterogeneous Sources. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.*, 7(3):124:1–124:26, Sept. 2023. doi: 10.1145/3610879

[40] Y. Sun, A. Armengol-Urpi, S. N. Reddy Kantareddy, J. Siegel, and S. Sarma. MagicHand: Interact with IoT Devices in Augmented Reality Environment. In *2019 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*, pp. 1738–1743, Mar. 2019. doi: 10.1109/VR.2019.8798053

[41] S. Vashisht. Exploration of Markers in Augmented Reality: Methodologies and Applications. In *2024 Second International Conference on Intelligent Cyber Physical Systems and Internet of Things (ICoICI)*, pp. 1082–1086, Aug. 2024. doi: 10.1109/ICoICI62503.2024.10696631

[42] W3C. Documentation – web of things (wot). https://www.w3.org/WoT/documentation/, 2025. Accessed: 13 August 2025.

[43] B. Yalcinkaya, J. Aguizo, M. S. Couceiro, and A. J. Figueiredo. A multimodal tracking approach for augmented reality applications. In *Proceedings of the 12th Augmented Human International Conference*, pp. 1–8. Association for Computing Machinery, New York, NY, USA, May 2021. doi: 10.1145/3460881.3460929

8 Biography Section

Jan Kolberg is a PhD student at the Visualization Research Center (VISUS), University of Stuttgart, Germany. He received the B.Sc. and M.Sc. degrees in Mechatronics from the University of Stuttgart in 2021 and 2024, respectively. His research interests focus on combining Augmented Reality with Internet of Things technologies.

Michael Pabst is a PhD student at the Visualization Research Center (VISUS), University of Stuttgart, Germany, and the International Max Planck Research School for Intelligent Systems (IMPRS-IS). He received his B.Sc. degree in Vehicle Engineering at the University of Stuttgart, Germany, in 2019, and his M.Sc. in Robotics, Cognition, Intelligence at the Technical University of Munich (TUM), Germany, in 2024. His main research interests are Augmented Reality and 3D Scene Understanding.

Verena Biener is a researcher at the Visualization Research Center (VISUS) of University of Stuttgart. She received her PhD degree from Coburg University and the University of Bayreuth. Her research interests lie in the area of VR/AR and HCI.

Shohei Mori is a Junior Research Group Leader at the Visualization Research Center (VISUS), Cluster of Excellence IntCDC, University of Stuttgart, Germany, and serves as a Guest Associate Professor (Global) at Keio University, Japan. His research focuses on Mediated Reality and its enabling core technologies, including computational imaging and real-time graphics. He received his B.Eng. (2011), M.Eng. (2013), and D.Eng. (2016) from Ritsumeikan University, Japan. He was a Research Fellow for Young Scientists (DC-1 & PD) under the Japan Society for the Promotion of Science (JSPS), and a postdoctoral researcher at Keio University (2016–2018) and Graz University of Technology (2018–2024). He has received Best Paper and Demo Awards at premier conferences such as IEEE VR and IEEE ISMAR.

Dieter Schmalstieg Dieter Schmalstieg is Alexander von Humboldt Professor of Visual Computing at the University of Stuttgart, Germany, and adjunct professor at the Institute of Visual Computing at Graz University of Technology, Austria. His research interests are augmented reality, virtual reality, computer graphics, visualization and human-computer interaction.