

Gensors: Authoring Personalized Visual Sensors with Multimodal Foundation Models and Reasoning

Michael Xieyang Liu*
Google DeepMind
Pittsburgh, PA, USA
lxieyang@google.com

Alexander J. Fiannaca
Google DeepMind
Seattle, WA, USA
afiannaca@google.com

Savvas Petridis*
Google DeepMind
New York, NY, USA
petridis@google.com

Alex Olwal
Google Research
Mountain View, CA, USA
olwal@acm.org

Vivian Tsai
Google DeepMind
Mountain View, CA, USA
vivtsai@google.com

Michael Terry
Google DeepMind
Cambridge, MA, USA
michaelterry@google.com

Carrie J. Cai
Google DeepMind
Mountain View, CA, USA
cjcai@google.com

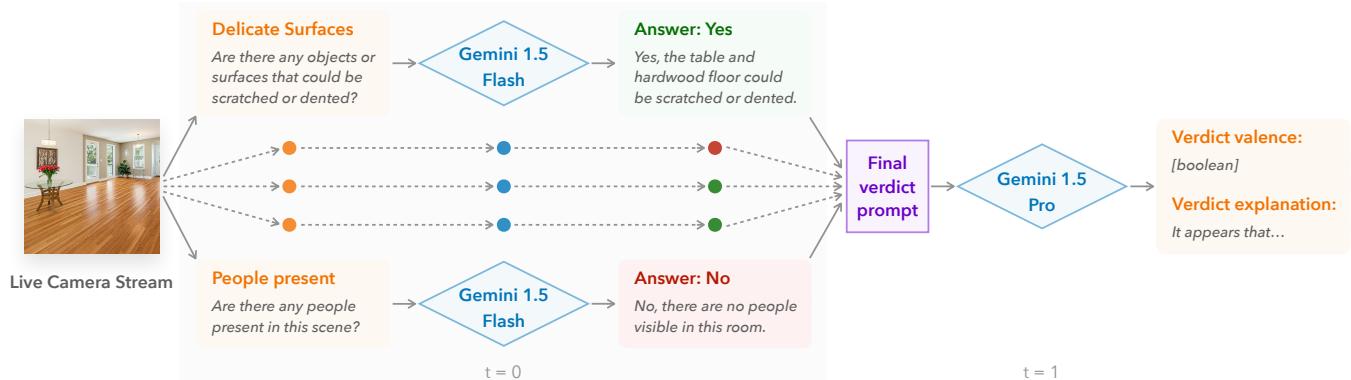


Figure 1: Gensors introduces a novel workflow to define and refine criteria for a sensor to monitor a live camera stream, analyze it using an MLLM, and provide a final verdict based on user-configurable logic and examples. The system uses a two-stage pipeline, where Gemini 1.5 Flash (optimized for speed) is called for all criteria to produce a collection of answers ($t=0$), which are subsequently sent to Gemini 1.5 Pro (optimized for reasoning capabilities and support for large context window), which is prompted to provide a final verdict ($t=1$).

Abstract

Multimodal large language models (MLLMs), with their expansive world knowledge and reasoning capabilities, present a unique opportunity for end-users to create personalized AI sensors capable of reasoning about complex situations. A user could describe a desired sensing task in natural language (e.g., “let me know if my toddler is getting into mischief in the living room”), with the MLLM analyzing the camera feed and responding within just seconds. In a formative study, we found that users saw substantial value in defining their

own sensors, yet struggled to articulate their unique personal requirements to the model and debug the sensors through prompting alone. To address these challenges, we developed Gensors, a system that empowers users to define customized sensors supported by the reasoning capabilities of MLLMs. Gensors 1) assists users in eliciting requirements through both automatically-generated and manually created sensor criteria, 2) facilitates debugging by allowing users to isolate and test individual criteria in parallel, 3) suggests additional criteria based on user-provided images, and 4) proposes test cases to help users “stress test” sensors on potentially unforeseen scenarios. In a 12-participant user study, users reported significantly greater sense of control, understanding, and ease of communication when defining sensors using Gensors. Beyond addressing model limitations, Gensors supported users in debugging, eliciting requirements, and expressing unique personal requirements to the sensor through criteria-based reasoning; it also helped uncover users’ own

*Equal contribution.



This work is licensed under a Creative Commons Attribution 4.0 International License.
IUI '25, Cagliari, Italy

© 2025 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-1306-4/25/03
<https://doi.org/10.1145/3708359.3712085>

“blind spots” by exposing overlooked criteria and revealing unanticipated failure modes. Finally, we describe insights into how unique characteristics of MLLMs—such as hallucinations and inconsistent responses—can impact the sensor-creation process. Together, these findings contribute to the design of future MLLM-powered sensing systems that are intuitive and customizable by everyday users.

CCS Concepts

- Human-centered computing → Interactive systems and tools; Natural language interfaces.

Keywords

Human-AI Interaction, Foundation Models, Intelligent Sensing

ACM Reference Format:

Michael Xieyang Liu, Savvas Petridis, Vivian Tsai, Alexander J. Fiannaca, Alex Olwal, Michael Terry, and Carrie J. Cai. 2025. Gensors: Authoring Personalized Visual Sensors with Multimodal Foundation Models and Reasoning. In *30th International Conference on Intelligent User Interfaces (IUI '25), March 24–27, 2025, Cagliari, Italy*. ACM, New York, NY, USA, 16 pages. <https://doi.org/10.1145/3708359.3712085>

1 Introduction

The proliferation of smart home technologies has popularized domestic sensing and monitoring [20, 21, 34, 75, 77]. Among these, smart camera sensors, offered by major consumer electronics companies, enable individuals to observe and react to events or activities within and around their homes, thereby enhancing safety and awareness. These devices leverage advances in computer vision (CV) and machine learning (ML) to identify and classify objects, distinguish between human and animal activities, or recognize specific events like package deliveries [64]. However, the current generation of smart camera sensors have limitations in their adaptability and configurability [51, 62, 76]. While capable of detecting predefined events, these systems do not readily support nuanced needs and priorities of individual users. This can lead to an excess of irrelevant notifications, missed critical events, and a mismatch between sensor capabilities, environmental conditions, and user needs.

In contrast to classical CV and ML models, recent multimodal large language models (MLLMs) integrate vast amounts of world knowledge and can understand and reason across a mixture of visual and textual inputs, much like humans [13, 38]. This advancement unlocks new opportunities for end-users to *define* personalized, MLLM-driven sensors capable of reasoning about complex situations (referred to hereafter as “AI sensors” for brevity)—a user could describe a desired sensing task in natural language, with MLLMs analyzing camera image frames and providing responses within seconds.

To illustrate, today’s standard CV/ML classifier-based sensors are typically trained on domain-specific data [27] and limited to *lower-level* specific sensing objectives (e.g. “are there unusual movements near the window”, “is there a pet in the kitchen”). In contrast, MLLM-driven AI sensors offer significantly greater flexibility with their ability to *reason* about *higher-level* problems (e.g. “alert me if my toddler is getting into trouble in the living room”, “is there something out of place in the backyard?”). These reasoning capabilities

may enable sensors to interpret the world in ways that would otherwise be challenging for traditional CV/ML classifiers. However, the seemingly infinite scope of tasks that can be accomplished with MLLMs also means that the semantic space of sensors users could define is also much broader than before, as they can now define tasks that are potentially more abstract, high-level, and *subjective* (e.g. “is my room messy?”).

In a formative study, we confirmed users’ desire for AI sensor personalization and configurability, but also discovered that the system must better support users in defining and communicating personal *criteria* and *constraints* (e.g. “is there something out of place in my room” [ignore the messy power cords]). This need was particularly pronounced for higher-level sensors with multiple possible semantic interpretations. For instance, a toddler “getting into mischief” might mean unrolling an entire roll of toilet paper in one household versus drawing on their sibling’s face in another. Additionally, users often struggled to generalize beyond their immediate, personal settings (e.g., their own room) and found it challenging to anticipate future scenarios and variants of those settings where their sensors might behave unexpectedly. These findings suggest that effective AI sensor definition requires systems to anchor *more* closely to users’ own criteria, while simultaneously making them aware of *other*, unanticipated scenarios.

To address these AI sensor definition opportunities and challenges, we developed Gensors, a system that empowers users to create and test real-time sensors powered by MLLMs. Beyond allowing users to create ad-hoc sensors via natural language prompting, Gensors leverages the reasoning capabilities and world knowledge of MLLMs. Specifically, users can (1) ask Gensors to break down the sensing problem into **automatically-generated relevant criteria**, (2) **manually define their own criteria**, and (3) **test and debug** multiple criteria simultaneously in real time. Furthermore, users can (4) ask Gensors to **generate new criteria not yet considered** based on frames from their video stream as positive and negative examples and also (5) **“future-proof”** their sensors by asking Gensors to suggest future situations that may lead to model failures, along with actionable tests for the user to try. Finally, users can configure how the final verdict is determined when considering the collective criteria, with options for an LLM-generated decision or rule-based combinations using boolean logic.

In a user study with 12 participants, we compared Gensors to a baseline condition where users iterated on a single prompt without structured assistance. We found that Gensors significantly increased users’ sense of control, understanding, and communication with the model. Specifically, Gensors enabled participants to decompose the AI sensor definition problem into lower-level criteria and explore them in parallel, granting them greater control and systematic insights. Additionally, Gensors’ reasoning capabilities also helped offset users’ own limitations and “blind spots,” by making them aware of potential failure modes and edge cases, and surfacing context-specific criteria they hadn’t considered. Finally, our study also highlighted how certain idiosyncrasies of MLLMs, such as hallucinations or “flickering” textual responses, impact sensor performance and perceived reliability.

Notably, participants used the Gensors tools for purposes *beyond* addressing model limitations: by enabling users to decompose sensors into bite-sized criteria and test them, Gensors helped users to

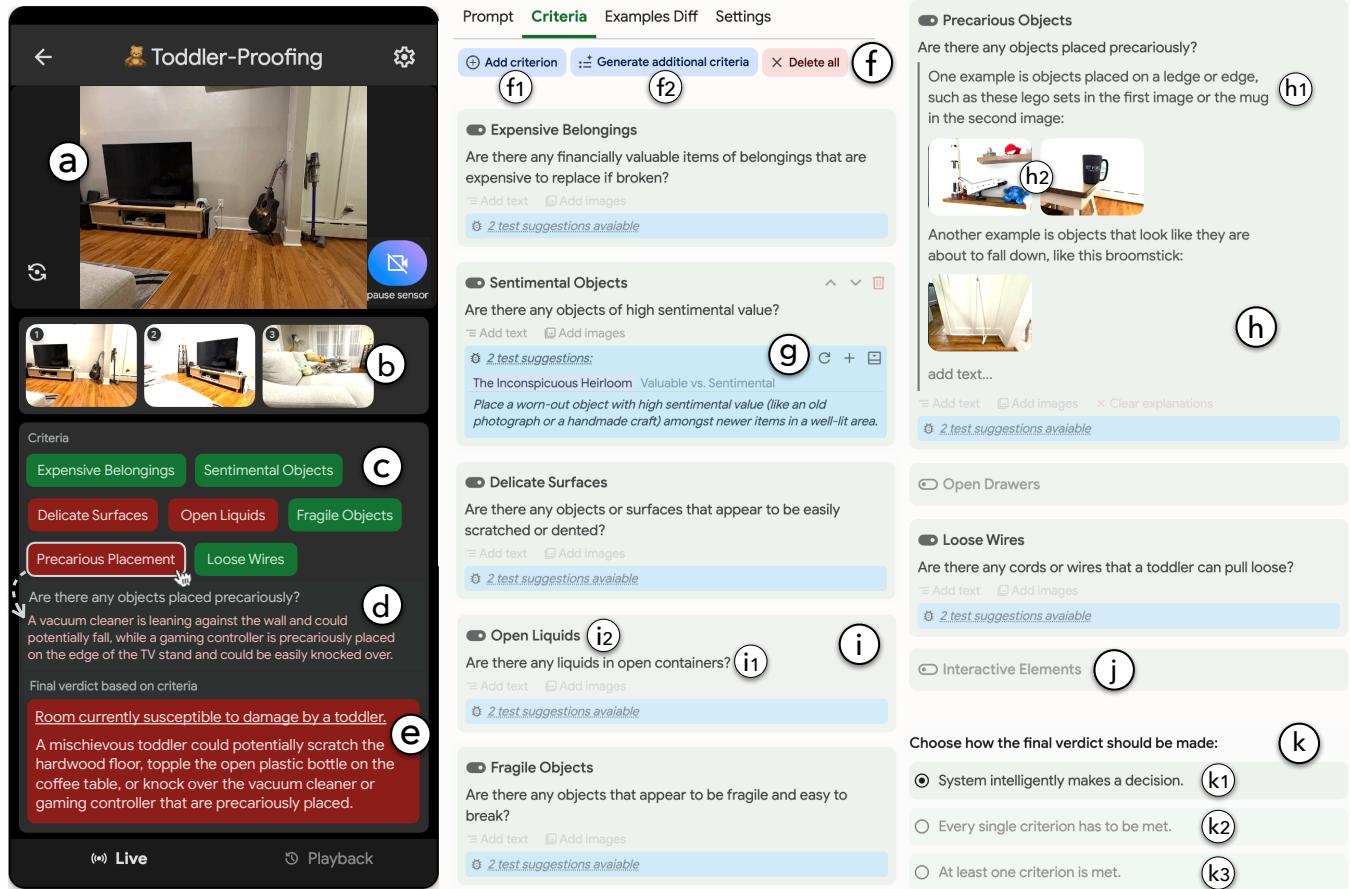


Figure 2: Gensors’ main user interface for formulating and curating criteria that govern a sensor’s behavior. For a high-level sensing task (in this case “tell me if my toddler might damage something”), the system operates as follows: At each time interval, the sensor evaluates all criteria (c) simultaneously using frames captured by a live camera (a) over the past few seconds (b). The result for each criterion is displayed as a green or red chip, indicating a positive or negative outcome, respectively. Users can click on a chip to view detailed results for that specific criterion, which is the description from the MLLM’s interpretation of the scene (d). The sensor then synthesizes these individual results to make an informed final decision regarding the original sensing task (e). In the criteria editor (f), users have the option to either add their own criterion (f1) or have Gensors automatically generate criteria (f2) based on the initial sensing task and the live camera view. To modify an existing criterion (i), users can update its description (i1), and Gensors will automatically generate a title (i2) for display in the Live sensor view (c). Furthermore, users can 1) add additional text (h1) or visual examples (h2) to explain a criterion (h) based on their personal context; 2) review Gensors-generated suggestions for testing a specific criterion (g); 3) temporarily enable or disable a criterion (j); and 4) configure how the sensor reaches its final verdict (k), ranging from allowing the MLLM to make an intelligent decision based on all criteria results (k1), to rule-based combinations using boolean logic (k2, k3).

more easily elicit their own personal requirements, debug the sensor through isolating sub-components of logic, and better understand the model’s capabilities and limitations. These needs will likely persist even as models continue to improve in the future. Collectively, these findings contribute to the design of future user-defined sensing systems supported by MLLM-powered reasoning.

Contributions. This paper makes the following contributions:

- **Formative study** (n=6) confirming the potential and desire for end-user AI sensor definition, alongside challenges with open-ended prompting.

- **Design goals for AI sensor definition:** support for control over sensors’ behavior, criteria elicitation, expression of personal constraints, and systematic testing and debugging.
- **The Gensors system** for more effective AI sensor definition through automatic and manual generation of criteria, debugging of individual criteria, example-driven specification, and actionable test cases.
- **Formal user study** (n=12) finding that Gensors significantly increased users’ sense of control, understanding, and communication with the model, through enabling users to focus more on formulating and debugging sensor requirements rather than on prompt phrasing. Beyond this, Gensors’ reasoning capabilities

also helped users consider failure modes beyond their own immediate situations, and helped users address their own “blind spots” by exposing criteria they didn’t already think of. Finally, we discovered how MLLM idiosyncrasies (“flickering” and hallucinations) affected the sensor creation process.

2 Related Work

2.1 Intelligent and DIY Visual Sensing

The concept of general-purpose, do-it-yourself (DIY) sensing has long been considered the ultimate goal of ubiquitous computing, particularly in smart home environments [20, 21, 34, 75, 77]. Many have aspired for sensing systems that end-users can intuitively customize [36, 54], yet current commercial smart home sensors still fall short of this vision. While affordable and accessible, they are typically highly specialized [29, 32, 67], and require users to invest significant time and effort in learning and creating custom workflows based on their outputs [70]. Moreover, these sensors usually produce low-level data that cannot directly answer users’ high-level questions [12, 18]. For example, a door sensor might indicate the door’s open/close status but cannot explicitly inform users whether their children have left or arrived home [35].

Much prior research has been directed at closing this gap between *what can technically be sensed* and *what users are actually interested in knowing* [12, 39], particularly in the visual sensing domain. Earlier work focused on leveraging “human intelligence” through online marketplaces such as Amazon Mechanical Turk [1]. For example, VizWiz [4] and VizLens [16] had crowd workers answer visual questions of photos taken from smartphones. Follow-up efforts, like VATIC [74] and Flock [9], utilize crowd-labeled data to subsequently train ML models. These efforts culminated in the Zensors system [17, 35]: initially, Zensors relies on human intelligence to directly answer users’ sensing questions, such as “Is there parking spots available?” or “How orderly is the line?” Over time, it uses this human-labeled data to train CV models, ultimately automating the sensing task by replacing human input with model predictions.

Though providing a general-purpose sensing solution, Zensors’ effectiveness and applicability remains constrained by the capabilities and limitations of traditional CV models. Additionally, users may struggle to debug or customize sensors [35], especially when outputs differ from expectations. In this work, we explore the potential of replacing crowd-ML hybrid sensing backbone with MLLMs, offering two distinct advantages: 1) MLLMs, with their reasoning capabilities, can directly explain their *thought processes*, aiding users in sensor debugging and understanding system capabilities and limitations, unlike previous methods where the rationale used by crowd workers is obscured by the later CV model; 2) MLLMs handle multimodal inputs, allowing users to define sensors using not only natural language but also direct visual examples, offering greater flexibility to express their unique personal contexts and needs.

2.2 LLM Prompting and Requirement Articulation

In the emerging paradigm of end-user-defined AI sensors, users are now responsible for ensuring their sensors operate as intended, largely through crafting effective prompts. Indeed, prompting has become crucial for crafting effective input instructions to guide

Large Language Models (LLMs) in generating desired outputs [2, 45, 48, 63, 78, 79], and has dramatically democratized and accelerated AI prototyping across various use cases and domains [22, 23, 25, 26, 37, 41, 46, 47, 57–59, 61, 65]. However, prompting remains a challenging and ambiguous task, particularly for users without technical expertise in LLMs. Common challenges include struggling with finding the right phrasing for a prompt, selecting appropriate demonstrative examples, experimenting with various hyper-parameters, and evaluating the effectiveness of their prompts [10, 22, 79]. Consequently, they may waste time on unproductive strategies, such as making trivial wording changes [53, 55]. This issue is further exacerbated when the task becomes more complex, involving multiple facets and requirements that need to be addressed within a single prompt [24, 81]. Similarly, we observed in the formative study that participants often haphazardly make ad hoc revisions to their sensing prompts in response to previous outputs, without a clear understanding of what needs improvement.

Akin to the concept of *requirement engineering* in software engineering [30], where humans define the desired outcomes and behavior of a program, often including expected inputs and outputs [33], recent research has shown that explicitly and clearly stating *requirements* within prompts is an effective strategy for systematically improving them [11, 49, 66]. However, articulating clear and complete requirements is a known challenge [19, 42, 43, 46, 52, 53], even for experts who need multiple iterations to refine them [68]. Poorly defined requirements frequently lead to program failures [10, 50]. While tools like EvalLM [28], SPADE [68], and EvalGen [69] have explored extracting user requirements from prompts to support prompt evaluation, there is limited emphasis on assisting users in effectively communicating requirements during prompt construction. In this work, we address this gap by prioritizing requirements as first-class entities in the form of *criteria* that govern the sensor behaviors. Rather than having end-users juggle the intricacies of prompt writing, we abstract these mechanics of raw prompt away from them, and instead direct their attention on defining criteria, thereby simplifying the task. Additionally, we offer user support through features such as auto-generating criteria based on common sense or user-provided visual examples, further assisting users in translating their personal contexts and preferences into criteria.

2.3 Interactive Model Refinement Through User Feedback

Prior work has explored various interactive systems that allow users to provide feedback to refine future model outputs. Programming-by-example tools enable users to provide input-output examples, with the system generating a function that fits these examples [8, 73, 80]. Similarly, recommender systems allow users to steer outputs through limited feedback [5, 44, 56], such as adjusting a 2D plane to influence movie recommendations [31]. Teachable Machines also offer an interactive approach, allowing users to train ML models by supplying labeled examples, with real-time feedback facilitating iterative refinement [7]. More recently, systems and methods like ConstitutionMaker [61] and ConstitutionalExperts [60] leverage LLMs to translate natural language feedback and critique into high-level *principles* (similar to the sensor criteria in our work), enabling conversational, human-like interactions to

steer the model. However, previous approaches often integrate natural language principles directly into prompts to steer model outputs [46, 61], making it difficult for users to understand how individual principles perform or how editing one principle might inadvertently affect the efficacy of others. In contrast, in Gensors, each criterion targets a single, specific aspect of the overall problem. Users adjust one criterion at a time and receive results specific to that criterion without having to disentangle insights from a mix of different factors bundled into a larger prompt response, and without fearing that adjustments might affect other criteria.

3 Formative Study & Design Goals

To understand the opportunities and potential challenges for user-specified AI sensors, we conducted a formative study with six professional designers (age range: 29–42, 3 female and 3 male) from a large technology company, where they were asked to brainstorm AI sensor use cases and create their own sensors with a prompt-based prototype. Based on the findings from this study and insights from prior research, we identified a set of design goals for Gensors.

3.1 Setup

3.1.1 Procedure. The overall outline of the formative study was as follows: (1) Participants spent 5 minutes individually brainstorming potential use cases for visual-based personal sensors, documenting their ideas and initial thoughts. (2) Participants were then shown the prompt-based prototype they would be using to build their sensors (see Figure 3). (3) Participants spent 40 minutes individually creating one to two sensors of their choice. While they prototyped, they were asked to take notes on how they iterated over their prompt and take screenshots. To ensure ecological validity, participants were asked to create sensors in their homes. (4) For the last 10 minutes, the facilitators led a group discussion with the participants to learn about their experience building sensors.

3.1.2 Prototype. The prototype (Figure 3) provided a basic interface for creating MLLM-powered sensors. The prototype allowed them to input prompts (Figure 3-d) and set an interval for execution using their laptop camera. The prototype displayed the MLLM's output for each execution (Figure 3-e), as well as the corresponding input image (Figure 3-c). Finally, participants could view the history of all the sensor inputs and outputs.

3.2 Findings

3.2.1 The opportunity for personal AI-powered sensors. All participants were quite excited about the possibility of building their own personal sensors. P3 explained, “Traditionally, I had to buy a product built for a task...being able to set up my own sensor - that’s the new part.” He then went on to describe that normally the company selling the sensor determines how the sensor behaves, but now he is able to create and customize the sensor to his own requirements.

Participants brainstormed a total of 47 personal sensors they would use in their daily lives. They ideated *reminder* sensors which would detect the last time they watered the plants, exercised, or took their vitamins. There were also *aggregative* sensors that detected: “how much family time are we spending,” “how much time did I spend practicing Korean,” “what food have I eaten during the day to manage health conditions.” In addition to these more reflective

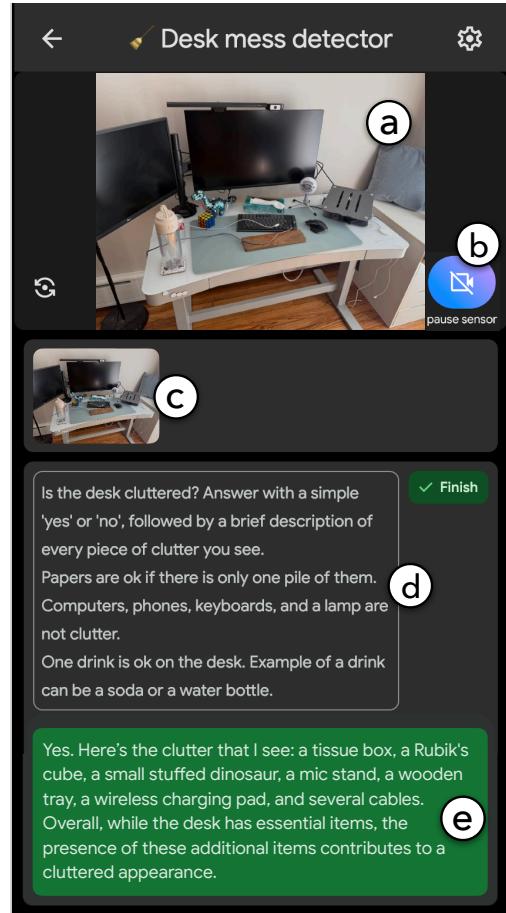


Figure 3: The prompt-editor authoring tool participants used in the formative study, as well as the baseline condition in the user study (see Section 5). Users can view what their camera currently sees (a) and pause and play the sensor (b). They can edit their prompt (d) and then view its latest result (e) as well as the corresponding input image it was run on (c).

sensors, participants also had many ideas for more *urgent* sensors, including: “let me know if my landlord is at the door,” “tell me if my pork chop is getting burnt,” and “tell me if there’s a leak at my water heater.” Finally, participants also had ideas for sensors that detected *new events*, such as “how many new people did I meet last week” and “tell me when I see something I have never seen before.”

3.2.2 Challenges in creating prompt-powered sensors. A common strategy participants employed to steer their sensors was to write criteria to specify when the sensor should output a label. For example, P6 built a sensor to determine when his plant needs fertilizer. Their prompt was initially: “Does this plant need fertilizer? Explain why.” The outputs from this prompt were a bit vague and seemed unjustified, like: “No. The plants look healthy.” So, they then appended the criterion: “Look for signs of yellowing and discoloration” which steered the model toward producing more relevant and informative explanations. Similarly, P5 was building a sensor that detects if any chores needed to be done in her living room, and to further steer the sensor, she appended a criterion to

check for any trash in the bins. By adding these criteria, participants were able to guide the model toward better decisions and explanations from their sensors.

However, steering the model using criteria presented several challenges. First, as participants added more criteria and clauses to their prompts, they became unwieldy; it became evident that the model neither adhered to nor explicitly checked all the specified criteria. Additionally, these convoluted, multi-clause prompts left participants uncertain about how to modify their criteria or adjust the prompt as a whole. For this reason, our first design goal (**D1**) was to support more **precise control over a sensor's behavior** and enable users to incorporate and test their criteria individually.

Identifying useful criteria was another significant hurdle for participants. For instance, P2, who built a sensor to detect if he was eating unhealthy snacks, struggled to define a comprehensive set of qualities that characterized the snacks they personally considered unhealthy. Therefore, our second design goal (**D2**) was to **accelerate requirement elicitation** by offering users an initial set of common-sense criteria to peruse and refine.

Next, participants sometimes struggled to articulate particular criteria in their prompts. For example, P3 made a sensor to detect if his living room was messy. He had a few decorative items on the couch, including a few unique pillows and a throw blanket featuring a pizza design. In his prompt, P3 specified the criterion: “look for a remote or wrappers on the couch, not the pillows or blanket.” However, despite adjusting the wording of this criteria, the sensor continued to classify the couch with the throw blanket as messy; P3 expressed a desire to illustrate this particular criterion with an image, emphasizing that the distinctive throw blanket should not be considered part of the mess. This led to our third design goal (**D3**): to provide **flexible ways for users to communicate their personal context**, whether by enabling them to use images to clarify criteria or by helping them verbalize more nuanced conditions.

Finally, two participants expressed doubts in their sensors' future performance. P3 described experiencing a sense of “tunnel vision” with his tests, as his experiments were limited to removing and adding nearby objects to the couch and coffee table in his living room. It was hard for P3 to step back from his immediate physical context and envision realistic changes his living room might undergo over time. Thus, our final design goal (**D4**) was to **scaffold testing** and support users in future-proofing their sensors.

3.3 Summary of Design Goals

In summary, we postulate that an effective system that helps end-users create flexible and intelligent AI sensors should support:

- **D1: Enabling precise control over the sensor behavior via fine-grained criteria.** Users should be able to author and adjust criteria individually and assess the sensor's performance for each one, without needing to revise an entire prompt.
- **D2: Accelerating requirement elicitation by bootstrapping common-sense criteria.** The system should assist users in getting started by generating relevant, common-sense criteria automatically.
- **D3: Providing flexible ways to communicate personal context.** Users should be able to express their specific context via

text, as well as visually. The system should also help users identify their more nuanced criteria.

- **D4: Scaffolding testing and debugging of criteria.** The system should offer tools that allow users to test, isolate, and debug each criterion separately, enabling users to address future scenarios that might confound their sensor, incrementally improve the sensor's accuracy, and understand how each criterion impacts overall performance.

4 The Gensors System

We begin by presenting a usage scenario that demonstrates the core functionalities of Gensors. This example incorporates key use cases identified in our formative study, along with specific criteria curated by participants in the subsequent user study.

4.1 Example Usage Scenario

Emma was concerned about her toddler “getting into trouble” in her home, as she has often observed him playing with her purses or breaking her valuable belongings. She couldn't find any commercial sensors specifically designed to monitor this sort of situation, so she decided to create a custom one using the Gensors platform. To get started, Emma entered her request as “tell me if toddler might damage something” into the system. Gensors automatically generated a basic LLM prompt based on that request: “Is the toddler likely to damage something valuable in this room? Answer with ‘Yes’ or ‘No’, and provide a brief explanation.” Emma then positioned her webcam to provide a clear view of her living room space where her toddler frequently played. Within seconds, the custom “Toddler Check” sensor was operational: it is configured to run every three seconds (a default frequency easy for users to test and debug the sensor prompt) using the latest three frames (Fig. 2-b) from the camera feed (Fig. 2-a, which takes one frame per second) and the prompt mentioned previously, providing continuous assessment of the room.

However, Emma quickly realized that the sensor considered her stack of clothes damageable, even though she is fine and well-acquainted with her toddler playing with her laundry. The sensor also did not notice her sacred wedding photo that her toddler often attempts to reach. Despite her efforts to tweak and add more clauses, it was unclear if these adjustments were consistently improving the responses. As a result, Emma clicked the “Generate criteria” button (Fig. 2-f2), and Gensors automatically produced several criteria (e.g., Fig. 2-i) based on the initial request and the environment (e.g. “Delicate Surfaces,” “Sentimental Objects,” “Fragile Objects”). To try out these criteria, she could view the real-time results for each criterion as small green or red chips in the sensor's Live View (Fig. 2-c), clearly indicating whether the room passed or failed at each criterion check. Emma could click on any chip to read a brief explanation, helping her understand what the model “saw” in the image (Fig. 2-d). She realized that the sensor was consistently missing “Precarious Objects,” so she modified the criterion to include a few image examples, such as objects placed on a ledge or edge or those that could potentially fall (Fig. 2-h). She also added her own “Open Liquids” criteria when she saw the open water bottle on the coffee table.

Later, Emma found it challenging to articulate the concept of “books the toddler shouldn't tear up” in words, because it was hard

to verbally distinguish between her own books from the toddler's children's books, so she tried the Example-Diff feature (See Fig. 4). She pointed the webcam to capture these items, then from the Playback view (that contains the history of all sensor runs), she selected image frames from that earlier webcam footage: a set of her books, and a set of children's books. To Emma's surprise, the system then suggested additional criteria she hadn't considered. For instance, it noted that adult books were less likely to have large pictures on the covers and often featured paper dust jackets. Emma incorporated these into her existing set of criteria, recognizing that the system had highlighted aspects she might have otherwise overlooked.

4.2 System and User Interface Design

We now discuss how the various Gensors features are designed and implemented to support the design goals. Gensors serves as a platform for end-users to create and experiment AI sensors powered by multimodal foundation models. These sensors are therefore hardware-agnostic, and can eventually be deployed on any existing hardware with a camera feed. It is important to note that challenges related to actual sensor deployment and maintenance fall outside the scope of this work.

4.2.1 D1: Enable precise control over the sensor behavior via fine-grained criteria. In Gensors, a criterion functions as an atomic unit of reasoning, and is essentially a minimal prompt that targets a single, specific aspect (e.g., “Are there any open outlets that are not properly covered?”) of the overall sensing problem (e.g., “Is this room safe for a toddler?”). This level of granularity helps prevent MLLMs from overlooking details from the input image frames, a common challenge identified in our formative study and corroborated by previous research on complex or multi-faceted prompts [45]. In addition, by isolating individual criteria, users can more easily interpret results (Fig. 2-d) without having to disentangle insights from a mix of different factors bundled into one larger prompt response, which can be cognitively demanding.

Creating a new criterion is as straightforward as writing a question in natural language, lowering the barrier for users without technical expertise to start making sensors. Users can create a criterion by clicking the “Add Criterion” button at the top of the Criteria tab (Fig. 2-f1). Once they type in their question, the system automatically generates a concise name for the criterion, making it easy to identify and parse among other criteria when viewing results in the Live view (Fig. 2-c).

Once a criterion is created, Gensors evaluates it against the scene during subsequent runs. Specifically, we instruct the MLLM through system instructions to always generate a brief description regarding the criterion, accompanied by a valence that reflects the semantic outcome of the evaluation. This valence, indicating whether the scene has passed the check or if issues have been detected, is visually represented in the Live view using color-coded chips: green signifies a positive outcome *with no issues*, while red indicates a negative one *with potential issues that require user attention*.¹ For instance, regarding the question “Are there any objects placed precariously?” (Fig. 2-d), if the model indeed identifies objects that are placed precariously (such as those mentioned in the Example

Usage Scenario, see Fig. 2-h), the valence will be negative (hence red criterion chip). Conversely, if the model considers all objects to be properly placed, the valence will be positive (hence green criterion chip). These color-coded chips enable users to quickly assess a sensor’s behavior at a high-level, and detect potential issues at a glance (Fig. 2-c). If a user notices something out of the ordinary—such as a red chip indicating an unmet condition—they can click on the chip to access a more detailed explanation of the issue (Fig. 2-d).

Furthermore, each criterion operates independently from others, ensuring that changes to one do not inadvertently affect the performance of others, unlike the behavior observed when iterating on a single, large prompt in the formative study (Fig. 3). Users also have the option to deactivate a criterion if they wish to suppress its results temporarily. Once satisfied with the performance of a particular criterion, they can proceed to the next, resulting in a more systematic and tractable debugging process.

At each time interval, all active criteria are executed in parallel, each yielding its own result. These individual results are then aggregated and fed into a subsequent prompt, which is tasked with reasoning through each criterion result and then formulating an informed final verdict accompanied by an explanation. Our informal testing suggests this divergent-then-convergent approach effectively ensures the sensor considers all the aspects that the user specified, while intelligently analyzing and prioritizing these aspects based on common sense (Fig. 2-k1). Alternatively, users have the flexibility to instead use Boolean logic for the final verdict. For instance, they can configure the system to require all criteria to be met for a positive outcome (AND) (Fig. 2-k2) or allow a positive verdict if at least one criterion is satisfied (OR) (Fig. 2-k3).

4.2.2 D2: Accelerating requirement elicitation by bootstrapping common sense criteria. To further reduce user friction, Gensors can also automatically generate relevant criteria (Fig. 2-f2) by leveraging the extensive world knowledge and reasoning capabilities of today’s foundation models. Specifically, we prompted the model to generate criteria that are contextually grounded in the user’s sensing task and environment, mirroring what a human would typically use to evaluate the task. These auto-generated criteria serve a useful starting point, particularly for users less unfamiliar with the intricacies of their tasks. Unlike previous systems that aimed for comprehensive coverage upfront (e.g., Selenite [47]), we intentionally limit the number of criteria generated per turn to four. This strategy avoids overwhelming users with too many suggestions at once, thereby maintaining their cognitive bandwidth and agency to define criteria based on their unique personal context. Users still can, however, generate additional sets of criteria, each guaranteed to differ from the existing ones, empowering them to explore as many perspectives as they wish without being locked into a single set of recommendations.

4.2.3 D3: Provide flexible ways to communicate personal context. In our formative study, participants often found it challenging to define criteria precisely using only natural language. Often, a more intuitive approach for them was to provide concrete visual examples, such as annotated images, in combination with textual descriptions, hoping that the system would infer the intended meaning more effectively. To address this challenge, Gensors enables users to communicate their criteria through not only text but also images

¹We acknowledge the accessibility concerns for color-blind users with this color scheme and plan to address them in future system iterations.

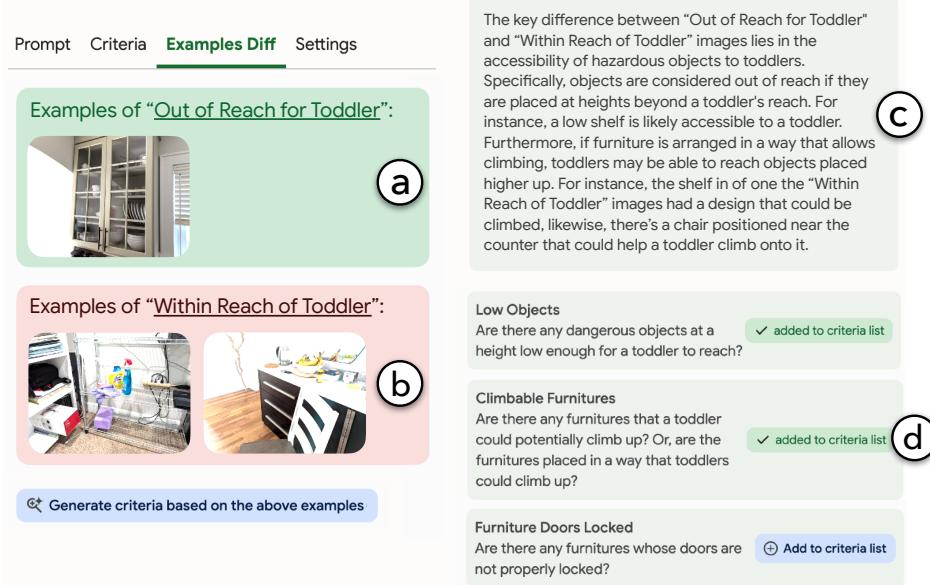


Figure 4: The Examples-diff feature in Gensors allows users to provide visual examples for each answer category (a and b) related to the high-level sensing task. Gensors then presents its reasoning process (c) and any additional criteria generated from that reasoning (d). Users can choose whether to incorporate these criteria into the main list.

and annotations that represent their expectations when needed (Fig. 2-h). This flexibility empowers users to select the modality they naturally gravitate towards depending on the circumstances, and can be particularly useful when describing visual or abstract features. Behind the scenes, Gensors combines all content under a single criterion together to steer the MLLM’s understanding and subsequent responses.

Furthermore, it could often be the case where users have an abstract idea of what they want a sensor to monitor but struggle to articulate these ideas into specific criteria. To address this, Gensors offers the Examples-Diff feature, which transforms labeled images into clear, actionable criteria (Fig. 4). For each sensing task, Gensors automatically generates distinct categories of possible answers based on the original sensing task, while also allowing users to customize these categories to their liking. Users can then select representative image frames from the sensor’s history to illustrate each category (Fig. 4-a&b). Behind the scenes, we leverage the visual reasoning capabilities of MLLMs, and instruct the model to first “reason through the provided images and think carefully about their differences as well as the subtleties the user is trying to convey through these examples” and then generate “actionable criteria that are absent from or inadequately represented by existing criteria.” We present the model’s reasoning process (Fig. 4-c) and the newly generated criteria (Fig. 4-d) to the user, who can review and decide whether to incorporate them into the main criteria list.

This feature is especially useful in boundary cases where it is challenging for users to manually differentiate and develop effective criteria for the model to understand and process. Additionally, when users struggle with inspiration for criteria, they can rely on the Examples-Diff feature to capture potentially missing details and ensure thoroughness. Currently, the Examples-Diff feature is optimized for binary classification sensing tasks, such as “Is my

The key difference between “Out of Reach for Toddler” and “Within Reach of Toddler” images lies in the accessibility of hazardous objects to toddlers. Specifically, objects are considered out of reach if they are placed at heights beyond a toddler’s reach. For instance, a low shelf is likely accessible to a toddler. Furthermore, if furniture is arranged in a way that allows climbing, toddlers may be able to reach objects placed higher up. For instance, the shelf in one of the “Within Reach of Toddler” images had a design that could be climbed, likewise, there’s a chair positioned near the counter that could help a toddler climb onto it.

Low Objects

Are there any dangerous objects at a height low enough for a toddler to reach? added to criteria list

Climbable Furnitures

Are there any furnitures that a toddler could potentially climb up? Or, are the furnitures placed in a way that toddlers could climb up? added to criteria list

Furniture Doors Locked

Are there any furnitures whose doors are not properly locked? Add to criteria list

desk cluttered?” It generates meaningful “positive” and “negative” classes, like “Cluttered desk” vs. “Uncluttered desk.” Handling other types of questions and allowing users to more freely add, remove, or customize these categories can be addressed in future work.

4.2.4 D4: Scaffold testing and debugging of criteria. Similar to best practices in software engineering, one important consideration when creating robust sensors is for users to test their sensors, ideally with edge cases that could lead to potential failures. Therefore, for each criterion, Gensors automatically generates two suggested test cases, with the flexibility to re-generate or produce more suggestions on demand (Fig. 2-g). Here, we again leverage the reasoning capabilities of foundation models, and instruct the model to first “reason about how it might be challenging to assess a particular criterion based on different situations and scenarios that the user might encounter when using the sensor,” and then generate “test cases that are practical for users to try and test.” This extends beyond common prompt engineering, which often focuses on adjusting the format and tone of the output based on a given input. Instead, users are empowered to actively manipulate their environment—for example, by introducing new foreign objects or altering spatial configurations—in order to “future proof” their sensors. They can observe the model’s responses and make targeted iterations on the criteria. The generated test suggestions are presented as expandable chips for users to view and engage with.

It is worth noting that, earlier in development, we explored generating broad, top-level test suggestions based on the initial sensing task. While useful, informal pilot testing revealed that these suggestions, were oftentimes too generic and lacked direct relevance to users’ specific criteria (e.g., testing under various lighting conditions). Instead, we discovered that tailoring test suggestions to each individual criterion yielded more actionable and effective results. This approach sparked deeper insights into the model’s capabilities

and limitations, facilitating users in refining their criteria more effectively and fostering a more robust testing process.

4.3 Technical Implementation

The Gensors web platform is developed using HTML, TypeScript, and CSS, utilizing the Lit Web Components library [40] for building UI elements. A Python backend was implemented to manage LLM calls and additional API requests. All sensor data is stored locally using the browser's IndexedDB [3].

Many of Gensors features are powered by the latest multimodal Gemini models as of October 2024. Specifically, for running individual sensor criteria, we utilize the Gemini 1.5 Flash version [14] to ensure near-instant response times (see Fig. 1). We set the temperature to 0 to minimize output randomness. For tasks related to criteria creation (e.g., automatically generating criteria and the examples-diff feature) as well as making the final verdict that aggregates results from individual criteria, we employ the Gemini 1.5 Pro version [15] for its advanced reasoning capabilities and support for long context window (see Fig. 1). To encourage creativity in these tasks, we set the temperature to 0.8. Users have the flexibility to customize these default model configurations via a sensor's settings page.

However, it is important to note that our primary contributions lie more in the *concept of scaffolded requirement elicitation and the design of user interface and experience for creating LLM-powered sensors*, which are independent of specific model usage. We anticipate that these designs will remain relevant as generative AI models continue to advance in the near future.

5 User Study

To gather insights into Gensors' potential to benefit the AI sensor specification process, we conducted a 12-participant within-subjects user study. The study compares Gensors to a prompt-editor version, similar to the one used in the formative study (Figure 3), where participants specified AI sensor behavior by iterating on a text prompt. We also included Gensors' playback feature in the baseline condition, where participants could view the history of sensor outputs with their inputs.

5.1 Procedure

The overall outline of the study is as follows: (1) Prior to the study, participants completed a 30-minute self-directed tutorial, where they watched instructional videos and built a sensor with Gensors and the prompt-editor version. (2) During the study, participants spent 50 minutes creating two AI-powered sensors, starting either with Gensors (25 minutes) or the prompt-editor version (25 minutes), in a counterbalanced design. (3) After building these two sensors, participants completed a post-study questionnaire, which compared the two sensor prototyping conditions. (4) In a semi-structured interview, participants gave feedback on each prototyping tool, including the benefits and drawbacks of each one. The total time commitment of the study was 90 minutes.

From the brainstorm conducted in the formative study, we picked two different types of sensors for participants to implement in the user study, one “urgent” and one “reminder” sensor. The two sensors were: (1) a “reminder” *desk clutter* sensor which determines when the user’s desk is messy and (2) an “urgent” *toddler safety*

sensor which determines if there is anything particularly dangerous to a toddler in the user’s bedroom. We chose these two sensors as they are realistic use cases of personal sensors, as well as general enough for users to be able to define and have personal opinions about them. The order in which participants implemented these provided sensors was counterbalanced, in addition to the condition order. To help situate the task, we asked participants to imagine that they were the target user for each sensor. For the desktop clutter sensor, participants were asked to imagine that they were a professional who worked from home and was creating a sensor that would send a reminder for them to organize their desk during the work week if it got too messy. For the toddler safety sensor, participants were asked to imagine that they were a new parent, building a sensor to help identify conditions in their living or bedroom which might be problematic for their toddler.

Finally, participants authored and tested their sensors generally via their laptop camera, though some were able to use a webcam focused on the area of interest and author the sensor separately on their laptop. The study was approved by our institution’s IRB.

5.2 Participants

We recruited 12 participants (6 female, 6 male) from our institution, covering a range of skill sets, including product managers, UX researchers, and software engineers, and from a variety of locations in the US, including Michigan, New York, California, Georgia, Wisconsin, and Washington. We were generally recruiting for “first-adopter” tech-savvy individuals who would be likely candidates for authoring AI-powered sensors in their own home. Participants were recruited via an email invitation. The study was conducted remotely, in participants’ homes to create a valid testing environment. Participants received a \$40 gift card for their participation.

5.3 Questionnaire

We wanted to understand the potential for Gensors to help participants think through their requirements for the sensor and steer the model to follow them. As such, our questionnaire (Table 1) probes participants’ self-perceived *control* over the sensor, as well as their perceived ability to *think through and communicate their personal requirements*. We were also interested in seeing how useful Gensors’ *test suggestions* were and if testing via modular criteria helped participants gauge the *underlying model’s capabilities*. We also included questions to assess the relative usefulness of Gensors’ features.

5.4 Data Analysis

To analyze the results from the post-study questionnaire, we conducted paired sample Wilcoxon tests with full Bonferroni correction to compare the ratings from the two conditions, since the study was within subjects and the questionnaire collected ordinal data.

In addition, the main study sessions and the post-study interviews were screen and audio recorded, then transcribed. The first two authors independently coded the recordings and transcriptions using an open coding approach [72] in accordance with Braun and Clarke’s thematic analysis [6]. Subsequently, they iteratively resolved disagreements and ambiguities, which included periodic discussions with the research team. We present the key themes and findings below.

Metrics (Both Conditions)	Statement (7-point Likert scale)
Control	With Prototype {A, B}, I felt I had control creating with the system.
Understand Capabilities	Prototype {A, B} helped me understand the underlying model's capabilities – i.e. what it could and could not detect.
Communicate Requirements	With Prototype {A, B}, I felt I was able to think through and communicate my personal requirements for the sensor.
Test	With Prototype {A, B}, I was able to test and probe the sensor.
Gensors Metrics	Statement (5-point Likert scale)
Manual Criteria	How helpful was: The tool where I could make my own criteria toward helping you accomplish your goals?
Automatic Criteria	How helpful was: The tool that automatically generated criteria toward helping you accomplish your goals?
Example-Diff	How helpful was: The tool that generated criteria based on positive/negative examples (“Examples-Diff”) toward helping you accomplish your goals?
Multimodal Criteria	How helpful was: The tool that let you provide images as examples for criteria toward helping you accomplish your goals?
Test Cases	How helpful was: The tool generated test cases for criteria toward helping you accomplish your goals?

Table 1: Post-task questionnaire filled out by participants after creating sensors. Participants rated both conditions for *Control*, *Understand Capabilities*, *Communicate Requirements*, and *Test* on a 7-point Likert scale. Then they rated the helpfulness of each of Gensors' features on a 5-point Likert scale.

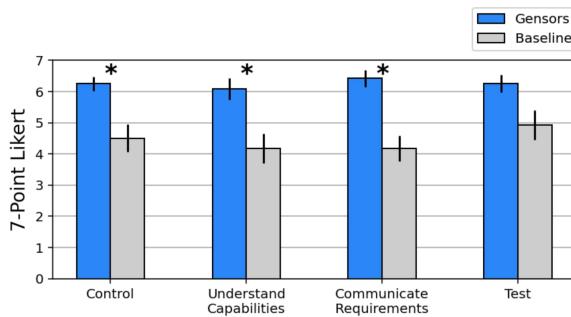


Figure 5: Comparing Gensors against baseline. Gensors had significantly higher ratings for *Control*, *Understand Capabilities*, and *Communicate Requirements*. (Bars are standard error and * indicates statistically significant difference, after full Bonferroni correction).

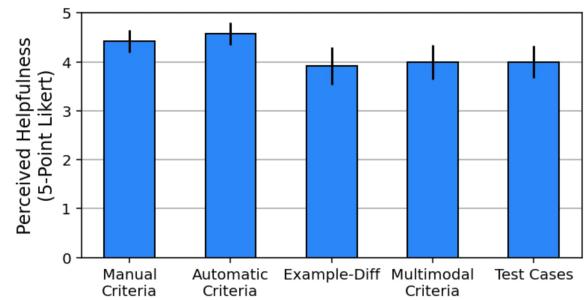


Figure 6: Participants' feedback on Gensors features. Of Gensors' features, all were considered helpful, with the automatically generated criteria perceived to be the most helpful.

well as the Examples-diff feature (which helped participants find additional “features” to differentiate between borderline cases).

Finally, although the difference was not statistically significant, participants rated their ability to test their sensors higher with Gensors ($\mu = 6.25$, $\sigma = 0.92$) compared to the baseline ($\mu = 4.95$, $\sigma = 1.55$).

The two highest rated features of Gensors (see Figure 6) were the ability to (1) automatically generate criteria ($\mu = 4.58$, $\sigma = 0.76$) and (2) manually create criteria ($\mu = 4.42$, $\sigma = 0.76$). The ability to define and debug the sensor's performance via individual criteria greatly supported users' workflows for creating sensors.

6 Findings

6.1 Quantitative Findings

The results from the questionnaire are summarized in Figure 5 & 6. Notably, participants reported significantly greater control over the sensor when using Gensors ($\mu = 6.26$, $\sigma = 0.72$) compared to the baseline ($\mu = 4.5$, $\sigma = 1.44$, $p < .01$). In the baseline condition, participants were often uncertain about how to best modify the prompt and felt their edits did little to influence the sensor.

Participants' perceived understanding of the underlying model was significantly higher with Gensors ($\mu = 6.08$, $\sigma = 1.11$) than with the baseline ($\mu = 4.17$, $\sigma = 1.57$). With Gensors, participants could pinpoint which individual criteria the sensor was failing on, iterate on them, and understand whether that criteria could be indeed detected by the model. In contrast, the baseline lacked the ability to isolate different criteria, making it difficult for participants to see how the model was attending to the requirements participants formulated in their prompts.

Participants also rated their ability to think through and communicate their requirements with Gensors ($\mu = 6.42$, $\sigma = 0.86$) significantly higher than the baseline ($\mu = 4.17$, $\sigma = 1.34$, $p < .001$). This was predominantly attributed to the automatically generated criteria (which provided a starting point for reflecting on requirements), as

6.2 How participants iterated on their sensors

The two conditions led to markedly different workflows for iterating on sensors. With the baseline prototype, participants typically observed the sensor's live output and spontaneously appended more criteria and caveats to their prompt in an attempt to better align the model's behavior with their own definition of the problem. For example, P1 began with a simple prompt: “Is my desk cluttered? Output yes or no and explain why.” The model initially stated that her desk was cluttered with too many items (e.g. a phone, pens, and keyboard). However, P1 considered these items as permanent fixtures of her desk, so she updated the prompt with: “I do not consider it cluttered if my phone, keyboard, mouse, pens, and picture frame are on the desk.” Later, she specified that empty food and drink containers should be considered cluttered. However, the model interpreted this too rigidly, flagging even a single container

a clutter. To fix this issue, she appended the clause: “One glass is not a problem.” Yet, not all misinterpretations were easy to correct. For instance, P1 tried to define a specific region of her desk to pay most attention to for clutter—“between here and the keyboard”—but the model could not grasp this distinction. After repeated attempts to rephrase, P1 expressed frustration, feeling that her efforts were counterproductive: “*I’ve gone too far down this rabbit hole, and it’s making it worse not better.*” Ultimately, in the baseline condition, participants *reactively* appended criteria and adjusted their prompts as they tested, spending a significant time tweaking their phrasing, as opposed to focusing predominantly on their criteria.

In contrast, with Gensors, participants adopted a more deliberate and top-down workflow, where they first defined a semi-comprehensive set of criteria and then revised these criteria through experimentation. For example, P2 started her desk clutter sensor with a few manually-written criteria, such as: “is the majority of the space in the view covered with objects?” When she was out of ideas, she decided to utilize the system’s auto-generate feature to expand her criteria set. The resulting suggestions, such as “are there non-work items on the desk?” complemented her initial set and created a more comprehensive set of requirements. With these criteria established upfront, participants then focused on refining them systematically rather than opportunistically discovering and iterating on criteria as they came up, as seen in the baseline condition. P6 specifically noted that he was “*trying to debug each criteria one-by-one,*” making adjustments to each one. Overall, Gensors encouraged a systematic, top-down workflow where participants first focused on the set of criteria that would control their sensor and then refined each of them individually. We further explore how users debugged and revised their criteria in the following section.

6.3 Gensors helped participants tune and troubleshoot sensors via criteria-level controls

A key benefit of Gensors was that participants could operate on their sensor at the criterion level, as opposed to the prompt level. Having the sensor explicitly check for each criteria helped participants better inspect and assess the model’s **underlying reasoning**. In the following section, we discuss how employing the model’s reasoning capabilities at the criteria-level helped participants with controlling, testing, and debugging their sensors.

6.3.1 Participants felt they had greater control via criteria than with the prompt. When working on their prompts in the baseline condition, participants were often unsure of how to best improve their prompt and if it was improving at all. For instance, P5, working on the desktop-clutter sensor in this condition, aimed to have the model output that his desk was cluttered when at least one empty can was present. However, despite multiple adjustments to the prompt, the sensor persistently reported his desk as clean. He carefully scanned the prompt for any potential miscommunications and hypothesized that the use of “and” in the phrase “Clutter also includes clothing accessories like hats **and** food waste like soda cans” might narrowly defined clutter to require both clothing accessories and food waste to be present. So, he changed this “and” to “or”, which then caused the sensor to briefly classify

the desk as cluttered, though not consistently so. Reflecting on this, P5 stated that prompts are “*easy to adjust, but it [the sensor] is not picking up the adjustments. I have to debug if it’s my wording or the model misfiring.*” When participants could not seem to influence the model, they began to question the prompt’s overall structure and efficacy. For example, when P1 could not get her prompt to focus on a particular portion of her desk, she eventually deleted her current version and started over. Overall, this sense of inadequacy in influencing the model often led to frustration, with some participants opting to restart the whole process in severe cases.

Meanwhile, in the Gensors’ condition, participants felt they could better influence the sensor’s performance through their edits to the criteria. All participants appreciated that each new criteria they added was explicitly checked, with an accompanying explanation. When the final verdict of the sensor wasn’t exactly what they expected, they were generally able to identify the corresponding criterion (or criteria) that wasn’t being interpreted correctly and modify it. For example, P3’s desk clutter sensor mistakenly classified his desk as uncluttered despite a stack of differently-sized books placed in the center. He expected the criterion which checked if the desk felt “chaotic and disorganized” to be triggered by this stack of books; however, the system considered the stack to be organized. He was able to refine the criterion’s wording to specify “book stacks with dissimilar-sized books” as chaotic, as well as add an image example, which ultimately steered his sensor to the results he expected. With Gensors, participants could make targeted edits and could add targeted examples to criteria to steer the sensor, as summarized by P3: “*It’s easier to be confident. It’s easier to change one criterion than to rewrite the whole thing [prompt].*”

6.3.2 Participants could robustly test and debug with Gensors. In the baseline condition, participants faced challenges isolating specific criteria and testing them effectively. To overcome this, they resorted to alternative approaches, such as adjusting their prompt to elicit more verbose descriptions of the scene. For instance, while working on the toddler-safety sensor, P7 quickly added to her prompt: “Please list any hazards visible.” Similarly, P4 added: “List all the objects you see and output if they are harmful to a toddler or not.” Despite these detailed descriptions, isolating and testing criteria still proved quite difficult. For instance, P4 was testing a clause in his prompt designed to check for “precariously placed” items that might fall on the toddler. He experimented with modifying the environment (e.g. placing a board on the edge of their bed), as well as adjusting the prompt to produce longer explanations. However, the model consistently classified the room as safe without explicitly addressing the “precariously placed” criterion. Despite further efforts to elicit longer explanations, P4 ultimately remained uncertain about the model’s reasoning regarding this particular criterion. In summary, participants sought to debug the sensor by requesting verbose explanations, but often struggled to get targeted insights pertaining to a specific criterion.

Meanwhile, with Gensors, participants could individually debug criteria, toggling each one on and off, inspecting each criterion’s output one by one, and iterating on each one if needed. For example, for the desktop clutter sensor, P6 first started with the “object count” criterion while disabling the rest. Observing that this criterion was counting upwards of eight objects on their desk but still

did not consider that enough for clutter, P6 refined it by adding “More than 4 objects is clutter” to this specific criterion. Next, he moved on to the “non-work item” criterion, which checked for non-work items on the desk. He noticed that the sensor marked this criterion as “true,” stating in its explanation that there was a mug and flower on the desk. P6 thus adjusted this criterion to consider their mug and flower vase as permanent fixtures of the desk, not an indicator of messiness. With these refinements, P6 was satisfied with the performance of the two criteria, enabled both, and quickly assessed the sensor’s overall effectiveness. Through this isolated testing, participants were better able to understand the underlying model’s capabilities and integrate the most effective criteria into their sensors. On rare occasions, some participants also removed criteria that did not seem to be working well deemed unnecessary.

6.4 Gensors helped participants consider failure modes beyond their specific context

In both conditions, participants manipulated their environment to test how their sensors would perform in a variety of scenarios. A common strategy was to create exaggerated setups, for example, making a desk conspicuously cluttered or impeccably clean. After establishing that the sensor could work effectively in these clear-cut scenarios, they would make incremental adjustments, generally by adding or removing objects, to bring the scenario closer to a *boundary* condition. For example, with Gensors, after his sensor stated his tidy kitchen was safe for toddlers, P9 placed a cutting knife on the countertop to test the model’s response to a highly unsafe object in an otherwise safe kitchen. Similarly, in the baseline condition, P5 organized his desk to what he considered a clean state, then introduced individual perturbations to make it “messy,” such as adding a can of soda. Overall, participants manipulated their environments to explore and test their sensors across a variety of scenarios.

While participants naturally engaged in creating these tests, they tended to overlook tricky visual situations or subtler failure modes in the baseline condition. Meanwhile, some participants appreciated that Gensors created suggested test cases for them, noting that it would otherwise be difficult to think of these alternative scenarios on their own. For example, P10, working on the desk clutter sensor with an “object count” criterion, appreciated the test case that suggested placing similar colored objects stacked on their desk to see if the model could distinguish them. They found that stacking objects together did impede the sensor’s ability to clearly distinguish them, especially if parts of objects were hidden beneath others. Similarly, for the “choking hazard” criterion in his toddler safety sensor, P4 appreciated a test case that suggested partially obscuring the choking hazard object (e.g. a coin) with another larger object. Overall, these criterion-specific test suggestions helped participants more thoroughly stress-test their sensors while gaining a deeper understanding of the underlying MLLM’s capabilities.

6.5 Gensors supplemented users’ own “blind spots” by suggesting complementary criteria

While Gensors enabled users to align the sensor most closely with their own personal criteria and preferences (e.g. via the manual-criteria feature), it also encouraged them to look beyond their immediate surroundings and field of view (e.g. via the auto-generated

criteria and Examples-diff features) for criteria that they would otherwise overlook.

For example, some users noted that the automatically-generated criteria prompted them to consider aspects of the problem they would not have thought of independently. For example, many users did not initially think of “uncovered outlets” as a potential hazard for toddlers, but realized they had missed this upon seeing it auto-generated. Similarly, for the desk clutter sensor, Gensors considered whether there was a “Usable Area” on the desk (e.g. “Is it easy to find a usable area on the desk without having to move items around?”). Participants found this surprising and compelling, as it reframed the problem of messiness in an unexpected way. However, one participant, P1, commented on a potential drawback of generated criteria, noting that they seemed to be “*too easy to say yes to*,” which might distract users from critically thinking about their own preferences. He then imagined alternative workflows where Gensors could help users actively consider criteria suggestions by posing yes/no questions.

Furthermore, the Examples-diff feature proved instrumental in helping users overcome “tunnel vision” when they struggled to identify additional criteria that could further differentiate borderline cases. For instance, while P11 was working on their desktop clutter sensor, they had a few borderline examples that their current set of criteria was not consistently differentiating. At this point, they felt that they had exhausted the criteria they could identify, so they turned to the Examples-diff feature, from which they selected two criteria: one that checks if the “objects on the desk appear to be randomly placed” and another that checks if the desk feels “chaotic and disorganized.” With these criteria, P11 felt his borderline examples were more consistently distinguished: “*It’s easy to describe the easy cases [with criteria]. It’s hard to describe the boundary cases. Having the model say here’s the difference between these two lets you focus on what the model is seeing.*” Occasionally, however, the Examples-diff feature would hallucinate differences between examples. For instance, P12 used the feature for her toddler safety sensor, and the model hallucinated that there was a toilet in their bedroom, creating a criterion that checked if the toilet’s lid was closed. Despite these occasional inaccuracies, the Examples-diff feature generally helped participants uncover more features to distinguish between more complex examples.

6.6 How MLLM peculiarities impacted sensor creation

MLLM hallucinations both benefited and inhibited participants’ processes in defining their sensor criteria. On one hand, when not entirely fabricated, hallucinations could sometimes be useful for helping participants become aware of criteria they did not consider. For example, while P3 was building his toddler safety sensor in baseline, the model deemed his bedroom unsafe due to the potential risk of a toddler falling out of open windows. Although P3’s windows were not actually open, he found this scenario plausible and ultimately adjusted his prompt to account for such a situation.

On the other hand, entirely fabricated hallucinations could confuse and distract users during the process. For instance, while P10 was working on her desk clutter sensor in the baseline, the model

falsely reported the presence of a stuffed animal on her desk. Hoping to prevent this hallucination in future runs, P10 added the clause “Ps: I don’t have stuffed animals” to her prompt. As a potential workaround, both P6 and P10 wanted a visual explanation (e.g. a bounding box around the supposed “stuffed animal”) to better understand the model’s perception.

In both conditions, the stochastic nature of MLLMs occasionally produced different results across runs with nearly identical inputs, which both confused and informed participants. Like traditional ML sensors, participants’ MLLM-powered sensors decisions exhibited “flickering” behavior, for example, alternating between “cluttered” and “uncluttered” assessments for the same desk setup. But perhaps what was more disorienting was that the accompanying explanation also varied across each run. Interestingly, P1, P2, and P12 all interpreted flickering as the sensor being “unsure” and used flickering as a cue that they needed to refine their prompt or criteria. In the baseline condition, P1 noticed her desk clutter sensor was flickering, and hypothesized that it was due to a clause in her prompt that specified “lots of papers” as clutter; she could see the model’s explanation alternated between claiming that there were “too many papers” to “just a few.” P12 gleaned a bit more information from flickering by aggregating the model’s decisions over a window (e.g. the sensor output “cluttered” the past 3 of 5 runs) to gauge which way the model was leaning towards and to better inform debugging. Ultimately, the stochasticity of the underlying model sometimes became a tool for participants to assess its uncertainty and the current state of their criteria.

7 Discussion

7.1 Supporting Active Criteria Elicitation

While participants appreciated the automatically generated criteria, some noted that it could potentially stifle them from carefully considering their own personal preferences for the sensors. They desired features that would more actively involve them in creating the criteria. There are many possibilities for alternative, more engaging workflows to help users realize their requirements. For instance, for a desk clutter sensor, one could first be presented with a set of personas, such as an “organized chaos” persona, who considers an uncluttered desk to be one that, while containing many items, is arranged in a neat and meticulously organized manner, or a “minimalist”, who prefers a desk with very few items. Users can reflect on these personas and their own preferences to *select the one they most identify with, which would then determine their initial set of criteria*. Another, perhaps simpler mechanism to actively engage users in thinking through their personal criteria could involve posing yes/no questions (e.g. “Do you typically keep a lot of items on your desk?”). Future research could explore the potential advantages and limitations of these alternative workflows for criteria elicitation.

7.2 An IoT Network of Sensors Proactively Reasoning Together

Reasoning can be extended beyond helping users achieve a particular sensing objective to creating an Internet of Things (i.e. a network of sensors) that communicate with each other and *proactively reason*. Instead of relying on a central node for decision-making, each node might reason over its own data and make proactive requests

of other nodes, such as requesting data or suggesting actions. For instance, a user’s smartwatch might detect a marathon training pattern from analyzing recent running data. It could then proactively request information from the user’s smart fridge to check if the necessary foods are available to support this training regimen. The fridge would analyze its contents, reason about missing items, and inform the smartwatch, which could then prompt the user to go grocery shopping.

Beyond this initial interaction, the fridge could proactively reason about further analyses it could conduct to continue supporting the user, such as by tracking their eating habits over the coming weeks (e.g., it might assess whether the user is consuming enough protein or carbohydrates for marathon preparation). To fulfill its analysis, the fridge might request fitness data from the smartwatch, such as the user’s weight and recent runs, to offer detailed meal recommendations. Overall, there are exciting possibilities for sensors within IoT networks to independently reason with their data, anticipate user needs, and collaborate with other sensors to provide proactive support.

7.3 Denoising Sensors via Reasoning

Another promising application of reasoning is in *denoising* AI sensors. Currently, temporary occlusions or disturbances in a sensor’s data stream can lead to erroneous decisions. For example, consider a sensor designed to track the duration of a user’s painting practice. At the start of a session, a pet might wander in front of the sensor’s camera, temporarily obscuring the user. This could cause the sensor to mistakenly conclude that the user has stopped painting. To address such noise, the model could incorporate reasoning about the *task state* and *recent outputs*. For instance, the model might infer that (1) painting is generally a longer task, and (2) that the user had just started painting based on prior video frames. Even with the temporary occlusion caused by the pet, the sensor could reasonably assume that the user is still painting. Future research could further explore equipping sensors with reasoning capabilities to enhance their resilience against noise by integrating contextual understanding of tasks and leveraging prior data outputs.

8 Limitations & Future Work

This work represents a first step towards user-friendly AI sensor definition with MLLMs, though several limitations suggest areas for future research.

We acknowledge that our study sample size of 12 participants, all relatively tech-savvy, may impact the generalizability of our findings. While we sought diversity in participant backgrounds and skill sets, future research could involve a larger, less tech-oriented sample to provide a more balanced interpretation of how users engage with Gensors. This would help ensure broader applicability of the results. In addition, we observed that users sometimes tested their sensors with limited example diversity, raising concerns about generalizability. Future iterations of Gensors could incorporate mechanisms to incentivize broader testing and evaluation across diverse scenarios, ensuring robust performance across a wider range of situations. Similarly, our study focused on a limited set of use cases, and some participants expressed less familiarity with scenarios outside their personal experience, such as those related to parenting.

Expanding the range of use cases and allowing participants to prioritize scenarios they are familiar with could provide further insights.

To gain a more comprehensive understanding of user needs and challenges, future research should move beyond the controlled lab setting and explore longitudinal studies where participants deploy and interact with AI sensors in their natural environment. This would provide valuable insights into long-term usability and identify challenges related to real-world deployment, such as integration with other smart home devices, enabling more complex actions triggered by sensor outputs, and exploring the potential for collaborative sensor development and sharing. This could also address limitations related to camera placement, field of view, and image quality, which were observed to impact sensor performance. Future work could explore providing standardized camera setups, developing tools for automatic scene adjustment, or incorporating depth information to enhance scene understanding.

In addition, addressing inherent limitations of MLLMs, such as hallucinations, is crucial. These hallucinations occasionally confused users and hindered their ability to understand and verify a sensor's behavior. On one hand, developing techniques to improve explainability and transparency, such as visualizing the model's focus on an image frame through bounding boxes or highlighting [71], could be beneficial. On the other hand, as the cost and latency associated with model calls continue to decrease, it may become feasible to run the same sensor prompt under varying temperatures or configurations and implement a subsequent “majority voting” mechanism. This approach, akin to how a final verdict prompt aggregates and determines an outcome based on individual criteria, offers promise in addressing the stochastic nature of MLLMs and mitigating hallucinations.

Furthermore, enhancing Gensors to support richer logical constructs beyond basic AND/OR operators and to handle continuous values or fuzzy boundaries for sensor outputs will broaden its applications and better capture real-world nuances. This may involve integrating confidence levels, thresholds, or fuzzy logic to deliver more informative and flexible sensor outputs. However, it will be crucial to balance this increased complexity with the implications on usability.

Last but not least, leveraging cloud-based LLMs could raise potential privacy concerns, particularly the risk of exposing sensitive home images or audio upon security breaches. Additionally, users may have reservations about their data being used for further model training by the cloud provider. As a first step towards mitigating these risks in a research prototype, we use the Gemini API, which follows strict data retention policies to ensure no user data is stored or used for training. In the future, smaller and more capable multimodal models could enable more inference tasks to be processed directly on users' devices, thereby significantly reducing security and privacy risks by limiting data transmission to external servers. This shift toward on-device intelligence offers a promising path for enhancing the privacy and security of personalized sensing systems.

9 Conclusion

This work explored using MLLMs to create customizable AI sensors, where users define complex sensing tasks through natural language.

However, our formative study revealed challenges in effectively articulating requirements and specifying desired sensor behavior through basic prompting. To address this, we developed Gensors, a system that facilitates AI sensor definition by decomposing high-level sensing tasks into explicit, testable criteria. Gensors leverages MLLM capabilities to offer relevant criteria, enable user customization, translate examples into new criteria, and suggest tests. Our user study demonstrated that Gensors significantly enhanced the AI sensor definition process, fostering deeper understanding, more systematic exploration of model capabilities, and ultimately, more robust and personalized sensor definitions. Despite challenges such as model hallucinations, Gensors demonstrates the potential to make intelligent sensing technologies more accessible and customizable. Future work could focus on mitigating these limitations and further enhancing user experience and control over sensor behavior.

Acknowledgments

We gratefully thank Jonathan Caton, Adam Connors, Aaron Donsbach, Lucas Dixon, Noah Fiedel, Jeff Gray, Minsuk Kahng, Alejandra Molina, Crystal Qian, Fernanda Viegas, Wing Wat, Martin Wattenberg, and Xiaotong Yang for their helpful comments. We also appreciate the valuable input from our study participants.

References

- [1] Amazon. 2024. Amazon Mechanical Turk. <https://www.mturk.com/>
- [2] Stephen H. Bach, Victor Sanh, Zheng-Xin Yong, Albert Webson, Colin Raffel, Nihal V. Nayak, Abheesh Sharma, Taewoon Kim, M. Saiful Bari, Thibault Fevry, Zaid Alyafeai, Manan Dey, Andrea Santilli, Zhiqing Sun, Srulik Ben-David, Canwen Xu, Gunjan Chhablani, Han Wang, Jason Alan Fries, Maged S. Al-shaibani, Shanya Sharma, Urmish Thakker, Khalid Almubarak, Xiangru Tang, Dragomir Radev, Mike Tian-Jian Jiang, and Alexander M. Rush. 2022. PromptSource: An Integrated Development Environment and Repository for Natural Language Prompts. doi:10.48550/arXiv.2202.01279 arXiv:2202.01279.
- [3] Joshua Bell. 2023. *Indexed Database API*. Technical Report. World Wide Web Consortium (W3C). <https://www.w3.org/TR/IndexerDB/>
- [4] Jeffrey P. Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C. Miller, Robin Miller, Aubrey Tatarowicz, Brandy White, Samuel White, and Tom Yeh. 2010. VizWiz: nearly real-time answers to visual questions. In *Proceedings of the 23rd annual ACM symposium on User interface software and technology (UIST '10)*. Association for Computing Machinery, New York, NY, USA, 333–342. doi:10.1145/1866029.1866080
- [5] Svetlin Bostandjiev, John O'Donovan, and Tobias Höllerer. 2012. TasteWeights: a visual interactive hybrid recommender system. In *Proceedings of the sixth ACM conference on Recommender systems (RecSys '12)*. Association for Computing Machinery, New York, NY, USA, 35–42. doi:10.1145/2365952.2365964
- [6] Virginia Braun and Victoria Clarke. 2006. Using thematic analysis in psychology. *Qualitative research in psychology* 3, 2 (2006), 77–101. Publisher: Taylor & Francis.
- [7] Michelle Carney, Barron Webster, Irene Alvarado, Kyle Phillips, Noura Howell, Jordan Griffith, Jonas Jongejan, Amit Pitaru, and Alexander Chen. 2020. Teachable Machine: Approachable Web-Based Tool for Exploring Machine Learning Classification. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems (CHI EA '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. doi:10.1145/3334480.3382839
- [8] Qiaochu Chen, Xinyu Wang, Xi Ye, Greg Durrett, and Isil Dillig. 2020. Multimodal synthesis of regular expressions. In *Proceedings of the 41st ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI 2020)*. Association for Computing Machinery, New York, NY, USA, 487–502. doi:10.1145/3385412.3385988
- [9] Justin Cheng and Michael S. Bernstein. 2015. Flock: Hybrid Crowd-Machine Learning Classifiers. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing (CSCW '15)*. Association for Computing Machinery, New York, NY, USA, 600–611. doi:10.1145/2675133.2675214
- [10] Paul Denny, Juho Leinonen, James Prather, Andrew Luxton-Reilly, Thezarie Amarouche, Brett A. Becker, and Brent N. Reeves. 2024. Prompt Problems: A New Programming Exercise for the Generative AI Era. In *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2024)*. Association for Computing Machinery, New York, NY, USA, 296–302. doi:10.1145/3626252.3630909

- [11] Michael Desmond and Michelle Brachman. 2024. Exploring Prompt Engineering Practices in the Enterprise. doi:10.48550/arXiv.2403.08950 arXiv:2403.08950.
- [12] Dan Ding, Rory A. Cooper, Paul F. Pasquina, and Lavinia Fici-Pasquina. 2011. Sensor technology for smart homes. *Maturitas* 69, 2 (June 2011), 131–136. doi:10.1016/j.maturitas.2011.03.016
- [13] Nanyi Fei, Zhiwu Lu, Yizhao Gao, Guoxing Yang, Yuqi Huo, Jingyuan Wen, Haoyu Lu, Ruihua Song, Xin Gao, Tao Xiang, Hao Sun, and Ji-Rong Wen. 2022. Towards artificial general intelligence via a multimodal foundation model. *Nature Communications* 13, 1 (June 2022), 3094. doi:10.1038/s41467-022-30761-2 Publisher: Nature Publishing Group.
- [14] Google DeepMind. 2024. Gemini 1.5 Flash. <https://deepmind.google/technologies/gemini/flash/> Published: Online; accessed October 10, 2024.
- [15] Google DeepMind. 2024. Gemini 1.5 Pro. <https://deepmind.google/technologies/gemini/pro/> Published: Online; accessed October 10, 2024.
- [16] Anhong Guo, Xiang 'Anthony' Chen, Haoran Qi, Samuel White, Suman Ghosh, Chiaki Asakawa, and Jeffrey P. Bigham. 2016. VizLens: A Robust and Interactive Screen Reader for Interfaces in the Real World. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. Association for Computing Machinery, New York, NY, USA, 651–664. doi:10.1145/2984511.2984518
- [17] Anhong Guo, Anuraag Jain, Shomiron Ghose, Gierad Laput, Chris Harrison, and Jeffrey P. Bigham. 2018. Crowd-AI Camera Sensing in the Real World. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 3 (Sept. 2018). doi:10.1145/3264921 Place: New York, NY, USA Publisher: Association for Computing Machinery.
- [18] Xin Hong, Chris Nugent, Maurice Mulvenna, Sally McClean, Bryan Scotney, and Steven Devlin. 2009. Evidential fusion of sensor data for activity recognition in smart homes. *Pervasive and Mobile Computing* 5, 3 (June 2009), 236–252. doi:10.1016/j.pmcj.2008.05.002
- [19] Jane Hsieh, Michael Xieyang Liu, Brad A. Myers, and Aniket Kittur. 2018. An Exploratory Study of Web Foraging to Understand and Support Programming Decisions. In *2018 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 305–306. doi:10.1109/VLHCC.2018.8506517 ISSN: 1943-6092.
- [20] Amy Hwang and Jesse Hoey. 2012. Smart home, the next generation: Closing the gap between users and technology. In *2012 AAAI Fall Symposium Series*.
- [21] Timo Jakobi, Gunnar Stevens, Nico Castelli, Corinna Ogonowski, Florian Schaub, Nils Vindice, Dave Randall, Peter Tolmie, and Volker Wulf. 2018. Evolving Needs in IoT Control and Accountability: A Longitudinal Study on Smart Home Intelligibility. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4 (Dec. 2018), 171:1–171:28. doi:10.1145/3287049
- [22] Ellen Jiang, Kristen Olson, Edwin Toh, Alejandra Molina, Aaron Donsbach, Michael Terry, and Carrie J Cai. 2022. PromptMaker: Prompt-based Prototyping with Large Language Models. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems (CHI EA '22)*. Association for Computing Machinery, New York, NY, USA, 1–8. doi:10.1145/3491101.3503564
- [23] Ellen Jiang, Edwin Toh, Alejandra Molina, Aaron Donsbach, Carrie J Cai, and Michael Terry. 2021. GenLine and GenForm: Two Tools for Interacting with Generative Language Models in a Code Editor. In *The Adjunct Publication of the 34th Annual ACM Symposium on User Interface Software and Technology (UIST '21)*. Association for Computing Machinery, New York, NY, USA, 145–147. doi:10.1145/3474349.3480209
- [24] Yuxin Jiang, Yufei Wang, Xingshan Zeng, Wanjun Zhong, Liangyou Li, Fei Mi, Lifeng Shang, Xin Jiang, Qun Liu, and Wei Wang. 2024. FollowBench: A Multi-level Fine-grained Constraints Following Benchmark for Large Language Models. doi:10.48550/arXiv.2310.20410 arXiv:2310.20410
- [25] Minsuk Kahng, Ian Tenney, Mahima Pushkarna, Michael Xieyang Liu, James Wexler, Emily Reif, Krystal Kallarackal, Minsuk Chang, Michael Terry, and Lucas Dixon. 2024. LLM Comparator: Interactive Analysis of Side-by-Side Evaluation of Large Language Models. *IEEE Transactions on Visualization and Computer Graphics* (2024), 1–11. doi:10.1109/TVCG.2024.3456354 Conference Name: IEEE Transactions on Visualization and Computer Graphics.
- [26] Minsuk Kahng, Ian Tenney, Mahima Pushkarna, Michael Xieyang Liu, James Wexler, Emily Reif, Krystal Kallarackal, Minsuk Chang, Michael Terry, and Lucas Dixon. 2024. LLM Comparator: Visual Analytics for Side-by-Side Evaluation of Large Language Models. In *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems (CHI EA '24)*. Association for Computing Machinery, New York, NY, USA, 1–7. doi:10.1145/3613905.3650755
- [27] V Kastrinaki, M Zervakis, and K Kalaitzakis. 2003. A survey of video processing techniques for traffic applications. *Image and Vision Computing* 21, 4 (April 2003), 359–381. doi:10.1016/S0262-8856(03)00004-0
- [28] Tae Soo Kim, Yoonjoo Lee, Jamin Shin, Young-Ho Kim, and Juho Kim. 2024. EvalLM: Interactive Evaluation of Large Language Model Prompts on User-Defined Criteria. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–21. doi:10.1145/3613904.3642216
- [29] Neil Klingensmith, Joseph Bomber, and Suman Banerjee. 2014. Hot, cold and in between: enabling fine-grained environmental control in homes for efficiency and comfort. In *Proceedings of the 5th international conference on Future energy systems (e-Energy '14)*. Association for Computing Machinery, New York, NY, USA, 123–132. doi:10.1145/2602044.2602049
- [30] Amy J. Ko, Robin Abraham, Laura Beckwith, Alan Blackwell, Margaret Burnett, Martin Erwig, Chris Scalfidi, Joseph Lawrence, Henry Lieberman, Brad Myers, Mary Beth Rosson, Gregg Rothermel, Mary Shaw, and Susan Wiedenbeck. 2011. The State of the Art in End-user Software Engineering. *ACM Comput. Surv.* 43, 3 (April 2011), 21:1–21:44. doi:10.1145/1922649.1922658
- [31] Johannes Kunkel, Benedikt Loepf, and Jürgen Ziegler. 2017. A 3D Item Space Visualization for Presenting and Manipulating User Preferences in Collaborative Filtering. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces (IUI '17)*. Association for Computing Machinery, New York, NY, USA, 3–15. doi:10.1145/3025171.3025189
- [32] Stacey Kuznetsov and Eric Paulos. 2010. UpStream: motivating water conservation with low-cost water flow sensing and persuasive displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. Association for Computing Machinery, New York, NY, USA, 1851–1860. doi:10.1145/175326.1753604
- [33] A. van Lamsweerde. 2009. *Requirements engineering : from system goals to UML models to software specifications*. John Wiley & Sons, Ltd. <https://thuviensho.asen.edu.vn/handle/123456789/9362>
- [34] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-Play Acoustic Activity Recognition. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, New York, NY, USA, 213–224. doi:10.1145/3242587.3242609
- [35] Gierad Laput, Walter S. Lasecki, Jason Wiese, Robert Xiao, Jeffrey P. Bigham, and Chris Harrison. 2015. Sensors: Adaptive, Rapidly Deployable, Human-Intelligent Sensor Feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1935–1944. doi:10.1145/2702123.2702416 event-place: Seoul, Republic of Korea.
- [36] Gierad Laput, Yang Zhang, and Chris Harrison. 2017. Synthetic Sensors: Towards General-Purpose Sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3986–3999. doi:10.1145/3025453.3025773 event-place: Denver, Colorado, USA.
- [37] Peter Lee, Sebastian Bubeck, and Joseph Petro. 2023. Benefits, Limits, and Risks of GPT-4 as an AI Chatbot for Medicine. *New England Journal of Medicine* 388, 13 (March 2023), 1233–1239. doi:10.1056/NEJMsr2214184 Publisher: Massachusetts Medical Society _eprint: <https://www.nejm.org/doi/pdf/10.1056/NEJMsr2214184>
- [38] Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, and Jianfeng Gao. 2024. Multimodal Foundation Models: From Specialists to General-Purpose Assistants. *Found. Trends. Comput. Graph. Vis.* 16, 1–2 (May 2024), 1–214. doi:10.1561/0600000110
- [39] Franklin Mingzhe Li, Michael Xieyang Liu, Shaun K. Kane, and Patrick Carrington. 2024. A Contextual Inquiry of People with Vision Impairments in Cooking. doi:10.1145/3613904.3642233 arXiv:2402.15108 [cs].
- [40] Lit. 2024. Lit Web Components. <https://lit.dev/>
- [41] Michael Xieyang Liu. 2023. *Tool Support for Knowledge Foraging, Structuring, and Transfer during Online Sensemaking*. Ph. D. Dissertation. Carnegie Mellon University. <http://reports-archive.adm.cs.cmu.edu/anon/usrl0/ftp/usr/ftp/hci/abstracts/23-105.html>
- [42] Michael Xieyang Liu, Jane Hsieh, Nathan Hahn, Angelina Zhou, Emily Deng, Shaun Burley, Cynthia Taylor, Aniket Kittur, and Brad A. Myers. 2019. Unakite: Scaffolding Developers' Decision-Making Using the Web. In *Proceedings of the 32Nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. ACM, New Orleans, LA, USA, 67–80. doi:10.1145/3332165.3347908 event-place: New Orleans, LA, USA.
- [43] Michael Xieyang Liu, Aniket Kittur, and Brad A. Myers. 2021. To Reuse or Not To Reuse? A Framework and System for Evaluating Summarized Knowledge. *Proceedings of the ACM on Human-Computer Interaction* 5, CSCW1 (April 2021), 166:1–166:35. doi:10.1145/3449240
- [44] Michael Xieyang Liu, Aniket Kittur, and Brad A. Myers. 2022. Crystalline: Lowering the Cost for Developers to Collect and Organize Information for Decision Making. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3491102.3501968 event-place: New Orleans, LA, USA.
- [45] Michael Xieyang Liu, Frederick Liu, Alexander J. Fiannaca, Terry Koo, Lucas Dixon, Michael Terry, and Carrie J. Cai. 2024. "We Need Structured Output": Towards User-centered Constraints on Large Language Model Output. doi:10.1145/3613905.3650756 arXiv:2404.07362 [cs].
- [46] Michael Xieyang Liu, Advait Sarkar, Carina Negreanu, Benjamin Zorn, Jack Williams, Neil Toronto, and Andrew D. Gordon. 2023. "What It Wants Me To Say": Bridging the Abstraction Gap Between End-User Programmers and Code-Generating Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–31. doi:10.1145/3544548.3580817

- [47] Michael Xieyang Liu, Tongshuang Wu, Tianying Chen, Franklin Mingzhe Li, Aniket Kittur, and Brad A. Myers. 2024. Selenite: Scaffolding Online Sense-making with Comprehensive Overviews Elicited from Large Language Models. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3613904.3642149
- [48] Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2023. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *ACM Comput. Surv.* 55, 9 (Jan. 2023), 195:1–195:35. doi:10.1145/3560815
- [49] Bertalan Meskó. 2023. Prompt Engineering as an Important Emerging Skill for Medical Professionals: Tutorial. *Journal of Medical Internet Research* 25, 1 (Oct. 2023), e50638. doi:10.2196/50638 Company: Journal of Medical Internet Research Distributor: Journal of Medical Internet Research Institution: Journal of Medical Internet Research Label: Journal of Medical Internet Research Publisher: JMIR Publications Inc., Toronto, Canada.
- [50] L.I. Millett, M. Thomas, and D. Jackson. 2007. *Software for Dependable Systems: Sufficient Evidence?* National Academies Press. https://books.google.com/books?id=_ACAgAAQBAJ
- [51] Changmo Nam, Jun-Cheol Park, and Dae-Shik Kim. 2016. Indoor Human Activity Recognition with Contextual Cues in Videos. In *Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication (IMCOM '16)*. Association for Computing Machinery, New York, NY, USA, 1–4. doi:10.1145/285746.2857645
- [52] Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. 2024. Using an LLM to Help With Code Understanding. doi:10.48550/arXiv.2307.08177 arXiv:2307.08177
- [53] Sydney Nguyen, Hannah McLean Babe, Yangtian Zi, Arjun Guha, Carolyn Jane Anderson, and Molly Q Feldman. 2024. How Beginning Programmers and Code LLMs (Mis)read Each Other. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI '24)*. Association for Computing Machinery, New York, NY, USA, 1–26. doi:10.1145/3613904.3642706
- [54] Mahda Noura, Sebastian Heil, and Martin Gaedke. 2020. Natural language goal understanding for smart home environments. In *Proceedings of the 10th International Conference on the Internet of Things (IoT '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. doi:10.1145/3410992.3410996
- [55] Chris Parnin, Gustavo Soares, Rahul Pandita, Sumit Gulwani, Jessica Rich, and Austin Z. Henley. 2023. Building Your Own Product Copilot: Challenges, Opportunities, and Needs. doi:10.48550/arXiv.2312.14231 arXiv:2312.14231 [cs].
- [56] Savvas Petridis, Nediyana Daskalova, Sarah Mennicken, Samuel F Way, Paul Lamere, and Jennifer Thom. 2022. TastePaths: Enabling Deeper Exploration and Understanding of Personal Preferences in Recommender Systems. In *Proceedings of the 27th International Conference on Intelligent User Interfaces (IUI '22)*. Association for Computing Machinery, New York, NY, USA, 120–133. doi:10.1145/3490099.3511156
- [57] Savvas Petridis, Nicholas Diakopoulos, Kevin Crowston, Mark Hansen, Keren Henderson, Stan Jastrzebski, Jeffrey V Nickerson, and Lydia B Chilton. 2023. AngleKindling: Supporting Journalistic Angle Ideation with Large Language Models. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–16. doi:10.1145/3544548.3580907
- [58] Savvas Petridis, Michael Xieyang Liu, Alexander J. Fiannaca, Vivian Tsai, Michael Terry, and Carrie J. Cai. 2024. In Situ AI Prototyping: Infusing Multimodal Prompts into Mobile Settings with MobileMaker. In *2024 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 121–133. doi:10.1109/VLHCC60511.2024.00023 ISSN: 1943-6106.
- [59] Savvas Petridis, Michael Terry, and Carrie Jun Cai. 2023. PromptInfuser: Bringing User Interface Mock-ups to Life with Large Language Models. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems (CHI EA '23)*. Association for Computing Machinery, New York, NY, USA, 1–6. doi:10.1145/3544549.3585628
- [60] Savvas Petridis, Ben Wedin, Ann Yuan, James Wexler, and Nithum Thain. 2024. ConstitutionalExperts: Training a Mixture of Principle-based Prompts. doi:10.48550/arXiv.2403.04894 arXiv:2403.04894 [cs].
- [61] Savvas Petridis, Benjamin D Wedin, James Wexler, Mahima Pushkarna, Aaron Donsbach, Nitesh Goyal, Carrie J Cai, and Michael Terry. 2024. Constitution-Maker: Interactively Critiquing Large Language Models by Converting Feedback into Principles. In *Proceedings of the 29th International Conference on Intelligent User Interfaces (IUI '24)*. Association for Computing Machinery, New York, NY, USA, 853–868. doi:10.1145/3640543.3645144
- [62] James Pierce, Richmond Y. Wong, and Nick Merrill. 2020. Sensor Illumination: Exploring Design Qualities and Ethical Implications of Smart Cameras and Image/Video Analytics. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–19. doi:10.1145/3313831.3376347
- [63] Crystal Qian, Michael Xieyang Liu, Emily Reif, Grady Simon, Nada Hussein, Nathan Clement, James Wexler, Carrie J. Cai, Michael Terry, and Minsuk Kahng. 2024. The Evolution of LLM Adoption in Industry Data Curation Practices. doi:10.48550/arXiv.2412.16089 arXiv:2412.16089 [cs].
- [64] Ring. 2024. Security Cameras - Outdoor Wireless & Wired.
- [65] Daniel M. Russell, Laura Koesten, Aniket Kittur, Nitesh Goyal, and Michael Xieyang Liu. 2024. Sensemaking: What is it today?. In *Extended Abstracts of the 2024 CHI Conference on Human Factors in Computing Systems (CHI EA '24)*. Association for Computing Machinery, New York, NY, USA, 1–5. doi:10.1145/3613905.3636322
- [66] Douglas C. Schmidt, Jesse Spencer-Smith, Quchen Fu, and Jules White. 2024. Towards a Catalog of Prompt Patterns to Enhance the Discipline of Prompt Engineering. *Ada Lett.* 43, 2 (June 2024), 43–51. doi:10.1145/3672359.3672364
- [67] James Scott, A.J. Bernheim Brush, John Krumm, Brian Meyers, Michael Hazas, Stephen Hodges, and Nicolas Villar. 2011. PreHeat: controlling home heating using occupancy prediction. In *Proceedings of the 13th international conference on Ubiquitous computing (UbiComp '11)*. Association for Computing Machinery, New York, NY, USA, 281–290. doi:10.1145/2030112.2030151
- [68] Shreya Shankar, Haotian Li, Parth Asawa, Madelon Hulsebos, Yiming Lin, J. D. Zamfirescu-Pereira, Harrison Chase, Will Fu-Hinthorn, Aditya G. Parameswaran, and Eugene Wu. 2024. SPADE: Synthesizing Data Quality Assertions for Large Language Model Pipelines. doi:10.48550/arXiv.2401.03038 arXiv:2401.03038
- [69] Shreya Shankar, J. D. Zamfirescu-Pereira, Björn Hartmann, Aditya G. Parameswaran, and Ian Arawjo. 2024. Who Validates the Validators? Aligning LLM-Assisted Evaluation of LLM Outputs with Human Preferences. doi:10.48550/arXiv.2404.12272 arXiv:2404.12272
- [70] Lei Shi, Maryam Ashoori, Yunfeng Zhang, and Shiri Azenkot. 2018. Knock knock, what's there: converting passive objects into customizable smart controllers. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '18)*. Association for Computing Machinery, New York, NY, USA, 1–13. doi:10.1145/3229434.3229453
- [71] Jean Y. Song, Stephan J. Lemmer, Michael Xieyang Liu, Shiyuan Yan, Juho Kim, Jason J. Corso, and Walter S. Lasecki. 2019. Popup: reconstructing 3D video using particle filtering to aggregate crowd responses. In *Proceedings of the 24th International Conference on Intelligent User Interfaces (IUI '19)*. Association for Computing Machinery, Marina del Ray, California, 558–569. doi:10.1145/3301275.3302305
- [72] Mojtaba Vaismoradi, Hannele Turunen, and Terese Bondas. 2013. Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study. *Nursing & Health Sciences* 15, 3 (2013), 398–405. doi:10.1111/nhs.12048 _eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/nhs.12048>
- [73] Gust Verbruggen, Vu Le, and Sumit Gulwani. 2021. Semantic programming by example with pre-trained models. *Replication Package for Article: "Semantic programming by example with pre-trained models"* 5, OOPSLA (Oct. 2021), 100:1–100:25. doi:10.1145/3485477
- [74] Carl Vondrick, Donald Patterson, and Deva Ramanan. 2013. Efficiently Scaling up Crowdsourced Video Annotation. *International Journal of Computer Vision* 101, 1 (Jan. 2013), 184–204. doi:10.1007/s11263-012-0564-1
- [75] Jong-bum Woo and Youn-kyung Lim. 2015. User experience in do-it-yourself-style smart homes. In *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '15)*. Association for Computing Machinery, New York, NY, USA, 779–790. doi:10.1145/2750858.2806063
- [76] Hui Ye and Hongbo Fu. 2022. ProGesAR: Mobile AR Prototyping for Proxemic and Gestural Interactions with Real-world IoT Enhanced Spaces. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22)*. Association for Computing Machinery, New York, NY, USA, 1–14. doi:10.1145/3491102.3517689
- [77] Sojeong Yun and Youn-Kyung Lim. 2023. Potential and Challenges of DIY Smart Homes with an ML-intensive Camera Sensor. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA. doi:10.1145/3544548.3581462 event-place: Hamburg, Germany.
- [78] J.D. Zamfirescu-Pereira, Heather Wei, Amy Xiao, Kitty Gu, Grace Jung, Matthew G Lee, Bjoern Hartmann, and Qian Yang. 2023. Herding AI Cats: Lessons from Designing a Chatbot by Prompting GPT-3. In *Proceedings of the 2023 ACM Designing Interactive Systems Conference (DIS '23)*. Association for Computing Machinery, New York, NY, USA, 2206–2220. doi:10.1145/3563657.3596138
- [79] J.D. Zamfirescu-Pereira, Richmond Y. Wong, Bjoern Hartmann, and Qian Yang. 2023. Why Johnny Can't Prompt: How Non-AI Experts Try (and Fail) to Design LLM Prompts. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. Association for Computing Machinery, New York, NY, USA, 1–21. doi:10.1145/3544548.3581388
- [80] Tianyi Zhang, London Lowmanstone, Xinyu Wang, and Elena L. Glassman. 2020. Interactive Program Synthesis by Augmented Examples. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST '20)*. Association for Computing Machinery, New York, NY, USA, 627–648. doi:10.1145/3379337.3415900
- [81] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-Following Evaluation for Large Language Models. doi:10.48550/arXiv.2311.07911 arXiv:2311.07911 [cs].