XR Blocks: Accelerating Human-Centered AI + XR Innovation

Preliminary White Paper, September 2025

David Li*, Nels Numan†, Xun Qian†, Yanhe Chen†, Zhongyi Zhou†, Evgenii Alekseev, Geonsun Lee, Alex Cooper, Min Xia, Scott Chung, Jeremy Nelson, Xiuxiu Yuan, Jolica Dias, Tim Bettridge, Benjamin Hersh, Michelle Huynh, Konrad Piascik, Ricardo Cabello, David Kim, Ruofei Du*‡
https://github.com/google/xrblocks
https://xrblocks.github.io
Google XR Labs

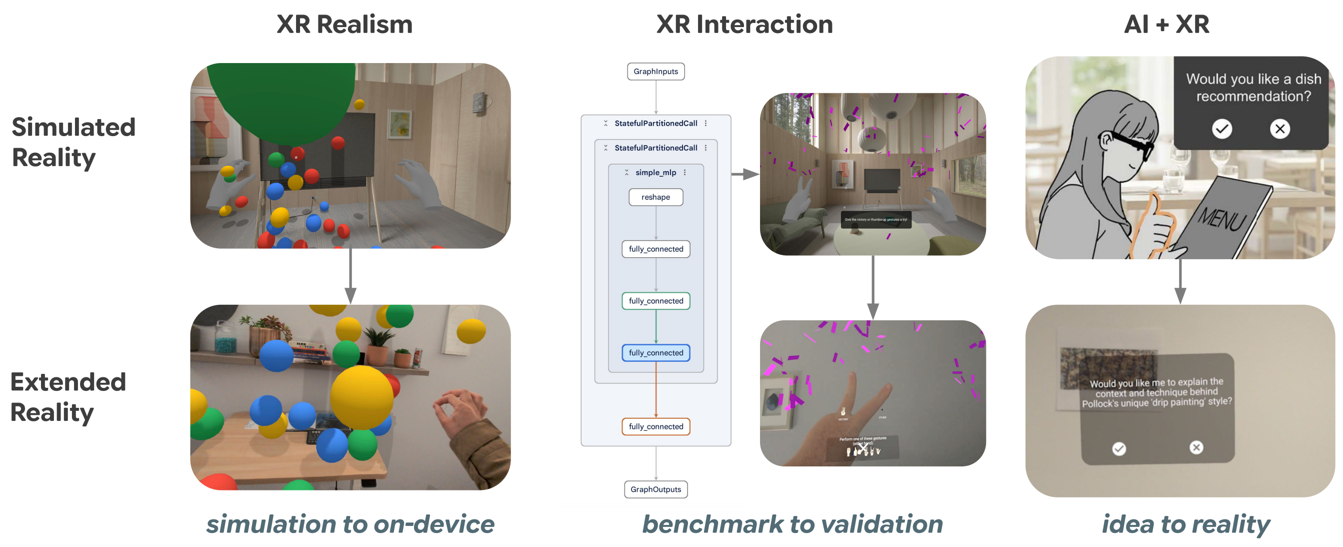

Figure 1: XR Blocks accelerates the prototyping of real-time AI + XR applications across desktop simulators and XR devices. Examples: (a) XR Realism: Prototype depth-aware, physics-based interactions [20] in simulation and deploy the same code to real-world XR devices. (b) XR Interactions: Seamlessly integrate custom gesture models to desktop simulator and on-device XR deployment. (c) AI + XR Integration: Build intelligent, context-aware assistants, like the Sensible Agent [34].

ABSTRACT

We are on the cusp where Artificial Intelligence (AI) and Extended Reality (XR) are converging to unlock new paradigms of interactive computing. However, a significant gap exists between the ecosystems of these two fields: while AI research and development is accelerated by mature frameworks like JAX and benchmarks like LMArena, prototyping novel AI-driven XR interactions remains a high-friction process, often requiring practitioners to manually integrate disparate, low-level systems for perception, rendering, and interaction. To bridge this gap, we present XR Blocks, a cross-platform framework designed to accelerate human-centered AI + XR innovation. XR Blocks strives to provide a modular architecture with plug-and-play components for core abstraction in AI + XR: user, world, peers; interface, context, and agents. Crucially, it is designed with the mission of "*reducing frictions from idea to reality*", thus accelerating rapid prototyping of AI + XR apps. Built upon accessible technologies (WebXR, three.js, TensorFlow, Gemini), our toolkit lowers the barrier to entry for XR creators. We demonstrate its utility through a set of open-source templates, samples, and advanced demos, empowering the community to quickly move from concept to interactive XR prototype.

KEYWORDS

Extended Reality, Software Development Kit, WebXR, WebGL, Programming Language, Depth-based Interaction, Mixed Reality, Augmented Reality, Virtual Reality, Toolkit, AI, Gemini, Android XR, TensorFlow Lite, LiteRT

1 INTRODUCTION

The convergence of Artificial Intelligence (AI) and Extended Reality (XR) is reshaping human-computer interaction, unlocking innovation to empower human with augmented perception, augmented reasoning, and augmented interaction with the physical world, other people, devices, and AI agents.

*Both authors contribute equally. † equal contributions, sorted alphabetically.
‡Corresponding author and directional lead: Ruofei Du, me [at] duruofei [dot] com.

The AI community has demonstrated the power of a robust ecosystem: frameworks like PyTorch [41] and JAX [7], coupled with open benchmarks like ImageNet [14] and LMArena [10], have created a virtuous cycle of rapid, community-driven innovation. XR, however, has yet to tap into this flywheel effect. Its research and development landscape remains fragmented and fraught with friction, hindering the translation of innovative ideas into robust, interactive, on-device experiences. For example, algorithms prototyped in a specific game engine (Unity) six years ago in DepthLab [20] are non-trivial to deploy on modern XR hardware. Oftentimes, creators are left to contend with the immense overhead of low-level systems programming, on-device testing, and endless one-off user studies just to validate a single interaction or real-time algorithm. While MediaPipe [38, 60] and Visual Blocks [18, 61] offer novel solutions with coding and no-code frameworks to streamline AI pipeline development, their applications in XR are limited due to the missing access to sensors and natural user input.

This challenge is becoming more significant with the rise of a new paradigm–vibe coding [22]–an intent-driven creation. Platforms such as Gemini Canvas [25] and Cursor [12] provide a no-coding environment for anyone to author end-to-end mobile and web applications. While their 3D generation process are still rapidly advancing, we believe there will still be a critical gap preventing this magic from extending seamlessly into the XR domain. It is one thing to *generate a 3D model of a frog-like hat* with core technologies like DreamFusion [44] and Magic3D [36]; it is another entirely to empower a user to simply *pinch and summon a frog-like hat on the real-world desk, then hand over to remote participant*, as demonstrated in `Thing2Reality` [29]. This seemingly simple instruction requires a complex, implicit understanding of environmental perception, user intent, and atomic hand interactions–a specification that is currently missing. We envision a future where developers and designers specify novel interactions *once*, using a high-level, human-centered language, and see them deployed *everywhere*, including desktops, mobile, and XR headsets. They should be free to focus on inventing new user experiences, not re-implementing the fundamental mechanics of perception and interaction, where the individual perception modules are easily interchangeable.

To invent this future, we introduce XR Blocks, a framework engineered to accelerate human-centered AI + XR innovation. As presented in Figure 1, XR Blocks offers an SDK to allow creators to simulate XR interaction directly on desktop and test them on device, benchmark a custom gesture model with virtual hands and then test it with real hand, and augment human abilities by fusing AI into XR. While XR Blocks is not the silver bullet for this grand challenge, but rather we create a proof-of-concept framework by distributed and part-time team members to provide the AI + XR community with the critical missing element: **a set of open source abstraction** for the core components of intelligent immersive systems, including user representation, environmental perception, and spatial interaction. It is architected to streamline the complex interplay between these elements, drastically reducing the code and expertise required to bridge the gap between concept to interactive prototype. Its initial version is built as `xrblocks.js`, a lightweight, cross-platform library on accessible web technologies (`WebXR` [4], `three.js` [52], `TensorFlow` [49], and `Gemini` [51]). We envision a future extension of XR Blocks will extend to native platform with LLM-powered compilers.

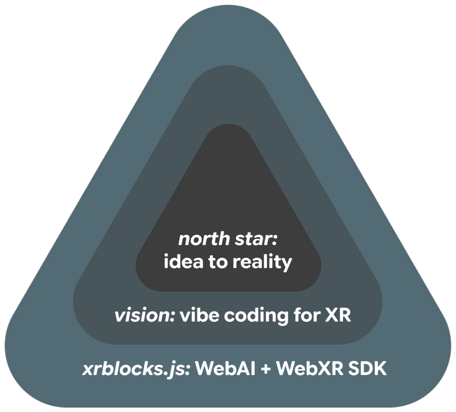

Figure 2: XR Blocks roadmap. This initial version of `xrblocks.js` serves as an opensource framework to accelerate prototyping with WebAI and WebXR. It should be iterated to empower vibe coding for XR. We envision this framework to achieve "idea to reality" in XR that follows human-centered objectives [9]: (i) aligning with human values in XR; (ii) assimilating human intents in XR; and (iii) augmenting human abilities in XR.

We demonstrate XR Blocks's power and flexibility through a suite of open-source toolkit at https://github.com/google/xrblocks, sample applications, and advanced demos that showcase tangible pathways for rapid prototyping, from depth-aware interaction, gestural controls, to dynamic generative human-AI interactions with a user's physical space.

2 RELATED WORK

XR Blocks strives to offer a high-level abstraction and a set of tools for creators to author AI+XR experiences. Our work is built upon `three.js`, a mature WebGL rendering library, and is informed by a rich history of prior art in systems, toolkits, and programming frameworks for both AI and XR. We position XR Blocks in the context of three main areas: frameworks for 3D and XR interaction, the "ecosystem effect" that accelerates AI innovation, and prior efforts to bridge AI with interactive 3D worlds.

2.1 Frameworks for 3D Interaction and Extended Reality

The challenge of simplifying 3D and XR application development is not new. Seminal research systems in the 1990s laid the foundational groundwork for device abstraction and interaction management. VR Juggler [6] provided a powerful, reconfigurable data-flow architecture for CAVE-like [11] virtual environments, while the MR Toolkit [47] was among the first to formalize device-independent interaction techniques. The widespread adoption of ARToolKit [32] democratized augmented reality research by providing a robust, open-source solution for fiducial marker tracking, enabling a generation of researchers to experiment with AR interfaces.

In the modern XR era, this pursuit has continued, largely bifurcated between native mobile SDKs and comprehensive game engines. Mobile-specific SDKs like Apple's `ARKit` [2] and Google's `ARCore` [26] provide powerful, low-level access to core tracking technologies like SLAM, plane detection, and image tracking. While essential, these SDKs primarily offer a foundation for perception, leaving the implementation of high-level interaction patterns and application logic to the developer. On the other end of the spectrum, engines like Unity [53], Unreal [55], and Godot [24], have become the de-facto standard for commercial XR development, offering a high-ceiling environment with extensive tooling. However, their complexity presents a high floor for rapid prototyping, and integrating novel, external AI models can introduce significant friction. Worsestill, newer game engines may deprecate or remove older APIs, requiring **extensive code modifications** to use the updated equivalents.

To address the need for XR interactivity, several higher-level frameworks have been built upon Unity. In 2016, the Mixed Reality Toolkit (MRTK) [39], designed for Microsoft HoloLens [40], provides creators with a vast library of production-grade UI controls, advanced interaction models (such as gaze-pinch indirect manipulation), and data binding systems for dynamic UI. Later in 2018, Unity's XR Interaction Toolkit (XRI) [54] offers a integral high-level, component-based system for creating VR and AR experiences. Further, community-driven framekworks such as MKRT3 [3] and VRTK [57], embraces XRI and the OpenXR standard, embodying a modular and performance-oriented design philosophy.

In constrast to the native ecosystem, `WebXR`, as a W3C standard, offers unparalleled accessibility and cross-platform compatibility for XR development on the open web. Libraries like `three.js` [52] and `Babylon.js` [5] offer developers direct, programmatic control over the rendering pipeline, scene graph, and interactions. Frameworks like `A-Frame` [1] have successfully lowered the barrier to entry for `WebXR` by providing a declarative, entity-component-system (ECS) architecture based on HTML. XR Blocks shares this goal of accessibility across platforms, but differs in focus, prioritizing a high-level, user-centric and AI-driven interaction model in XR over a purely declarative scene graph. Our work builds directly on the rendering capabilities of `three.js`, but provides a distinct contribution in the *abstraction layer* for orchestrating perception, interaction, and AI-driven behavior.

2.2 The Ecosystem Effect in Artificial Intelligence

A key motivation for our work is the "flywheel effect" observed in the AI research community. The unprecedented pace of innovation in AI can be attributed to a symbiotic ecosystem composed of three pillars. First, standardized and extensible frameworks like `TensorFlow` [49] and `PyTorch` [41] provided a common language and powerful abstractions like automatic differentiation, freeing researchers to focus on model architecture. Second, canonical datasets and benchmarks like `ImageNet` [14] and LMArena [10] created shared goals and objective measures of progress, fostering healthy competition and collaboration. Finally, open model hubs like `Hugging Face` [30, 48] and `TensorFlow Hub` [27] democratized access to state-of-the-art pretrained models, creating a culture of sharing and composition.

This virtuous cycle does not yet exist for interactive XR research. Even reproducing a piece of XR research on an updated game engine or new hardware requires lots of work. XR Blocks is designed as a step toward fostering such an ecosystem: all demos made with XR Blocks can be reproducible by hosting independently on the web. By providing a common, high-level framework and open-sourcing our examples, we aim to create a shared substrate upon which the community can build, compare, and distribute novel human-centered AI+XR interactions.

2.3 Bridging AI and Interactive 3D Worlds

The goal of integrating AI with interactive 3D environments has a long history, particularly in the context of training embodied agents. Simulation platforms such as AI2-THOR [33] and Habitat [45, 46, 50] provide photorealistic and physically accurate environments for training agents in tasks like navigation and object manipulation. These systems, while powerful, are designed for the offline or non-real-time training of autonomous agents. Our work differs in its focus: XR Blocks is designed for creating real-time, human-in-the-loop interactive experiences for end-users, where the goal is often human-AI collaboration rather than pure agent autonomy.

Other research has explored using natural language to manipulate 3D worlds like LLMR [13] and Thing2Reality [29], allowing users to construct or edit scenes declaratively. XR Blocks builds directly on this legacy, but integrates modern LLMs as a central mechanism for interpreting user intent from a richer, multi-modal context that includes not just language, but also gesture and environmental sensing. In doing so, XR Blocks synthesizes these threads, taking inspiration from the interactivity of HCI frameworks and the goals of embodied AI to provide a unique tool focused on the rapid prototyping of human-centered AI + XR systems.

3 DESIGN PRINCIPLES

The XR Blocks team strives to foster an ecosystem of Human-AI creation for perceptive XR experiences. Our architectural and API design choices are guided by three core principles:

(1) **Embrace Simplicity and Readability**: Inspired by the philosophy of Python's Zen [43], we prioritize clean, human-readable abstractions. Our goal is that a developer's Script should read like a high-level description of the desired experience. Simple tasks should be simple to implement, and complex logic should remain explicit and understandable. This is critical for a framework intended for rapid prototyping by a diverse community of creators.
(2) **Prioritize the Creator Experience**: Our primary goal is to make authoring intelligent and perceptive XR applications as seamless as possible. We believe that creators should focus on the user experience, not on the low-level "plumbing" of sensor fusion, AI model integration, or cross-platform interaction logic. XR Blocks is designed to absorb this incidental complexity, providing powerful, ready-to-use primitives.
(3) **Pragmatism over Completeness**: We follow a design philosophy of pragmatism, akin to "worse is better" [23]. The

fields of AI and XR are evolving at an immense speed. A comprehensive, complex framework that attempts to be perfect will be obsolete upon release. We favor a simple, modular, and adaptable architecture that is "good enough" for a wide range of applications.

4 AN ARCHITECTURAL DESIGN FOR GENERATIVE REALITY

Designing the next generation of XR experiences — interactive worlds that are context-aware, augmented, and deeply personal — requires a paradigm shift in our authoring tools. We must move beyond managing low-level rendering and perception pipelines and toward sculpting interactions and intent. Drawing inspiration from Visual Blocks for ML [18] and InstructPipe [61], we designed XR Blocks around a core principle: **to provide a high-level, human-centered abstraction layer that separates the *what* of an interaction** (the `Script`) **from the *how* of its low-level implementation**. This architecture is our first step toward a future of `Generative Reality`.

4.1 The `Reality Model`: A High-Level Abstraction of Reality

At the heart of our design is `Script`, the narrative and logical center of an application. As shown in Figure 3, the `Script` does not operate on disconnected data streams, but on a unified `Reality Model`, a coherent, interactive representation of XR that elegantly fuses the physical and digital bits.

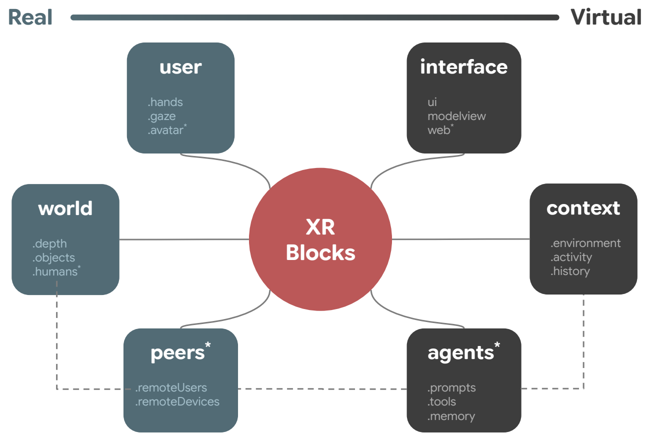

Figure 3: The conceptual `Reality Model` of the XR Blocks framework. At the center, the `Script` contains the application's logic and operates on a unified model of first-class primitives including the user, the physical world, AI agents, and the application context. Entities with * have not yet been fully implemented in the GitHub.

This model is composed of several first-class primitives:

- **User & the Physical World**: Our model is centered around the `User`, which is represented by their hands, gaze, and avatar. The physical `World` allows the `Script` to query the perceived reality such as depth, estimated lighting condition, objects, and tracked humans.
- **Virtual Interfaces & Context**: The model augments the blended reality with virtual UI elements, from 2D panels to fully 3D assets. Perception pipeline analyzes the context of environment, activities, and histories of interaction.
- **Intelligent & Social Entities**: AI-driven agents and remote human peers are not external components but are treated as primary entities within the model. This allows for rich, contextual interactions between all participants, real and artificial.

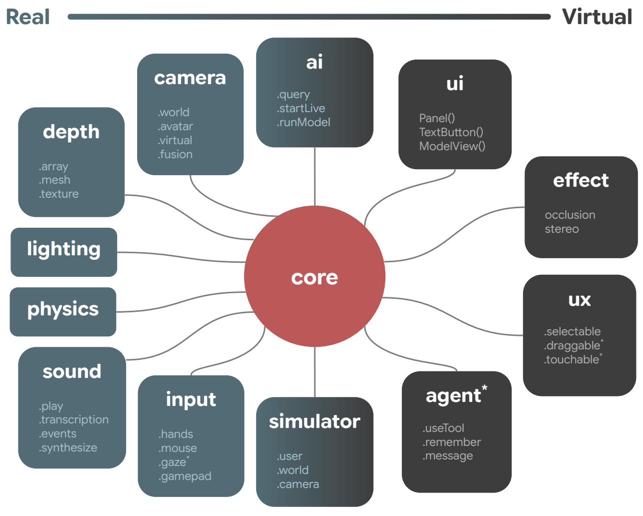

Figure 4: The modular architecture of the XR Blocks's `Script` engine. The `Script` orchestrates a suite of essential subsystems to realize the framework's high-level abstractions, spanning perception (`depth`, `input`), AI integration (`ai`, `agent`), and user experience (`ui`, `ux`). Subsystems with * have not yet been fully implemented in the GitHub.

4.2 The Core Engine: An Extensible Toolkit for Authoring AI + XR Experiences

This high-level `Reality Model` is realized by XR Blocks's modular Core engine (Figure 4), an extensible toolkit that orchestrates the complex subsystems required for spatial computing. The Core provides a high-level API that enables developers to harness these subsystems without needing to master the implementation.

- **Perception & Input Pipeline** The `camera`, `depth`, and `sound` modules continuously feed and update the `Reality Model`'s representation of physical reality. The `input` module normalizes user actions from various devices, providing the raw data for the Interaction Grammar to interpret.
- **AI as a Core Utility** The `ai` module acts as a central nervous system, providing simple yet powerful functions (`.query`, `.runModel`) that make large models an accessible utility. This is complemented by the `agent` module, which provides AI entities with concrete capabilities to act and remember (`.useTool`, `.remember`).

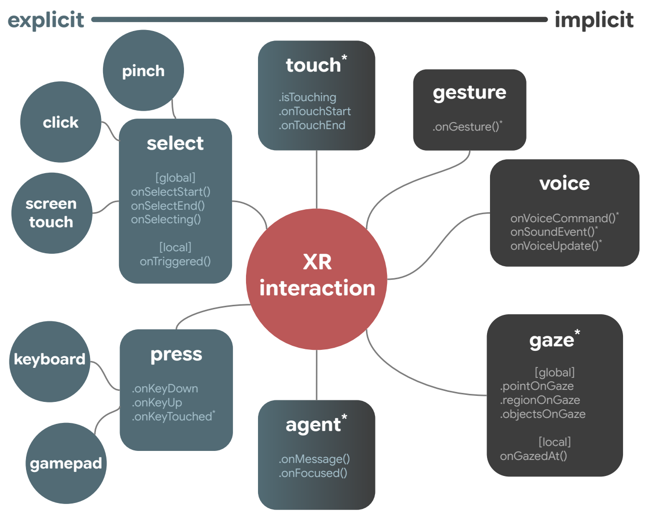

Figure 5: The Interaction Grammar of XR Blocks, which abstracts user input by distinguishing between two types of interaction. Explicit events are direct, low-level inputs (*e.g.*, a touch or click), while implicit intents are higher-level interpretations (*e.g.*, a gesture or voice command), allowing creators to build interaction against user intent.

- **Experience & Visualization Toolkit** To enable rapid creation, the toolkit provides a library of common affordances. The `ux` module offers reusable interaction behaviors like `.selectable` and `.draggable`, while the `ui` and `effect` modules handle the rendering of interfaces and complex visual effects like occlusion.

4.3 A Vision for AI + XR Creative Prototyping

By separating the abstract Reality Model from the concrete Core engine, XR Blocks enables a powerful new creative workflow. It allows creators to move from high-level, human-centric ideas to interactive prototypes at unprecedented speeds. We envision a future, where a declarative prompt, *"When the user pinches at an object, an agent should generate a poem of it."*, could be directly translated to high-level instructions in XR Blocks:

```
for (const object of world.objects) {
  if (user.isSelectingAt(object)) {
    const prompt = 'Write a poem with ${object.name}.';
    const poem = agent.query(prompt);
    this.textView.setText(poem);
  }
}
```

Hence, creator's prompt is no longer a pseudocode but a direct summary of the implementation logic. This envisioned framework seamlessly translates this intent into a system-level execution flow, composing capabilities from the `input`, `sound`, `ai`, `world`, `ui`, and `agent` modules to generate an emergent, intelligent behavior.

This is merely the first step. The architecture of XR Blocks points toward a future where generative AI is applied not just to conversational agents, but to the fabric of reality itself. Future work can explore scripting with even higher-level intents, such as, *"Make this room feel like a cyberpunk cafe"* allowing the `depth`, `lighting`, `sound`, and `agent` modules to collaborate in synthesizing a complete, multi-sensory experience. This opens avenues for differentiable world-building, or world-building with LLM-driven "Gradient" inspired by TextGrad [59] and ToolGrad [62], where environmental aesthetics and even procedural geometry could be optimized by AI in response to real-world context or user emotion. Our framework aims to provide a substrate for exploring learnable interaction grammars, where systems could co-evolve personalized and more expressive communication modalities with their users over time.

Ultimately, XR Blocks is not just a toolkit but a hypothesis about the future of creative expression: a future where the boundary between programming, design, and conversation dissolves, enabling everyone to script realities as fluidly as we script stories.

5 XR BLOCKS APPLICATIONS

To demonstrate the expressive power and flexibility of the XR Blocks architecture, we developed a suite of demonstrative applications and interactive examples in https://xrblocks.github.io/samples. These are not merely tech demos but tangible explorations into future interactive paradigms, showcasing how our framework's core principles enable the rapid prototyping of experiences that were previously complex and costly to build. The following examples highlight how XR Blocks facilitates the creation of realistic, interactive, and intelligent mixed-reality worlds.

5.1 Enhancing XR Realism

A fundamental requirement for immersive experiences is the seamless blending of real and virtual elements. XR Blocks's unified `Reality Model` allows creators to achieve this with minimal effort by composing high-level modules.

Dynamic Occlusion and Relighting: By leveraging the depth [56] and lighting estimation [35] modules, virtual objects are rendered with a deep awareness of the physical environment. A person can realistically walk behind a real-world sofa to observe a virtual cat, and a physical light source can cast light and shadows that interact correctly with the virtual cat and the room's actual geometry. This complex behavior is achieved with a few declarative lines, as XR Blocks handles the continuous fusion of real-world sensor data from WebXR API with the virtual scene's rendering pipeline.

Physics-Based World Interaction: We demonstrate how the physics module can be applied to virtual objects, allowing them to interact with the real world's geometry [20, 31]. For example, a creator can enable physics in XR Blocks and instantiate a virtual ball in `onSelect()`. When a user shoots the ball, it realistically bounces off real-world geometries, with its trajectory and behavior governed by the live depth map of the room.

5.2 Intelligent Environments with AI + XR Integration

The true power of our framework is realized when the `Reality Model` is deeply integrated with generative AI. This allows for the creation of dynamic, personalized environments that respond intelligently to user intent.

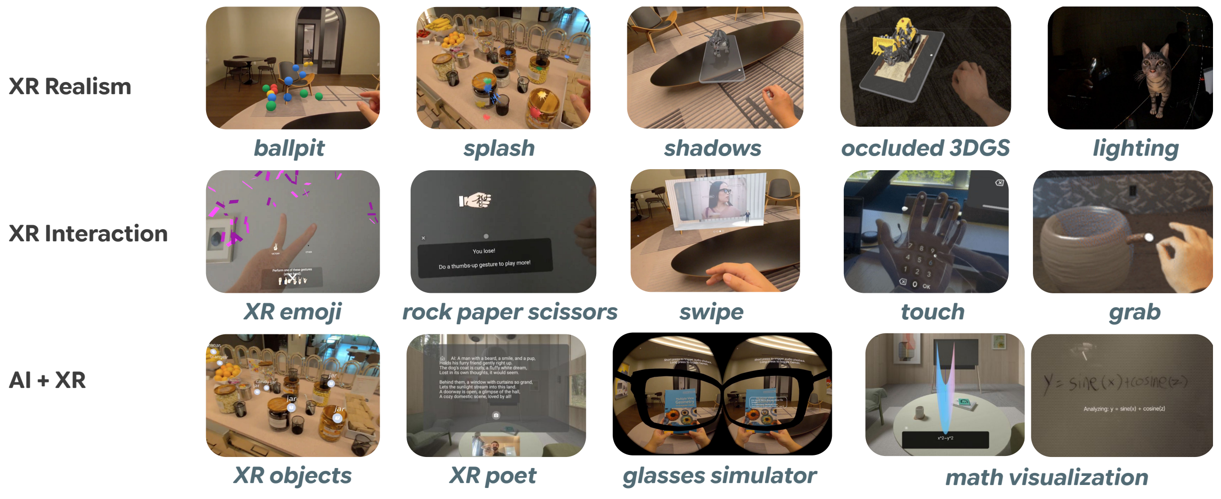

Figure 6: Applications of XR Blocks. (1) XR Realism: depth-aware and physics-based ballpit and splash games; geometry-aware shadows, 3D Gaussian splatting with occlusion, and lighting estimation. (2) XR Interaction: immersive emoji and rock paper scissors game empowered by custom ML models, dynamic swipe recognition, touch and grab with the physical world. (3) AI + XR: integration with Gemini Live, XR objects, glasses simulation in XR, and poem generation with real-world camera.

Augmented Object Intelligence (XR-Objects): Building on our prior work [16], we use XR Blocks to imbue everyday physical objects with interactive, digital affordances. When a user performs a long pinch, the `ai` module identifies the objects in the physical world. The framework then dynamically attaches a set of virtual buttons and allows users to touch these buttons to ask questions. This envisions a future of turning passive physical objects into active, programmable interfaces.

Proactive and Unobtrusive Assistance: XR Blocks served as the foundation for Sensible Agent [34], a system for proactive and unobtrusive interaction with an AR assistant. Sensible Agent tackles the critical challenge that many AR agents are intrusive and distracting, proposing a framework where agents only intervene when it is appropriate and helpful. Developing this system was significantly accelerated by XR Blocks. The agent's ability to perceive the user's current activity and environment is directly supported by our unified `Reality Model`. Its core logic for deciding when to intervene was implemented using the `ai` module, while the design and testing of its subtle, peripheral notifications were rapidly prototyped with the `ui` modules. Ultimately, Sensible Agent is a powerful testament to our framework's primary goal: by providing robust, high-level tools for perception and interaction, XR Blocks empowers HCI researchers to focus on higher-order challenges, such as the social and cognitive principles of human-agent collaboration.

6 DISCUSSION

The demos and applications presented in this paper are more than just technical showcases; they are tangible evidence of a new paradigm for creative work. By abstracting the immense complexity of underlying hardware and AI models, XR Blocks envisions a future where creators and AI can operate at the speed of thought. The core contribution is not merely a new library, but an argument for a fundamental shift in how we approach the authoring of interactive systems: a shift from low-level programming to high-level, intent-driven creation.

6.1 From Language Models to AI+XR Models

The rapid commoditization of large language models has unlocked unprecedented generative capabilities. However, a model that can describe a "cyberpunk cafe" in text is fundamentally different from a system that can render one in a user's physical space, complete with appropriate lighting, sound, and interactive agents. The challenge lies in grounding generative AI in the rich, multi-modal context of a user's reality. XR Blocks is our first step toward bridging this gap. Our "`Reality Model`" provides the necessary structure to channel the raw potential of generative AI, transforming it from a disembodied "chatbot" into an embodied, context-aware collaborator capable of perceiving, understanding, and modifying a shared reality.

6.2 Limitations and Future Work

XR Blocks provides a robust architectural blueprint for a new era of AI+XR development. Our current implementation prioritizes the core abstractions, and its limitations highlight several exciting avenues for future research.

System design *vs.* implementation: Our framework's conceptual primitives—agents, peers, and context—are depicted with a starting point, yet not fully implemented. The agent primitive, for instance, could be extended to support richer personalities, persistent memory, and more sophisticated proactive behaviors. Similarly, the future peers model should lay the groundwork for complex cross-device interactions that go beyond simple data sharing. A critical next step is to enhance the context and user models to perceive and adapt to Situationally Induced Impairments and Disabilities (SIIDs) [37], making AI-driven assistance truly accessible and equitable.

Performance and Latency: Our choice of web technologies prioritizes accessibility but brings known trade-offs. The framework can never match the rendering performance of native engines like Unity or Unreal, and reliance on cloud AI models introduces network latency. Near-term future work will explore hybrid architectures that combine our framework's flexibility with native performance, alongside on-device model distillation for real-time AI. A more visionary goal is to develop *an LLM-driven cross-compiler* capable of translating a high-level XR Blocks script into optimized, native code for multiple target engines, effectively solving the trade-off between ease-of-use and raw performance.

The Abstraction Ceiling: Our high-level abstractions, by design, prioritize rapid prototyping over ultimate expressivity. This "abstraction ceiling" means there will always be novel or performance-critical interactions that require lower-level control. A key future goal is to design an elegant "escape hatch" mechanism. This would allow expert developers to seamlessly write custom shaders, interface directly with device sensors, or implement bespoke interaction logic, all without breaking the integrity of the high-level XR Blocks framework.

WebXR and Privacy: We chose WebXR to maximize accessibility and efficiency, but this comes with a trade-off: to protect user privacy, web standards deliberately restrict access to sensitive sensor data like raw eye-tracking or face-tracking feeds, which are available in native OpenXR environments. We envision future browser-level APIs that allow for privacy-preserving on-device processing. In this model, raw sensor data would never leave the user's device. Instead, the browser may expose a secure, sandboxed API providing high-level, privacy-safe signals. This requires more resources and efforts from both industry and community.

6.3 The Future of XR Interaction and Creation

Our architecture opens the door to several exciting, long-term research directions.

Learnable Interaction Grammars: Currently, our mapping from implicit intents to explicit events is hand-designed. The XR Blocks framework provides the ideal substrate to explore systems that could learn a user's unique interaction preferences and gestural vocabulary over time, leading to truly personalized and co-adaptive interfaces. Future endeavors shall graudually extend seminal works in gestures [42, 58], locomotion methods [15], and cross-device interaction [8, 63] as primitives into the SDK to allow for a richer library towards learnable interaction grammar.

Differentiable and Co-Adaptive Realities: For the graphics and AI communities, our framework suggests a future where entire scenes and agent behaviors are procedurally optimized. If a user states, "make this room feel more relaxing", a differentiable rendering pipeline could be guided by an AI model to tune lighting, color, and even geometry to achieve the desired affective goal. This same principle extends to the agents within a mirrored world [17, 21]. An agent's behaviors, conversational style, and animations could be co-optimized within this loop to better suit the user's personality and context. Realizing this vision will require deep integration between interactive platforms like XR Blocks and modern ML frameworks, a challenge our modular architecture is designed to facilitate.

Multi-Sensory Synthesis: We have focused on audio-visual experiences, but the XR Blocks model is designed to be extensible. Integrating modules for haptics, EEG, and other sensory modalities could allow creators to compose truly immersive, multi-sensory narratives. E.g., creators today can extend XR Blocks with Arduino through WebUSB protocol.

7 CONCLUSION

The process of creating intelligent, interactive XR experiences is currently too fragmented and complex, placing a significant barrier between a creator's vision and its realization. In this paper, we presented XR Blocks, an architecture and toolkit designed to dissolve this complexity. By providing a high-level, human-centered abstraction layer that separates the *what* of an interaction from the *how* of its low-level implementation, XR Blocks dramatically accelerates the prototyping process. We have demonstrated how this approach enables the rapid development of sophisticated applications that seamlessly blend the real and virtual, and are imbued with contextual intelligence. XR Blocks is more than a toolkit; it is a foundational step toward a future where the boundary between programming, design, and conversation disappears, enabling us to script realities as fluidly as we script stories. However, XR Blocks is far from completion and this white paper serves as an initial visionary documentation to attract more creators to join our journey. We believe: with the right set of tools, everyone can unleash their inner creativity with AI.

ACKNOWLEDGMENTS

We would like to deeply thank Xiuxiu Yuan, Oren Haskins, Brian Collins, Barak Moshe, Seeyam Qiu, Steve Toh for the UX Design Support; Yinghua Yang and Brenton Simpson for the cross-team collaboration; Tim Bettridge, Kevin Lam for significant contributions to XR Blocks x Gemini Canvas; Furthermore, we would like to extend thanks to the seminal work in Android XR Unity Samples with Mahdi Tayarani, Max Dzitsiuk, Patrick Hackett, all of DepthLab [20], Ad hoc UI [19], Rapsai (Visual Blocks) [18], InstructPipe [61], DialogueLab [28], and Sensible Agent [34] co-authors, which greatly inspired this project along the way; we further extend thanks to Eric Gonzalez, Nicolás Peña Moreno, Max Spear, Yi-Fei Li, Ziyi Liu, Adarsh Kowdle, and Guru Somadder for their insightful feedback and support in XR Blocks.

REFERENCES

[1] A-Frame Authors. 2025. A-Frame. https://aframe.io/.
[2] Apple Inc. 2025. ARKit. https://developer.apple.com/documentation/arkit.
[3] Mixed Reality Toolkit Authors. 2025. MRTK3. https://github.com/MixedRealityToolkit/MixedRealityToolkit-Unity
[4] WebXR authors. 2022. WebXR. https://immersiveweb.dev/
[5] babylon.js authors. 2022. Babylon.js. https://www.babylonjs.com/
[6] Allen Bierbaum, Christopher Just, Patrick Hartling, Kevin Meinert, Albert Baker, and Carolina Cruz-Neira. 2001. VR Juggler: A Virtual Platform for Virtual Reality Application Development. In *Proceedings IEEE Virtual Reality 2001*. IEEE, 89–96.
[7] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. 2018. *JAX: Composable Transformations of Python+NumPy Programs*. http://github.com/jax-ml/jax
[8] Frederik Brudy, Christian Holz, Roman Rädle, Chi-Jui Wu, Steven Houben, Clemens Nylandsted Klokmose, and Nicolai Marquardt. 2019. Cross-Device Taxonomy: Survey, Opportunities and Challenges of Interactions Spanning Across Multiple Devices. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–28. https://doi.org/10.1145/3290605.3300792

[9] Xiang'Anthony' Chen, Jeffrey Burke, Ruofei Du, MatthewK. Hong, Jennifer Jacobs, Philippe Laban, Dingzeyu Li, NanyunViolet Peng, KarlD.D. Willis, Chien-Sheng Wu, and Bolei Zhou. 2023. Next Steps for Human-Centered Generative AI: A Technical Perspective. , 34 pages. https://doi.org/10.48550/arXiv.2306.15774

[10] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. 2024. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning.*

[11] Carolina Cruz-Neira, Daniel J Sandin, Thomas A DeFanti, Robert V Kenyon, and John C Hart. 1992. The CAVE: Audio visual experience automatic virtual environment. *Commun. ACM* 35, 6 (1992), 64–73.

[12] Cursor. [n.d.]. Cursor - The AI Code Editor. https://cursor.com.

[13] Fernanda De La Torre, CathyMengying Fang, Han Huang, Andrzej Banburski-Fahey, Judith Amores Fernandez, and Jaron Lanier. 2024. LLMR: Real-time Prompting of Interactive Worlds Using Large Language Models. In *Proceedings of the CHI Conference on Human Factors in Computing Systems.* ACM. https://doi.org/10.1145/3613904.3642579

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-scale Hierarchical Image Database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE. https://doi.org/10.1109/CVPR.2009.5206848

[15] Massimiliano Di Luca, Hasti Seifi, Simon Egan, and Mar Gonzalez-Franco. 2021. Locomotion Vault: the Extra Mile in Analyzing VR Locomotion Techniques. In *Proceedings of the 2021 CHI conference on human factors in computing systems.* 1–10. https://doi.org/10.1145/3411764.3445319

[16] Mustafa Doga Dogan, Eric J Gonzalez, Karan Ahuja, Ruofei Du, Andrea Colaço, Johnny Lee, Mar Gonzalez-Franco, and David Kim. 2024. Augmented Object Intelligence With XR-Objects. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology.* 1–15. https://doi.org/10.1145/3654777.3676379

[17] Ruofei Du, David Li, and Amitabh Varshney. 2019. Geollery: A Mixed Reality Social Media Platform. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI).* ACM, 13. https://doi.org/10.1145/3290605.3300915

[18] Ruofei Du, Na Li, Jing Jin, Michelle Carney, Scott Miles, Maria Kleiner, Xiuxiu Yuan, Yinda Zhang, Anuva Kulkarni, XingyuBruce Liu, Ahmed Sabie, Sergio Orts-Escolano, Abhishek Kar, Ping Yu, Ram Iyengar, Adarsh Kowdle, and Alex Olwal. 2023. Rapsai: Accelerating Machine Learning Prototyping of Multimedia Applications Through Visual Programming. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI).* ACM, 1–23. https://doi.org/10.1145/3544548.3581338

[19] Ruofei Du, Alex Olwal, MathieuLe Goc, Shengzhi Wu, Danhang Tang, Yinda Zhang, Jun Zhang, DavidJoseph Tan, Federico Tombari, and David Kim. 2022. Opportunistic Interfaces for Augmented Reality: Transforming Everyday Objects Into Tangible 6DoF Interfaces Using Ad Hoc UI. In *Extended Abstracts of the 2022 CHI Conference on Human Factors in Computing Systems (CHI).* ACM, 1–4. https://doi.org/10.1145/3491101.3519911

[20] Ruofei Du, Eric Turner, Maksym Dzitsiuk, Luca Prasso, Ivo Duarte, Jason Dourgarian, Joao Afonso, Jose Pascoal, Josh Gladstone, Nuno Cruces, Shahram Izadi, Adarsh Kowdle, Konstantine Tsotsos, and David Kim. 2020. DepthLab: Real-Time 3D Interaction With Depth Maps for Mobile Augmented Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST).* ACM, 829–843. https://doi.org/10.1145/3379337.3415881

[21] Ruofei Du and Amitabh Varshney. 2016. Social Street View: Blending Immersive Street Views with Geo-Tagged Social Media. In *Proceedings of the 21st International Conference on Web3D Technology (Web3D).* ACM, 77–85. https://doi.org/10.1145/2945292.2945294

[22] Benj Edwards. [n.d.]. Will the Future of Software Development Run on Vibes. https://arstechnica.com/ai/2025/03/is-vibe-coding-with-ai-gnarly-or-reckless-maybe-some-of-both.

[23] Richard Gabriel. 1991. The rise of worse is better. *Lisp: Good News, Bad News, How to Win Big* 2, 5 (1991).

[24] Godot. 2022. Godot Engine. https://godotengine.org/

[25] Google. [n.d.]. Google Gemini. https://gemini.google.com/canvas. Accessed: August 8, 2025.

[26] Google. 2025. ARCore. https://developers.google.com/ar.

[27] Google. 2025. TensorFlow Hub. https://www.tensorflow.org/hub.

[28] Erzhen Hu, Yanhe Chen, Mingyi Li, Vrushank Phadnis, Pingmei Xu, Xun Qian, Alex Olwal, David Kim, Seongkook Heo, and Ruofei Du. 2025. DialogLab: Authoring, Simulating, and Testing Dynamic Group Conversations in Hybrid Human-AI Conversations. In *Proceedings of the 39th Annual ACM Symposium on User Interface Software and Technology (UIST).* ACM. https://doi.org/10.1145/3746059.3747696

[29] Erzhen Hu, Mingyi Li, Andrew Hong, Xun Qian, Alex Olwal, David Kim, Seongkook Heo, and Ruofei Du. 2025. Thing2Reality: Enabling Spontaneous Creation of 3D Objects From 2D Content Using Generative AI in XR Meetings. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology.* Association for Computing Machinery. https://doi.org/10.1145/3746059.3747621

[30] Hugging Face. 2025. Hugging Face – The AI community building the future. https://huggingface.co.

[31] Shahram Izadi, Andrew Davison, Andrew Fitzgibbon, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Dustin Freeman. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology - UIST '11.* ACM. https://doi.org/10.1145/2047196.2047270

[32] Hirokazu Kato and Mark Billinghurst. 1999. Marker Tracking and HMD Calibration for a Video-Based Augmented Reality Conferencing System. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99).* IEEE, 85–94. https://doi.org/10.1109/IWAR.1999.803809

[33] Eric Kolve, Roozbeh Mottaghi, Winson Han, Eli VanderBilt, Luca Weihs, Alvaro Herrasti, Matt Deitke, Kiana Ehsani, Daniel Gordon, Yuke Zhu, et al. 2017. AI2-THOR: An Interactive 3D Environment for Visual AI. *arXiv preprint arXiv:1712.05474* (2017).

[34] Geonsun Lee, Min Xia, Nels Numan, Xun Qian, David Li, Yanhe Chen, Achin Kulshrestha, Ishan Chatterjee, Yinda Zhang, Dinesh Manocha, David Kim, and Ruofei Du. 2025. Sensible Agent: A Framework for Unobtrusive Interaction with Proactive AR Agent. In *Proceedings of the 39th Annual ACM Symposium on User Interface Software and Technology (UIST).* ACM. https://doi.org/10.1145/3746059.3747748

[35] Chloe LeGendre, Wan-Chun Ma, Graham Fyffe, John Flynn, Laurent Charbonnel, Jay Busch, and Paul Debevec. 2019. DeepLight: Learning Illumination for Unconstrained Mobile Mixed Reality. In *ACM SIGGRAPH 2019 Talks (SIGGRAPH '19).* ACM, Article 46, 46:1–46:2 pages. https://doi.org/10.1145/3306307.3328173

[36] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. 2023. Magic3D: High-resolution Text-to-3D Content Creation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.* 300–309. https://doi.org/10.1109/CVPR52729.2023.00037

[37] Xingyu Bruce Liu, Jiahao Nick Li, David Kim, Xiang 'Anthony' Chen, and Ruofei Du. 2024. Human I/O: Towards a Unified Approach to Detecting Situational Impairments. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems (CHI).* ACM, 18. https://doi.org/10.1145/3613904.3642065

[38] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. 2019. MediaPipe: A Framework for Building Perception Pipelines. https://doi.org/10.48550/arXiv.1906.08172

[39] Microsoft. 2020. Mixed Reality Toolkit. https://github.com/microsoft/MixedRealityToolkit-Unity

[40] Microsoft. 2025. HoloLens. https://learn.microsoft.com/en-us/hololens.

[41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. , 8026 – 8037 pages. https://doi.org/10.48550/arXiv.1912.01703

[42] Siyou Pei, Alexander Chen, Jaewook Lee, and Yang Zhang. 2022. Hand Interfaces: Using Hands to Imitate Objects in AR/VR for Expressive Interactions. In *CHI Conference on Human Factors in Computing Systems.* ACM. https://doi.org/10.1145/3491102.3501898

[43] Tim Peters. 2004. The zen of python. In *Pro Python.* Springer, 301–302.

[44] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. 2022. DreamFusion: Text-To-3D Using 2D Diffusion. *ArXiv* (2022). https://doi.org/10.48550/arXiv.2209.14988

[45] Xavi Puig, Eric Undersander, Andrew Szot, Mikael Dallaire Cote, Ruslan Partsey, Jimmy Yang, Ruta Desai, Alexander William Clegg, Michal Hlavac, Tiffany Min, Theo Gervet, Vladimir Vondrus, Vincent-Pierre Berges, John Turner, Oleksandr Maksymets, Zsolt Kira, Mrinal Kalakrishnan, Jitendra Malik, Devendra Singh Chaplot, Unnat Jain, Dhruv Batra, Akshara Rai, and Roozbeh Mottaghi. 2023. Habitat 3.0: A Co-Habitat for Humans, Avatars and Robots.

[46] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. 2019. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV).*

[47] Chris Shaw, Mark Green, Jiandong Liang, and Yunqi Sun. 1993. Decoupled Simulation in Virtual Reality with the MR Toolkit. *ACM Transactions on Information Systems (TOIS)* 11, 3 (1993), 287–317.

[48] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. 2024. HuggingGPT: Solving AI Tasks with ChatGPT and Its Friends in Hugging Face. *Advances in Neural Information Processing Systems* 36 (2024). https://doi.org/10.48550/arXiv.2303.17580

[49] Daniel Smilkov, Nikhil Thorat, Yannick Assogba, Charles Nicholson, Nick Kreeger, Ping Yu, Shanqing Cai, Eric Nielsen, David Soegel, Stan Bileschi, Michael Terry, Ann Yuan, Kangyi Zhang, Sandeep Gupta, Sarah Sirajuddin, D Sculley, Rajat Monga, Greg Corrado, Fernanda Viegas, and Martin M Wattenberg. 2019.

TensorFlow.js: Machine Learning for the Web and Beyond. 1 (2019), 309–321. https://doi.org/10.48550/arXiv.1901.05350

[50] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. 2021. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. In *Advances in Neural Information Processing Systems (NeurIPS)*.

[51] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: A Family of Highly Capable Multimodal Models. *arXiv preprint arXiv:2312.11805* (2023). https://doi.org/10.48550/arXiv.2312.11805

[52] three.js authors. 2022. Three.js. https://threejs.org

[53] Unity. 2022. Unity Game Engine. https://unity.com/products/unity-platform

[54] Unity. 2025. XR Interaction Toolkit. https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@3.0/manual/index.html.

[55] Unreal. 2022. Unreal Engine. https://www.unrealengine.com

[56] Julien Valentin, Adarsh Kowdle, Jonathan T. Barron, Neal Wadhwa, Max Dzitsiuk, Michael Schoenberg, Vivek Verma, Ambrus Csaszar, Eric Turner, Ivan Dryanovski, Joao Afonso, Jose Pascoal, Konstantine Tsotsos, Mira Leung, Mirko Schmidt, Onur Guleryuz, Sameh Khamis, Vladimir Tankovitch, Sean Fanello, Shahram Izadi, and Christoph Rhemann. 2018. Depth From Motion for Smartphone AR. *ACM Transactions on Graphics (TOG)* 37, 6, Article 193 (2018), 193:1–193:19 pages. https://doi.org/10.1145/3272127.3275041

[57] VRTK Authors. 2025. VRTK. https://www.vrtk.io.

[58] Tianyi Wang, Xun Qian, Fengming He, Xiyun Hu, Yuanzhi Cao, and Karthik Ramani. 2021. GesturAR: An Authoring System for Creating Freehand Interactive Augmented Reality Applications. In *9*. ACM. https://doi.org/10.1145/3472749.3474769

[59] Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. 2024. TextGrad: Automatic "Differentiation" Via Text. https://doi.org/10.48550/arXiv.2406.07496

[60] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. Mediapipe Hands: On-Device Real-Time Hand Tracking. *ArXiv Preprint ArXiv:2006.10214* (2020). https://doi.org/10.48550/arXiv.2006.10214

[61] Zhongyi Zhou, Jing Jin, Vrushank Phadnis, Xiuxiu Yuan, Jun Jiang, Xun Qian, Jingtao Zhou, Yiyi Huang, Zheng Xu, Yinda Zhang, Kristen Wright, Jason Mayes, Mark Sherwood, Johnny Lee, Alex Olwal, David Kim, Ram Iyengar, Na Li, and Ruofei Du. 2023. InstructPipe: Building Visual Programming Pipelines With Human Instructions. https://doi.org/10.48550/arXiv.2312.09672

[62] Zhongyi Zhou, Kohei Uehara, Haoyu Zhang, Jingtao Zhou, Lin Gu, Ruofei Du, Zheng Xu, and Tatsuya Harada. 2025. ToolGrad: Efficient Tool-use Dataset Generation with Textual " Gradients". *arXiv preprint arXiv:2508.04086* (2025). https://doi.org/10.48550/arXiv.2508.04086

[63] Fengyuan Zhu, Xun Qian, Daniel Kalmar, Mahdi Tayarani, Eric Gonzalez, Mar Gonzalez-Franco, David Kim, and Ruofei Du. 2025. Beyond the Phone: Exploring Context-Aware Interaction Between Mobile and Mixed Reality Devices. In *2025 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE. https://doi.org/10.1109/VR59515.2025.00099