ILLIXR: An Open Testbed to Enable Extended Reality Systems Research

Muhammad Huzaifa, Rishi Desai, Samuel Grayson, Xutao Jiang, Ying Jing, Jae Lee, Fang Lu, Yihan Pang, Joseph Ravichandran, Finn Sinclair, Boyuan Tian, Hengzhi Yuan, Jeffrey Zhang, and Sarita V. Adve,
University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA

We present Illinois Extended Reality testbed (ILLIXR), the first fully open-source XR system and research testbed. ILLIXR enables system innovations with end-to-end co-designed hardware, compiler, OS, and algorithms, and driven by end-user perceived Quality-of-Experience (QoE) metrics. Using ILLIXR, we provide the first comprehensive quantitative analysis of performance, power, and QoE for a complete XR system and its individual components. We describe several implications of our results that propel new directions in architecture, systems, and algorithms research for domain-specific systems in general, and XR in particular.

With the end of conventional CMOS scaling, domain-specific acceleration has emerged as a key architectural technique to meet the requirements of emerging applications. A parallel trend is the rise of new application domains increasingly deployed on resource-constrained edge devices, where they interface directly with the end-user and the physical world (e.g., robotics, virtual reality, and autonomous vehicles). In response to these trends, our research conferences have seen an explosion of papers on efficient accelerators.

To truly achieve the promise of efficient domain-specific edge computing, however, will require architects to broaden their portfolio from individual domain-specific accelerators to domain-specific *systems*. Such systems may consist of multiple interacting subdomains (or applications), requiring multiple accelerators that interact with each other to collectively meet end-user demands. Architects must also be cognizant of the programming stack that must grapple with this heterogeneity and the runtime that must manage the heterogeneous architectural resources. Meeting the end-user quality demands of such systems will likely require co-designing the hardware, compiler, and runtime along with the application.

The principled design of such systems requires a new approach to architecture research, based on a generalizable and scalable science for designing end-to-end quality-driven and end-to-end hardware–software-application co-designed domain-specific systems. The foundation for such a science rests upon the availability of system testbeds that can enable an understanding of the end-to-end requirements of such application domains, and enable prototyping and experimentation of innovative techniques to meet these requirements.

We argue that virtual, augmented, and mixed reality, collectively referred to as extended reality (XR), is a domain of increasing societal importance that is in need for architects to embrace such research, and provides the foundation to enable such research.

CASE FOR XR AS A DRIVING DOMAIN

1) *Pervasive:* XR is envisioned to be the next interface for most of computing and to transform many aspects of our lives; e.g., teaching, medicine, science, and entertainment.
2) *Challenging demands:* While XR systems exist today, they are far from providing a tetherless experience approaching the perceptual abilities of humans. There is a gap of several orders of magnitude between what is needed and achievable in performance, power, and usability, giving systems

Digital Object Identifier 10.1109/MM.2022.3161018
Date of publication 24 March 2022; date of current version 30 June 2022.

TABLE 1. Ideal requirements of VR and AR versus state-of-the-art devices, Varjo VR-3 for VR and Microsoft HoloLens 2 for AR.

Metric	Varjo VR-3	Ideal VR	Microsoft HoloLens 2	Ideal AR
Resolution (MPixels)	15.7	200	4.4	200
Field-of-view (Degrees)	115	Full: 165×175 Stereo: 120×135	52 diagonal	Full: 165×175 Stereo: 120×135
Refresh rate (Hz)	90	90–144	120	90–144
Motion-to-photon latency (ms)	< 20	< 20	< 9	< 5
Power (W)	N/A	1–2	> 7	0.1–0.2
Silicon area (mm^2)	N/A	100–200	> 173	< 100
Weight (grams)	944	100–200	566	10 s

Notes: We identified the values of the various aspirational metrics through an extensive survey of the literature.[2–5] VR devices are typically larger and so afford more power and thermal headroom. The Varjo VR-3 offloads most of its work to an attached server, so its power and area values are not meaningful. The ideal case requirements for power, area, and weight are based on current devices, which are considered close to ideal in these (but not other) respects—Snapdragon 835 in the Meta Quest VR headset and APQ8009w in the North Focals small AR glasses.

researchers a potentially rich space to innovate. Table 1 summarizes various system-level quality-related metrics for the state-of-the-art XR devices and the aspiration for ideal future devices.

3) *Multiple and diverse components:* XR involves several diverse subdomains—e.g., vision, robotics, graphics, machine learning, optics, audio, and video—making it challenging to design a system that executes each one well within available resources.
4) *Full-stack implications:* The combination of real-time constraints, complex interacting pipelines, and ever-changing algorithms creates a need for full-stack optimizations involving the hardware, compiler, operating system, and algorithm.
5) *Flexible accuracy for end-to-end user experience:* The end user, being a human with limited perception, enables a rich space of accuracy-aware resource tradeoffs, but requires the ability to quantify impact on end-to-end experience.

CASE FOR AN XR SYSTEM TESTBED

A key obstacle to architecture research for XR is that there are no open-source benchmarks providing the entire XR workflow. As we move from the era of general purpose, homogeneous cores-on-chip to domain-specific, heterogeneous *system*-on-chip architectures, benchmarks need to follow the same trajectory. While previous benchmarks comprising suites of independent applications (e.g., PARSEC, Rodinia, SPEC, and SPLASH) sufficed to evaluate general-purpose architectures, there is now a need for a *full-system benchmark* methodology, better viewed as a *full system testbed*, to design and evaluate system-on-chip architectures. Such a methodology must bring together the diversity of components that will interact with each other in the domain-specific system and also be extensible to accept future new components.

An XR full-system benchmark or testbed will continue to enable traditional research for accelerating a given XR component with conventional PPA metrics, but will additionally allow evaluations for the end-to-end system impact. More importantly, the integrated system will enable new research that co-designs acceleration of its multiple, diverse, and demanding components with the full-system stack, driven by end-to-end user experience.

MORE IMPORTANTLY, THE INTEGRATED SYSTEM WILL ENABLE NEW RESEARCH THAT CO-DESIGNS ACCELERATION OF ITS MULTIPLE, DIVERSE, AND DEMANDING COMPONENTS WITH THE FULL-SYSTEM STACK, DRIVEN BY END-TO-END USER EXPERIENCE.

We present 1) Illinois Extended Reality testbed (ILLIXR),[1] the first fully open-source XR system and testbed consisting of the state-of-the-art XR components orchestrated by a modular and extensible communication interface and runtime; 2) the first detailed quantitative characterization of performance, power, and Quality-of-Experience (QoE) metrics for a complete XR system on desktop and embedded class machines; and 3) several resulting future directions for architecture and systems research that are enabled by ILLIXR.

ILLIXR SYSTEM

Figure 1(a) presents ILLIXR. It contains three interacting pipelines—perception, visual, and audio—each with state-of-the-art components that belong to a modern XR runtime (e.g., Meta VR) and are shipped with an XR headset. The ILLIXR components interact with each other through the ILLIXR communication interface and runtime illustrated in Figure 1(b). ILLIXR supports the emerging and increasingly popular Khronos OpenXR API for XR applications (e.g., games running on a game engine, which in turn runs on

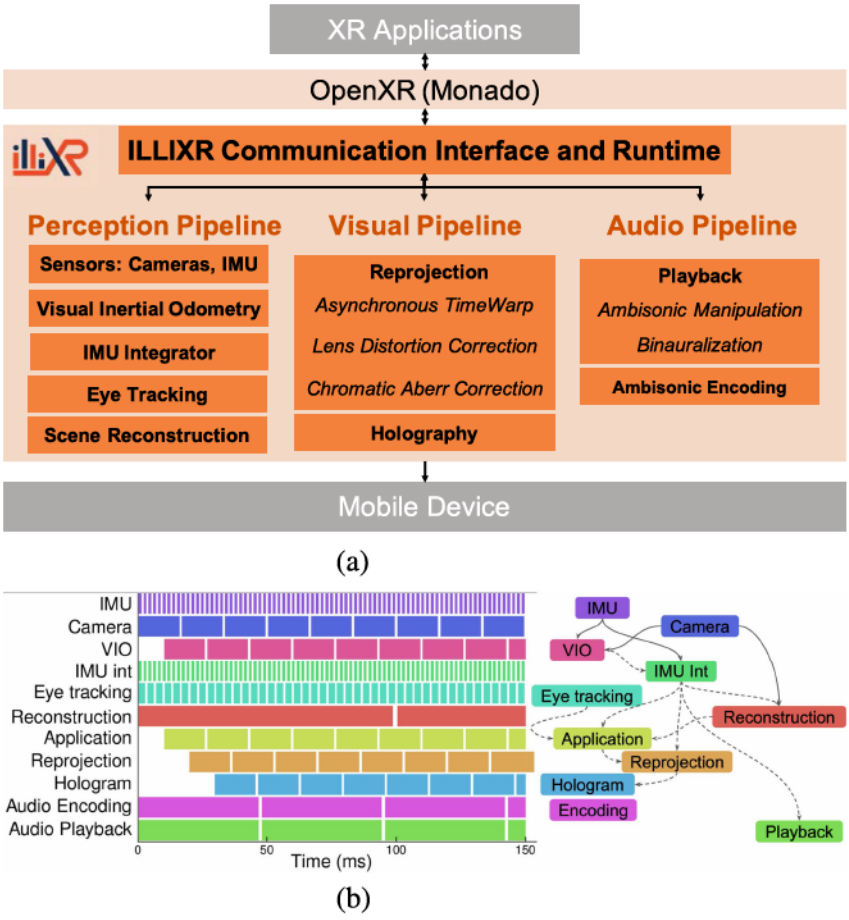

FIGURE 1. (a) ILLIXR system and its relation to XR applications, the OpenXR interface, and mobile platforms. (b) Interactions between ILLIXR components. The left part shows an ideal ILLIXR execution schedule and the right part shows intercomponent dependencies that the scheduler must maintain. Solid arrows are synchronous and dashed arrows are asynchronous dependencies.

ILLIXR), by leveraging the open-source OpenXR interface from the Monado project. When we started this work, the XR ecosystem was closed behind company walls—we distilled the ILLIXR workflow from multiple sources, including conversations with several industry and academic domain experts.

Pipelines

The perception pipeline translates the user's physical motion and surrounding world into information understandable to the system. It uses input from sensors [e.g., cameras and inertial measurement units (IMUs)] and consists of components, such as visual inertial odometry (VIO) for obtaining low frequency but precise estimates of the user's pose (the position and orientation of their head), IMU integration for high-frequency pose estimates, eye tracking for the user's gaze, and scene reconstruction to build a 3-D model of the user's surroundings. *The visual pipeline* obtains the user's new pose from the perception pipeline and the frame from the rendered application and produces the final display, after compensating for rendering latency and optical distortions. ILLIXR supports computational holography, but until holographic displays are more widely available, ILLIXR displays its frames on a standard LCD monitor or the North Star AR headset (an open source headset display). *The audio pipeline* generates 3-D spatial audio using the pose from the perception pipeline.

Runtime and Communication Framework

Figure 1(b) shows an ideal timeline for different XR components (left) and their dependencies (right). On the left, each colored rectangle represents the component's period—ideally, the component would finish execution

TABLE 2. Key ILLIXR parameters that required manual system-level tuning.

Component	Parameter Range	Tuned	Deadline
Camera and VIO	Frame rate = 15–100 Hz Resolution = VGA – 2 K Exposure = 0.2–20 ms	15 Hz VGA 1 ms	66.7 ms – –
IMU and IMU integrator	Frame rate = ≤800 Hz	500 Hz	2 ms
Display, visual pipeline, and application	Frame rate = 30 – 144 Hz Resolution = ≤2 K Field-of-view = ≤ 180	120 2 K 90	8.33 ms – –
Audio encoding and playback	Frame rate = 48–96 Hz Block size = 256–2048	48 Hz 1024	20.8 ms –

Notes: Several other parameters were tuned at the component level.

before its next invocation. The right-hand side illustrates the components' interactions through their synchronous (solid) and asynchronous (dashed) dependencies. An XR system is unlikely to follow the idealized schedule due to shared and constrained resources and variable running times. Thus, an explicit runtime is needed for effective resource scheduling while maintaining inter-component dependencies, resource constraints, and QoE. The ILLIXR runtime schedules resources while enforcing dependencies among components, in part deferring to the native (Linux) kernel and GPU driver.

For extensibility and modularity, ILLIXR provides a well-defined communication framework, with components implemented as plugins. The framework is structured around *event streams*, each supporting writes and asynchronous and synchronous reads, implemented via copy-free shared memory for efficiency. Plugins are distributed as shared-object files and, for modularity, are given access to other plugins only through event streams. A plugin is interchangeable with another as long as it complies with the event-stream interface. Researchers can test alternative implementations of a plugin without needing to reinvent the rest of the system. Development can iterate quickly because plugins are compiled independently.

Metrics

ILLIXR provides several metrics for evaluation. In addition to conventional performance metrics, such as per-component frame rate and execution time, ILLIXR reports several QoE metrics, such as motion-to-photon latency (a standard measure of lag between user motion and image updates), and SSIM and FLIP for image quality. While ILLIXR currently implements SSIM and FLIP, its pose and image collection infrastructure is generic and extensible, enabling evaluation of other (evolving) metrics for image or video quality. This is important as such metrics for XR are still an active area of research.

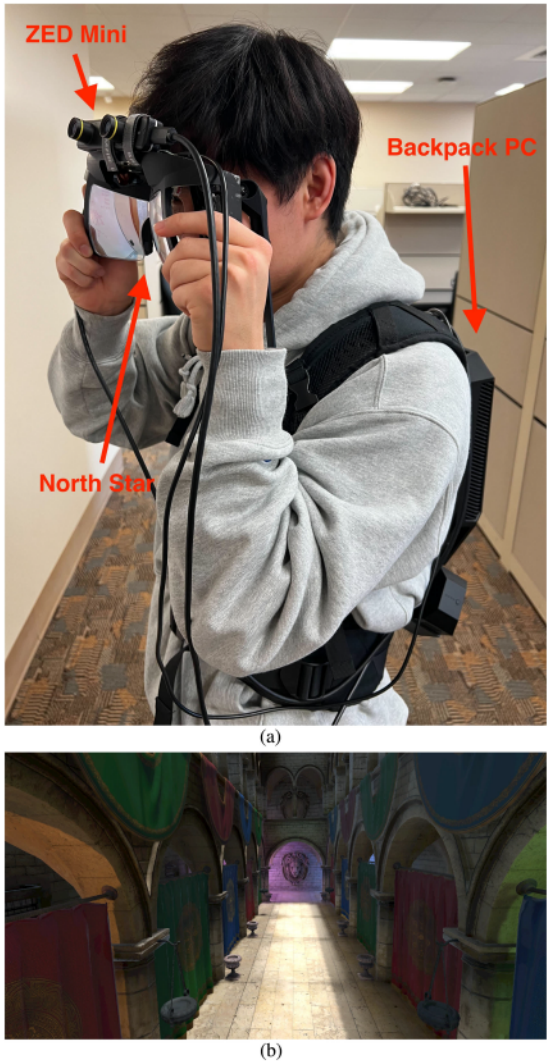

FIGURE 2. (a) Example hardware setup running ILLIXR. This setup shows a North Star AR headset (used only as a display) with a ZED Mini camera mounted on top, connected to a backpack PC running ILLIXR. (b) User's view of the application (only the left eye is shown).

EXPERIMENTAL METHODOLOGY

We chose to run ILLIXR on two hardware platforms, with three total configurations: 1) a high-end desktop

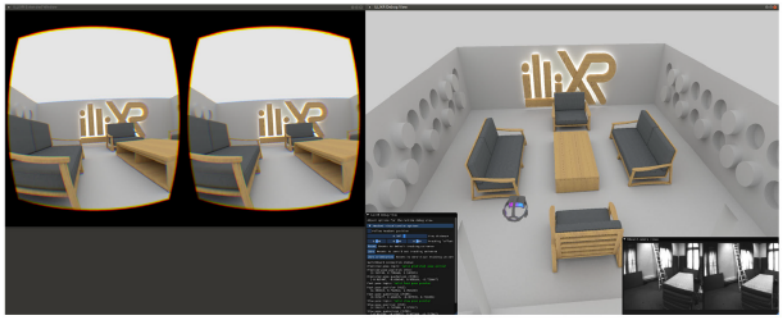

FIGURE 3. Left: Stereoscopic images presented to the user. Right: Third-person debug view showing the virtual environment and the pose of the headset within it. The debug view also shows the tracking camera's stereo images on the bottom right and detailed pose information on the bottom left.

platform; 2) an NVIDIA Jetson AGX Xavier in both high performance; and 3) low-power mode representing a broad spectrum of power-performance tradeoffs and current XR devices. Table 2 summarizes the key parameters for ILLIXR that require system-level tuning, the range available in our system for these parameters, and the final value we chose for our experiments.

For the perception pipeline, we connect a ZED Mini camera to the abovementioned platforms via a USB-C cable. For the visual pipeline, we run representative VR and AR applications—Sponza, Materials, Platformer, and a custom AR demo application with sparse graphics—on the Godot game engine on ILLIXR; these applications interact with the perception pipeline to provide the visual pipeline with the image frames to display. ILLIXR can display the (corrected and reprojected) images on both a desktop LCD monitor and a North Star AR headset connected to the abovementioned hardware platforms. For the audio pipeline, we use prerecorded input. Figure 2(a) shows the setup with a North Star AR headset (used only as a display) and a ZED Mini camera attached to a backpack PC running ILLIXR. Figure 2(b) shows a frame seen by the user. Figure 3 shows a development set up with a desktop LCD monitor and a simple application—it shows the stereoscopic images that ILLIXR presents to the user as well as a third-person debug view showing the virtual and physical environment.

The end-to-end integrated ILLIXR configuration in our experiments uses the components shown in Figure 1(a) except for scene reconstruction, eye tracking, and hologram. The OpenXR standard only recently added an interface for an application to use the results of scene reconstruction and eye tracking. We, therefore, do not have any applications available to use these components in an integrated setting. Although we can generate holograms, we do not yet have a holographic display.

RESULTS AND IMPLICATIONS FOR ARCHITECTURE AND SYSTEMS RESEARCH

Architects have embraced specialization, but most research focuses on accelerators for single programs. ILLIXR is motivated by research for specializing an entire domain-specific system. Our results characterize the end-to-end performance, power, and QoE of an XR device, exposing new systems research opportunities and demonstrating ILLIXR as a unique testbed to enable exploration of these domain-specific systems, as follows. (The results are described in more detail in Huzaifa *et al.*'s work.[1])

Performance, Power, and QoE Gaps

Figures 4–6 quantitatively show that collectively there is a several orders of the magnitude performance, power, and QoE gap between current representative desktop and embedded class systems and the goals in Table 1. The gap will be further exacerbated with higher fidelity displays and more components for a more feature-rich XR experience (e.g., scene reconstruction, eye tracking, hand tracking, and holography).

While the presence of these gaps itself is not a surprise, we provide the first such quantification and analysis. This provides insights for directions for architecture and systems research (in the following section?) as well as demonstrates ILLIXR as a one-of-a-kind testbed that can enable such research.

Component and Task Diversity

Figure 4(d) and results from our original paper[1] show that no one component dominates all the metrics of interest. Thus, to close the aforementioned gaps, *all* components have to be considered together, even those that may appear relatively inexpensive in terms of one metric at

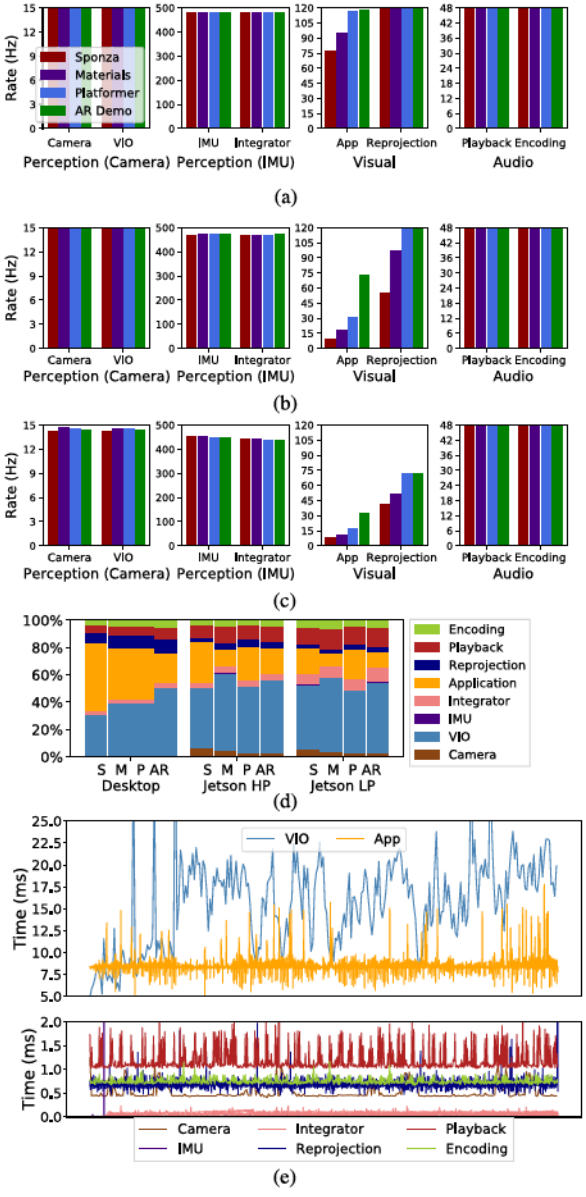

FIGURE 4. (a)–(c) Average frame rate for each component in the different pipelines for each application hardware platform (Desktop, Jetson-HP, and Jetson-LP, respectively). The *y*-axis is capped at the target frame rate of the pipeline, specified in Table 2. (d) Contributions of ILLIXR components to CPU time for different applications and hardware. S = Sponza, M = Materials, P = Platformer, and AR = AR Demo. (e) Per-frame execution times for Platformer on the desktop. The top graph shows VIO and the application. The bottom graph shows the remaining components. Note the different scales of the *y*-axes.

first glance. Moreover, there is a diversity of tasks within and across components, and no single-task dominates. It is likely impractical to build a unique accelerator for every task given the large number of tasks and the severe power and area constraints for XR devices: leakage power will be additive across accelerators, and interfaces between these accelerators and other peripheral logic will further add to this power and area (we identified 27 tasks across all components, and expect more tasks with new components). At the same time, our results show that a number of common primitives exist across components, making the case for shared hardware across components.

Full-System Power Contributions

Figure 5 shows that addressing the power gap requires considering *system-level* hardware components, such as display and other input/output (I/O), including numerous sensors, as Sys power constitutes more than 30% of total power on Jetson-LP. While we do not measure individual sensor power, it is included in Sys power on the Jetson. This motivates research in unconventional architecture paradigms, such as on-sensor computing to save I/O power; e.g., the image processing tasks of VIO can be moved to the sensor so that only detected features and not entire camera frames are sent from the camera to the System-on-Chip (SoC).

Variability

Figure 4(e) shows large variability in per-frame processing times in many cases, either due to inherent input-dependent nature of the component (e.g., VIO) or due to the resource contention from other components. This variability poses challenges to, and motivates research directions in, scheduling and resource partitioning and allocation of shared hardware resources. The components exhibit a variety of memory access patterns, which further complicates the design of shared resources. For instance, several components are memory bandwidth bound while others are more sensitive to memory latency, making it challenging to design a memory system that supports both types of components efficiently.

RESEARCH ENABLED BY ILLIXR

This work provides the research community with a novel, one-of-a-kind infrastructure and foundational quantitative analyses to enable a new era of research in domain-specific *systems* in general and XR systems in particular. We expect this work to shape a broad and long-term research agenda of the science of designing end-to-end quality-driven and end-to-end hardware–software-application co-designed domain-specific systems. ILLIXR and the research it can enable have the potential to particularly transform XR, an emerging domain of critical importance and likely to transform all the endeavors of human activity. We

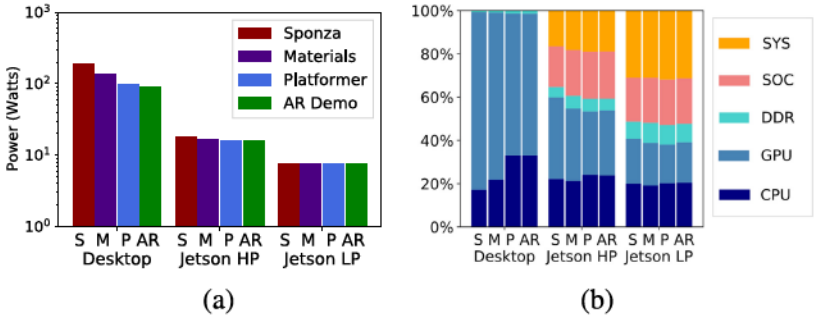

FIGURE 5. (a) Total power (note log scale) and (b) relative contribution to power by different hardware units for ILLIXR running each application on each hardware platform. S = Sponza, M = Materials, P = Platformer, and AR = AR Demo. SoC power includes on-chip microcontrollers but excludes CPU and GPU power. Sys power includes display, storage, and I/O.

highlight several research projects already using or considering using ILLIXR.

Representing Heterogeneous Parallelism in Software

The compiler intermediate representation (IR) is critical for performance and portability. Traditional IRs, such as LLVM, do not capture the parallelism or heterogeneity that is prevalent in current hardware architectures. HPVM[6] is a compiler IR that uses a hierarchical dataflow graph (with side effects) to capture several types of parallelism: task, streaming, nested, data, and fine-grained vector parallelism. The hierarchical dataflow graph nodes naturally and flexibly map to potentially heterogeneous compute elements, and the edges represent communication between the elements. There is an ongoing effort to compile all of ILLIXR to HPVM, providing a rich representation to develop techniques for a compiler and runtime to perform automated accelerator selection, software and hardware approximations, and local and distributed resource mapping for XR and other similar complete domain-specific systems (e.g., Zacharopoulos *et al.*'s work[7]).

Automated Selection and Generation of Accelerators

ILLIXR is being used to develop techniques for automated accelerator selection for complex domain-specific system workloads, with tight performance, power, and area budgets. In initial work, the Trireme tool uses the HPVM IR representation to guide a design space exploration that considers the exposed loop, task, and streaming parallelism to select accelerators for a subset of ILLIXR components.[7] Moving forward, we plan on developing compiler analyses and transformations to determine common compute patterns across components to enable accelerator reuse, and automatically generate accelerator hardware and software.

Accelerator Communication Interface

Future SoCs will require multiple heterogeneous accelerators running in parallel to meet QoE of domains, such as XR. A shared memory programming model would be able to alleviate the need for programming complex DMA engines to explicitly orchestrate data movement between these accelerators. However, how to design cache coherence protocols, memory consistency models, and the memory and communication fabric of the SoC are open questions regarding the implementation of shared memory in a heterogeneous environment. To answer these questions, we are building upon the Spandex[8] heterogeneous coherence interface for coherence specialization, and using ILLIXR's diverse range of communication patterns to drive the design.

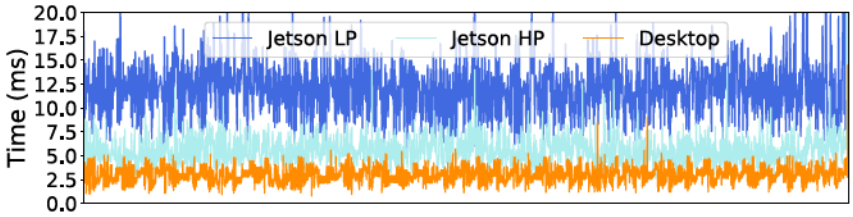

FIGURE 6. Motion-to-photon latency per frame of AR Demo. Target latency is 5 ms.

QoE-Driven Automated Approximation Selection

Current approximation techniques typically look at component-level or subsystem metrics, whereas, it is the end-to-end QoE that ultimately matters in many emerging domains. Determining whether and how a certain approximation will impact the end-to-end QoE, how errors will be composed across components, and how to tradeoff accuracy among components are all unanswered questions. To answer the latter question, we are studying how eye-tracking accuracy impacts foveated rendering[a] (a new ILLIXR component currently under development). Our eventual goal is to develop disciplined end-to-end QoE-driven approximation techniques and methodologies similar to Sharif *et al.*'s work.[9]

QoE-Driven Scheduling

In QoE-driven emerging domains, tasks have to be scheduled and resources managed to meet one or more QoE metrics. ILLIXR is well-suited for studying this phenomenon, as ILLIXR's task graph is a directed acyclic graph (DAG) with multiple critical paths and QoE constraints. ILLIXR is being used to develop a scheduler that automatically determines an optimal frame rate of each component and schedules components to meet QoE for a given hardware mapping. In the future, we aim to study the effects of approximations on the schedule, and determine minimum acceptable component frequencies in order to reduce resource utilization and power consumption.

Edge-Cloud Resource Partitioning

At the moment, all ILLIXR components run on the edge device. However, it is neither feasible to run all components on the edge device due to the limited power budget nor required since some components are inherently latency tolerant and can be offloaded to an edge server or the cloud. We are evaluating which components can be offloaded and how in a collaboration integrating ILLIXR with FleXR, an edge-assistance framework for XR.[10] Our end goal is to develop a methodology for offloading-driven hardware–software-algorithm co-design and perform real-time offloading decisions based on device resource usage and network capabilities.

[a]A rendering technique that renders the center (fovea) of the image at full fidelity and the periphery at lower fidelity to save time and energy.

Multiparty XR

Currently, ILLIXR supports a single end-user device. The full potential of XR is in multiuser applications, such as telepresence. We are expanding ILLIXR to support networked multiparty applications in collaborations integrating ILLIXR with FleXR and with ARENA, a distributed XR system that enables multiple users to have shared XR experiences.[11] The end result will be a full-stack multiparty XR system that will be fully open source, enabling the study of distributed XR applications and allowing for cross-stack optimizations.

Other Research Directions

There are several other projects using or considering ILLIXR, including for AR security and privacy, low-latency networks for XR, use of integrated sensing and compute through 2.5- and 3-D packaging techniques, QoE metrics for XR, cross-component co-design driving novel XR algorithms, and simulation techniques to use ILLIXR to drive novel architectures. An independent group has already published a paper in a top conference using ILLIXR for hologram acceleration.[12]

ILLIXR CONSORTIUM

We have worked with many companies and academics to develop ILLIXR. The culmination of these interactions was the recent launch of the industry-backed ILLIXR consortium.[b] The consortium aims to democratize XR systems research, development, and benchmarking by:

1) evolving ILLIXR into a consensus *reference* open-source end-to-end XR system testbed backed by industry and the academic community;
2) providing a reference benchmarking methodology for XR systems, including reference system configurations, applications, datasets, and metrics; and
3) creating a community where the multidisciplinary XR systems R&D stakeholders come together.

The consortium already has several industry members—Arm, Meta Reality Labs, Micron, NVIDIA, and Project North Star—and we are in discussions with others. The advisory board also consists of several academics spanning the cross-disciplinary boundaries required for such work (in addition to architects).

[b][Online]. Available: https://illixr.org/

ILLIXR received a 2021 NSF CISE Community Research Infrastructure grant to expand the project to multiple software and hardware platforms, add curated datasets and applications, and engage the broader XR community. The ILLIXR consortium will manage ongoing development and community contributions.

CONCLUSION

This article presents ILLIXR, the first open-source full-system XR testbed for driving future end-to-end QoE-driven, co-designed architecture, systems, and algorithm research. Although ILLIXR already incorporates a representative workflow, research is already exposing new frontiers, such as the integration of ML and hybrid low-latency edge-cloud applications. Through the ILLIXR consortium, we envision ILLIXR will become a community resource that continues to evolve, providing an increasingly comprehensive resource to solve the most difficult challenges of QoE-driven domain-specific system design in general and XR in particular.

THIS ARTICLE PRESENTS ILLIXR, THE FIRST OPEN-SOURCE FULL-SYSTEM XR TESTBED FOR DRIVING FUTURE END-TO-END QoE-DRIVEN, CO-DESIGNED ARCHITECTURE, SYSTEMS, AND ALGORITHM RESEARCH.

ACKNOWLEDGMENTS

The authors would like to thank Ameen Akel, Wei Cui, Aleksandra Faust, Liang Gao, Rod Hooker, Matt Horsnell, Amit Jindal, Steve LaValle, Steve Lovegrove, David Luebke, Andrew Maimone, Vegard Øye, Maurizio Paganini, Martin Persson, Archontis Politis, Eric Shaffer, Paris Smaragdis, and Chris Widdowson. The development of ILLIXR was supported in part by the Applications Driving Architectures Research Center, a JUMP Center co-sponsored by SRC and DARPA, the National Science Foundation under Grants CCF 16-19245 and 21-20464, and in part by the Google Faculty Research Award. The development of ILLIXR was also aided by generous hardware and software donations from Arm and NVIDIA.

REFERENCES

1. M. Huzaifa *et al.*, "ILLIXR: Enabling end-to-end extended reality research," in *Proc. IEEE Int. Symp. Workload Characterization*, 2021, pp. 24–38.
2. D. Kanter, "Graphics processing requirements for enabling immersive VR," White Paper, 2015. [Online]. Available: http://developer.amd.com/wordpress/media/2012/10/gr_proc_req_for_enabling_immer_VR.pdf
3. M. McGuire, "Exclusive: How NVIDIA research is reinventing the display pipeline for the future of VR, part 2," 2017. [Online]. Available: https://www.roadtovr.com/exclusive-nvidia-research-reinventing-display-pipeline-future-vr-part-2/
4. M. S. Elbamby, C. Perfecto, M. Bennis, and K. Doppler, "Toward low-latency and ultra-reliable virtual reality," *IEEE Netw.*, vol. 32, no. 2, pp. 78–84, Mar./Apr. 2018.
5. D. Takahashi, "Oculus chief scientist Mike Abrash still sees the rosy future through AR/VR glasses," Sep. 2018. [Online]. Available: https://venturebeat.com/2018/09/26/oculus-chief-scientist-mike-abrash-still-sees-the-rosy-future-through-ar-vr-glasses/
6. M. Kotsifakou *et al.*, "HPVM: Heterogeneous parallel virtual machine," in *Proc. 23rd ACM SIGPLAN Symp. Princ. Pract. Parallel Program.*, 2018, pp. 68–80.
7. G. Zacharopoulos *et al.*, "Trireme: Exploring hierarchical multi-level parallelism for domain specific hardware acceleration," 2022, *arXiv:2201.08603*.
8. J. Alsop, M. Sinclair, and S. Adve, "Spandex: A flexible interface for efficient heterogeneous coherence," in *Proc. ACM/IEEE 45th Annu. Int. Symp. Comput. Archit.*, 2018, pp. 261–274.
9. H. Sharif *et al.*, "Approxtuner: A compiler and runtime system for adaptive approximations," in *Proc. 26th ACM SIGPLAN Symp. Princ. Pract. Parallel Program.*, 2021, pp. 262–277.
10. J. Heo *et al.*, "FleXR." Accessed: Mar. 17 2022. [Online]. Available: https://github.com/GTkernel/FleXR
11. N. Pereira, A. Rowe, M. W. Farb, I. Liang, E. Lu, and E. Riebling, "ARENA: The augmented reality edge networking architecture," in *Proc. IEEE Int. Symp. Mixed Augmented Reality*, 2021, pp. 479–488.
12. S. Zhao *et al.*, "HoloAR: On-the-fly optimization of 3D holographic processing for augmented reality," in *Proc. 54th Annu. IEEE/ACM Int. Symp. Microarchit.*, 2021, pp. 494–506.

MUHAMMAD HUZAIFA is a Ph.D. candidate in computer science at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him at huzaifa2@illinois.edu.

RISHI DESAI is a software engineer with Anduril Industries, Irvine, CA, 92612, USA. As a student at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, he was a member of the ILLIXR development team. Contact him at rdesai@anduril.com.

SAMUEL GRAYSON is a Ph.D. candidate in computer science with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, where is studying academic software engineering. Contact him at grayson5@illinois.edu.

XUTAO JIANG is an AR/VR software engineer with Apple, Los Altos, CA, 94022, USA. As a student at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, he was a member of the ILLIXR development team. Contact him at xutaoj2@illinois.edu.

YING JING is a Ph.D. candidate in electrical and computer engineering with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him at yingj4@illinois.edu.

JAE LEE is currently an undergraduate student in computer science with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him at jael2@illinois.edu.

FANG LU is a GPU ASIC design engineer with Apple, Los Altos, CA, 94022, USA. As a student at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, he was a member of the ILLIXR development team. Contact him at fanglu2@illinois.edu.

YIHAN PANG is a Ph.D. student in computer science with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him at yihanp2@illinois.edu.

JOSEPH RAVICHANDRAN is a Ph.D. student with the Massachusetts Institute of Technology, Cambridge, MA, 02139, USA, where he is studying computer architecture security. Contact him at jravi@mit.edu.

FINN SINCLAIR is a software engineer with Microsoft, Albuquerque, NM, USA. As a student at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, he was a member of the ILLIXR development team. Contact him at finnnorth@gmail.com.

BOYUAN TIAN is a Ph.D. candidate in electrical and computer engineering with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him at boyuant2@illinois.edu.

HENGZHI YUAN is currently working toward the master's degree in computer science with Boston University, Boston, MA, 02215, USA, where he is studying networks and security. As a student at the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA, he was a member of the ILLIXR development team. Contact him at hengzhiy@bu.edu.

JEFFREY ZHANG is currently working toward the master's degree in computer science with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact him jfzhang2@illinois.edu.

SARITA V. ADVE is the Richard T. Cheng professor of computer science with the University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA. Contact her at sadve@illinois.edu.