Portals: Persistent, Editable 4D Spatial World Models on Edge Devices

James A. Tunick[1,2] Ryan Brant[1] Jacob D. Pennock[1] Justin Kasowski[1]

[1]H3M, Inc., USA [2]The IMC Lab, USA

JTunick@TheIMCLab.com Ryan@H3M.ai Jacob@H3M.ai Justin@H3M.ai

H3M Website: https://h3m.ai

Project Page: https://imclab.github.io/portals-cvpr2026/

Abstract

*We present **Portals**, a deployed systems architecture that bridges 4D world-model research and persistent spatial experiences on phones, smart glasses, and augmented-reality headsets. The defining shift in spatial computing is not from 3D to 4D, but from* stateless scenes *to* stateful worlds*—scenes that persist, compound, and stay editable across sessions, devices, and users. Delivering such worlds on constrained hardware is the central systems problem: they must render in real time, survive revisits, and remain authorable through voice, gesture, and no-code tools. Built on 3D Gaussian Splatting [8] and informed by 4D-GS [9] and Generalizable Human Gaussians [14], Portals has been deployed on iOS, with web-based viewers (including Apple Vision Pro) for reconstructed environments, volumetric humans, and holographic spatial media. We contribute: (1) an edge-device runtime built around LOD-adaptive Gaussian splatting (SPAG) and a shared spatial-media compute substrate that fuses depth, stencil, audio, and ML-pose channels, driving 360+ source-agnostic VFX effects at 60 fps on iPhone 14 Pro (2.7–4.1× speedup); (2) a persistent geospatial scene-state architecture with layered world metadata, reloadable scene payloads, and anchor-guided re-alignment across sessions; (3) a creator-facing composition pipeline that bridges reconstruction and generation through VFX composition, voice-driven semantic actions (with on-device intent parsing and cloud fallback for ambiguous utterances), and no-code authoring; and (4) benchmark axes for evaluating 4D world models under deployment constraints such as mobile rendering efficiency, persistent scenes, and editable world state. Prior clinical deployment of volumetric AR at Memorial Sloan Kettering [23] established the real-time rendering primitives underlying this work.*

1. Introduction

The dream of augmented reality—a digital layer seamlessly co-located with the physical world—has been technologically constrained for decades by two coupled problems: *representation* (how do we encode the scene?) and *persistence* (how do we keep it there?). Recent advances in neural scene representations, beginning with the landmark Neural Radiance Field (NeRF) [1] and accelerating through 3D Gaussian Splatting (3DGS) [8], have largely solved the representation problem for static scenes. The frontier has consequently moved to 4D: encoding space *and* time to model the dynamic, ever-changing real world.

This workshop, *4D World Models: Bridging Generation and Reconstruction*, identifies exactly the gap we have been working to close in production: how do 4D scene representations transition from controlled laboratory settings to deployed use on phones and headsets? In this device transition, the research question is no longer only how to model dynamic scenes well in isolation, but how to make those models efficient, persistent, and authorable on edge devices whose thermal, memory, and interaction budgets are dramatically tighter than desktop or server settings. We use *4D* to denote spatially persistent, temporally interactive worlds—content that persists across sessions, responds to real-time input (audio, gesture, voice), and supports live holographic media—complementing the deformation-field approaches discussed in Section 2.

We offer Portals as a worked example from industry that makes these deployment pressures concrete across three settings:

1. **Medical volumetric AR**: Real-time rendering of SPECT/CT volumes on head-mounted displays for COVID-19 diagnosis in collaboration with Memorial Sloan Kettering Cancer Center, Magic Leap, and the NIH National Cancer Institute [23], demonstrating the runtime constraints that edge-native spatial systems must satisfy in practice.

2. **Large live experiences**: Deploying spatial AR overlays for live sports events, virtual concerts, and large-scale live AI motion-capture performances, where synchronized scene state and temporal consistency matter across repeated views and distributed audiences.
3. **Consumer-scale spatial computing**: A deployed edge-device system for geographically anchored AR scenes, enabling creators to publish, revisit, and edit persistent 4D spatial content without expert graphics tooling.

Our central thesis is that the *academic* and *industrial* problems of 4D world modeling are inseparable: algorithmic advances in 4D-GS [9], K-Planes [7], MoSca [10], and dynamic human Gaussians [14] only create value when they can run in real time on commodity edge devices, integrated with persistent location anchors, and exposed to creators through non-expert interfaces. A central deployment challenge is therefore not only how to represent and render 4D scenes, but also how to make them authorable by creators outside expert graphics pipelines. More broadly, we see persistent 4D world models as the operating system of the spatial web: the substrate on which perception, memory, and creator intent compound rather than reset per session. As geometry delivery standardizes around glTF 2.0 and the Khronos `KHR_gaussian_splatting` extension, the open systems problem moves up the stack—to intent capture, scene provenance, and cross-session continuity. The shift is not from 3D to 4D—it is from *stateless scenes* to *stateful worlds*.

Contributions.

- We present **Portals**, a deployed 4D spatial world-model system that unifies scene reconstruction, generation, persistence, and deployment in a single creator-facing pipeline.
- We introduce an edge-device runtime architecture built around LOD-adaptive Gaussian splatting (SPAG) with adaptive quality management and a shared spatial-media compute substrate: demand-driven depth-to-world passes feed 360+ VFX effects spanning depth, stencil, audio, and ML-pose channels, with source-agnostic binding so that live AR and recorded spatial video share a common interface (*runtime on constrained hardware*).
- We describe a persistent geospatial scene-state architecture that survives session boundaries through layered world metadata, reloadable scene payloads, and anchor-guided re-alignment on return visits (*editable world persistence*).
- We show how a creator-facing composition pipeline bridges phone-camera reconstruction, photogrammetry, and AI-driven generation, allowing semantic actions, VFX tooling, voice-driven interaction, and no-code authoring to modify the same deployed spatial scene (*creator-facing world editing*).
- We identify four evaluation dimensions for production 4D world models—mobile rendering efficiency, geospatial alignment accuracy, temporal persistence fidelity, and creator editability—and provide preliminary measurements along each (Tables 2 and 4) (*benchmarks*).

2. Background and Related Work

2.1. From NeRF to 3D Gaussian Splatting

Neural Radiance Fields [1] established the paradigm of representing a scene as a continuous volumetric function mapping (x, y, z, θ, ϕ) to color and density. Subsequent refinements addressed aliasing (Mip-NeRF [2]; Mip-NeRF 360 [3]; Zip-NeRF [4]), and relighting (Ref-NeRF [5]), largely by Barron, Mildenhall, Srinivasan, and colleagues at Google Research. A parallel line of work [6, 7] removed the neural network entirely, replacing it with an explicit sparse 3D grid (Plenoxels) or factored tensor-planes (K-Planes), achieving $100\times$ faster training than the original NeRF while matching visual quality.

3D Gaussian Splatting [8] represents a decisive shift: rather than querying a continuous field, the scene is stored as a finite set of anisotropic Gaussian primitives rasterized by GPU tile-sorting. Real-time frame rates (>30 fps at 1080p on a desktop GPU) became achievable for novel-view synthesis, making 3DGS the de-facto backbone for AR/VR content today. Key quality improvements followed: Mip-Splatting [20] addressed aliasing; Revising Densification [17] (Bulò et al., Meta Reality Labs) improved primitive management through pixel-error-driven adaptive density control; and RayGS from the same group enables hardware-rasterized ray-based rendering at frame rates suitable for mixed-reality headsets.

2.2. 4D Scene Representations

K-Planes [7] extended the planes-factorization idea to the temporal domain: adding a fourth plane xy-t, xz-t, yz-t enables compact, factored representations of dynamic scenes. Dynamic 3D Gaussians [25] demonstrated persistent tracking through dynamic view synthesis, coupling Gaussian primitives with per-timestep optimization. 4D-GS [9] achieved real-time dynamic scene rendering by coupling 3D Gaussians with learned deformation fields conditioned on timestamps—a design that informs the Portals content pipeline. 4K4D [28] pushed real-time 4D rendering to 4K resolution, establishing quality targets for production systems. MoSca [10] from Jiahui Lei et al. (UC Berkeley / GRASP Lab) further demonstrated *4D Motion Scaffolds* enabling dynamic Gaussian fusion from casual monocular video, removing the requirement for multi-camera rigs. Shape of Motion [11] by Qianqian Wang et al. shows that single-video 4D reconstruction is now practical, a capability with direct implications for user-generated content in Portals. 4DGS-1K [32] further demonstrates that temporal redundancy in 4D Gaussians can be eliminated through short-lifespan pruning, achieving 1000+ FPS at $\sim$2% storage cost.

2.3. Dynamic Human Avatars

Rendering animatable humans is a central use case for social AR. Neural Human Performer [12] introduced generalizable performance capture via temporal and multi-view transformers over a parametric body prior (SMPL). DELIFFAS [13] (Youngjoong Kwon, NeurIPS 2023) achieved fast avatar synthesis via deformable light fields. Generalizable Human Gaussians (GHG) [14] (Kwon et al., ECCV 2024, with De la Torre and colleagues) further demonstrated real-time 3DGS-based human novel-view synthesis in a feed-forward manner, providing the building block for live avatar overlays in social AR. Our architecture is designed to accept such feed-forward Gaussian avatars; the current deployment uses depth-driven VFX and skeletal tracking (Section 6). Most recently, Instruct-4DGS [38] enables efficient instruction-guided editing of dynamic Gaussian scenes, validating that editable 4D representations are a timely research direction.

2.4. Generative-Reconstruction Bridges

A key workshop theme is *bridging generation and reconstruction*. Indoor scene understanding, grounded by datasets such as ScanNet [27], has provided the training data enabling this bridge. ReconFusion [15] (Srinivasan et al., CVPR 2024) showed that diffusion model priors can regularize sparse-view 3D reconstruction, dramatically reducing the capture burden—transforming reconstruction from a multi-hour photogrammetry session into a seconds-long phone sweep. L3DG [16] (Kontschieder, Roessle, Dai et al., SIGGRAPH Asia 2024) introduced latent 3D Gaussian diffusion, enabling room-scale scene generation through a VQ-VAE operating in a compressed Gaussian latent space. CUT3R [19] (Wang et al., CVPR 2025 Oral) extends DUSt3R [18] with a recurrent temporal mechanism and persistent state, enabling online incremental 4D reconstruction from video sequences—essential for in-the-wild AR capture. G-CUT3R [35] further integrates depth priors from sensors such as ARKit LiDAR, paralleling Portals' use of device depth. VGGT [34] (CVPR 2025 Best Paper) demonstrated that a single feed-forward transformer can predict cameras, pointmaps, depths, and 3D tracks from unposed images, and ZipMap [36] achieves linear-time stateful reconstruction—both validating the trajectory toward real-time, feed-forward 3D perception that Portals' edge pipeline targets. CAT4D [37] (CVPR 2025) creates 4D content from monocular video via multi-view diffusion models.

A complementary direction pursues persistence through generation rather than reconstruction. Video world models, exemplified by DeepMind's Genie 3[1] and the earlier Genie work [31], generate interactive environments autoregressively at real-time frame rates, achieving visual consistency over multiple minutes of exploration through learned temporal attention—but require cloud-scale compute and provide no explicit geometry. WorldGen [39] takes a text-to-traversable-world approach, generating interactable 3D environments from natural language but likewise targeting server-side rendering. Portals takes the reconstruction-first path: explicit Gaussian representations enable indefinite persistence via stored geometric state, deterministic session re-entry, and per-primitive editing on edge devices. We view these as complementary endpoints of a spectrum this workshop explores: generation-first models excel at open-ended world synthesis, while reconstruction-first systems provide the geometric grounding needed for persistent, multi-user, spatially anchored applications.

[1] DeepMind technical report, "Genie 3: A new frontier for world models," https://deepmind.google/discover/blog/genie-3-a-new-frontier-for-world-models/, Aug. 2025.

2.5. Medical Volumetric AR

Prior to Portals, our team deployed COVIDvis [23], a clinical medical AR system rendering SPECT/CT volumes on Magic Leap 1 at Memorial Sloan Kettering (NIH/NCI P30 CA008748). That work established real-time volumetric rendering on XR hardware, providing the primitives underlying Portals' Gaussian pipeline. Kasowski et al. [30] assessed 227 XR studies for augmenting vision loss, identifying persistent gaps in real-world validation that further motivate Portals' emphasis on deployed, edge-device evaluation. Kasowski's additional experience in low-latency motion capture and cross-platform immersive systems directly informs the avatar pipeline described in Section 7. Pennock, whose background spans immersive theater, video games, and mixed reality with prior collaborations including Facebook and Intel—where he was a first-place winner of the Perceptual Computing Challenge—owns Portals' core platform architecture and real-time interaction pipeline.

3. The Portals Platform

3.1. Architecture Overview

Portals is a cross-platform spatial computing application built on a four-layer architecture spanning mobile capture, real-time rendering, geospatial persistence, and creator-facing authoring. Figure 1 shows representative deployment captures across the device surfaces discussed in this paper. The application shell is React Native 0.81 with Expo 54, communicating with Unity via a bidirectional bridge. Unity 6000.2 handles all 3D/4D rendering, physics, and spatial anchoring. Firebase provides real-time synchronization of geospatially-indexed scene graphs across devices. Across the stack, the practical systems problem is to keep these scene graphs small enough to discover on mobile networks, rich enough to reload as editable worlds, and fast enough to render and modify in situ on phones and headsets.

Figure 1. Representative Portals deployment captures across immersive and mobile spatial-computing surfaces. The same system must support persistent world re-entry, on-device rendering, and creator-facing interaction under edge-device constraints.

3.2. Scene Representation

Each *Portal*—a persistent, geolocated AR scene—is stored as a hierarchical **Spatial Scene Graph** (SSG):

$$\mathcal{S} = \{(G_i, \mathbf{a}_i, \tau_i, \text{LOD}_i)\}_{i=1}^{N} \tag{1}$$

where G_i is a Gaussian primitive set (or mesh, video, or avatar), $\mathbf{a}_i = (\text{lat}_i, \text{lon}_i, \text{alt}_i, \mathbf{R}_i)$ is the geospatial anchor, τ_i is the temporal validity window (enabling content expiration or animation scheduling), and $\text{LOD}_i \in \{\text{high}, \text{medium}, \text{low}, \text{cull}\}$ is the distance-adaptive quality tier.

3.3. Content Ingestion Pipeline

Content enters Portals through three asset pathways: (1) **Gaussian Splatting capture**: users scan an object or environment with their phone camera; the pipeline uploads frames to our GPU cloud cluster, runs Structure-from-Motion (COLMAP), then 3DGS optimization; the resulting `.splat` asset is quantized and uploaded to Firebase Storage. (2) **Photogrammetry**: multi-photo submissions go through MVS to produce mesh+texture assets for static objects. (3) **AI Generation**: text or reference-image prompts invoke third-party generative APIs (e.g., Meshy, Luma) to produce object- or room-scale assets within seconds; latent Gaussian generation in the style of L3DG [16] is identified as future work.

These asset pathways are complemented by multiple *authoring surfaces* that lower to the same persistent scene operations: direct manipulation in the composer, voice-driven semantic edits, and no-code controls. In other words, the ingestion problem in Portals is not only how assets are captured or generated, but how heterogeneous creator inputs are normalized into a shared editable scene graph that can be saved, reloaded, and modified across devices.

3.4. Generative Scene Encoding

A recurring bottleneck in 4D world models is asset size: a single Gaussian Splat scene can exceed 50 MB, making large-scale persistent worlds impractical on mobile networks.

Tier	d range	Scale	Gaussian density
HIGH	0–3 m	1.0×	Full (N primitives)
MEDIUM	3–8 m	0.6×	Pruned ($0.4N$)
LOW	8–25 m	0.3×	Sparse ($0.15N$)
CULL	>25 m	0	Not rendered

Table 1. LOD tier thresholds and Gaussian primitive budgets. The per-tier distance schedule is a design target; the currently deployed runtime uses FPS- and thermal-driven adaptive quality (Table 1) rather than distance-dependent splat switching.

Portals serializes scene state through a portable manifest format (XRAI) with out-of-line references for heavy geometry (Gaussian splats, meshes, video). The compact manifest captures scene structure together with creator-edit history; heavy payloads are resolved on demand from object storage. This separation keeps revisit traffic small while preserving full edit semantics across sessions and devices. The format's design principle is to *store the generation process, not the generated output*, leaving room for procedural encoding extensions in future work.

4. Real-Time LOD-Adaptive Gaussian Splatting on Edge Devices

4.1. Problem Formulation

Mobile GPUs impose hard per-frame budgets: at 60 fps on iPhone 14 Pro (Apple A16) the rendering budget is ≈16.7 ms. Rendering a complex AR scene—multiple Gaussian assets, VFX layers, depth compute, and video overlays—can exceed this budget under sustained thermal load. We address this through an adaptive quality controller that continuously monitors frame rate and thermal state, adjusting rendering fidelity to maintain interactive performance.

4.2. Distance-Adaptive LOD Tiers

Portals uses a four-tier distance-adaptive LOD scheme that prunes Gaussian primitives by viewer distance, using the notation $\text{LOD}_i \in \{\text{HIGH, MEDIUM, LOW, CULL}\}$ from Section 3.2:

In practice, tier selection uses hysteresis around the distance thresholds and is further modulated at runtime by an adaptive quality controller that adjusts URP render scale, shadow distance, VFX particle spawn rates, and depth compute dispatch frequency under sustained thermal load, so the effective per-frame cost tracks both scene distance and device headroom. For Gaussian content specifically, we use Spherical Pixel-Aligned Gaussians (SPAG): a panoramic equirectangular image is projected onto 3D Gaussian primitives with latitude-aware scaling, pole reconstruction, and O(1) per-Gaussian editing via structured JSON payloads. City-scale extensions analogous to LODGE [22] and VastGaussian [24] are planned future work.

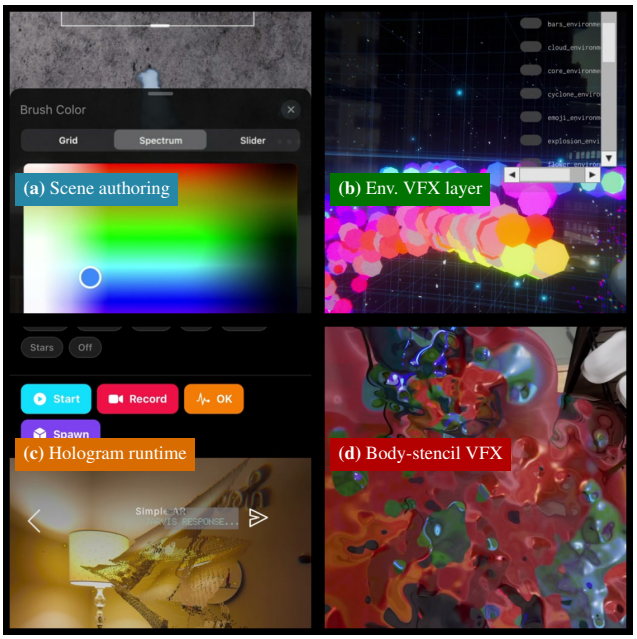

Figure 2. Real in-app captures from Portals on iPhone. **(a)** No-code scene authoring with in-app brush and material controls. **(b)** Environment VFX playback driven by the runtime effect library over a live camera feed. **(c)** Interactive hologram placement and rendering inside the same mobile app shell. **(d)** Body-segmented bubble VFX on a live human subject, demonstrating stencil-masked depth reconstruction. All four modes consume the same shared depth-to-world textures from the runtime compute substrate (Table 2).

4.3. Shared Spatial-Media Runtime Substrate

Beyond adaptive quality control, Portals relies on a shared spatial-media runtime substrate rather than a separate world-space conversion path for each feature. A set of demand-driven GPU compute dispatches per frame (32×32 thread groups, optimized for mobile Metal/Vulkan) transforms AR depth maps into world-space position, hue-encoded depth, stencil, and velocity textures via a shared `DepthToWorld` compute shader; optional passes (edge detection, environment mapping) fire only when bound VFX request them. The resulting buffers, together with optional color capture, audio-reactive bands (4-band FFT + beat detection), and ML pose keypoints (21 hand keypoints via Unity Sentis and MediaPipe), are bound to VFX graphs through a universal property binder that auto-resolves 20+ naming conventions from heterogeneous VFX sources. This alias-based binding enables 360+ production VFX effects — spanning depth-driven particles, body-segmented overlays, audio-reactive visualizations, keypoint-driven pose effects, hologram variants, and combinatorial compositions — to consume the same shared textures without per-effect code changes.

The binding layer presents the same five-property interface (ColorMap, DepthMap, RayParams, InverseView, DepthRange) regardless of whether input is live AR sensor data or decoded spatial video, enabling a record–edit–replay workflow over spatial media.

Device	No LOD	LOD	FPS
iPhone 14 Pro (A16)	38 ms	14.2 ms	60+
iPhone 12 (A14)	81 ms	19.7 ms	48
Galaxy S23 (SD 8G2)	44 ms	15.8 ms	57

Table 2. Median render times with and without LOD management. Target budget: 16.7 ms (60 fps).

In a representative iPhone 14 Pro runtime profile, the shared depth-to-world compute pass contributes a small fixed cost, while additional concurrent effects primarily increase binding and rendering cost. The full pipeline remains within the 16.7 ms frame budget for interactive playback with several simultaneous effects. This $O(1)$ compute scaling is the key systems point: adding effects increases only binding and rendering cost, never the shared depth-to-world computation.

4.4. Performance Results

Table 2 summarises median frame times measured on a representative Portals scene (12 Gaussian assets, 3 video overlays, 1 avatar; identical configuration across all devices, median of 30 consecutive seconds after thermal steady-state) on three device classes:
LOD reduces render time by 2.7–4.1×, enabling 60 fps on current-generation mobile hardware for scenes of practical complexity.

Relation to concurrent work. 4DGS-1K [32] achieves 1000+ FPS on desktop GPUs via temporal Gaussian pruning, and Mobile-GS [33] validates real-time Gaussian rendering on mobile hardware. Portals' contribution beyond raw rendering speed is the full deployment stack—adaptive quality management, persistence, and creator editing—needed to move from benchmark to production.

5. Temporal Coherence through Geospatial Persistence

5.1. World Anchor Architecture

Temporal coherence in 4D world models extends beyond frame-to-frame consistency within a single capture: production AR demands that scenes persist across sessions, devices, and users separated by arbitrary time intervals. A Portal placed at a physical location must be re-discoverable days or weeks later. Our anchoring layer separates *currently deployed* persistence from a *proposed* three-signal architecture. The deployed path uses Firestore-indexed scene discovery combined with ARKit local anchors and ARWorldMap serialization to restore device-local coordinate frames across sessions. The proposed three-signal recovery layer, under active development, combines: (1) GPS/GNSS for coarse localization (±3–10 m); (2) ARKit/ARCore Visual-Inertial Odometry (VIO) for centimetre-scale alignment once the device is within the anchor neighbourhood; (3) a pre-baked

reference Gaussian Splat of the anchor location, against which incoming frames would be registered via photometric alignment inspired by CUT3R's [19] online reconstruction paradigm. Full photometric relocalization against a pre-baked splat is future work.

5.2. Temporal Persistence Schema

Persistence in Portals is not only an anchor table. Each world combines lightweight metadata for discovery and access control with a reloadable scene payload containing the creator-authored spatial state. In the current app stack, world metadata is stored in Firestore, while heavier scene payloads and media assets are resolved through object storage references and fetched lazily on load. This separation keeps discovery queries small while allowing substantially richer scene state than would fit inside anchor metadata alone.

At the format level, this design follows the XRAI[2] pattern: a model- and human-readable scene manifest, heavy geometry or binary data stored out-of-line, and a compact edit history that lets creator intent be replayed across sessions. Temporal validity windows enable time-gated experiences (e.g., a Portal visible only during a concert), while persistent world identifiers allow the same scene to be reloaded across devices and sessions. This architecture directly addresses the ephemerality problem of mobile AR: content is not merely placed in space, but re-hydrated as an editable world state. The persistent edit log also introduces a compounding effect absent from traditional single-session AR: each interaction can incrementally refine the persistent scene state, so that worlds evolve from static captures into environments shaped by the accumulated purpose and intent of their users.

Operationally, this yields a three-tier persistence path that is well matched to edge-device constraints: metadata is small enough for rapid discovery and listing, the edit history preserves creator semantics for revisit workflows, and heavy payloads are hydrated only when the world is opened for playback or editing. A useful production benchmark should therefore distinguish *metadata-only revisit cost* ($<$300 ms in our deployment) from *full payload hydration cost* ($<$1 s on iPhone 14 Pro over LTE), since these correspond to very different user experiences.

5.3. Drift Mitigation

GPS drift of 3–10 m would cause visible mis-registration at table-top scale. We handle this through two-phase activation: *coarse* (50 m geofence pre-loads the SSG) and *fine* (VIO aligns to the reference splat via photometric loss $\mathcal{L}_{\text{align}} = \|\hat{I}_{\text{ref}} - \hat{I}_{\text{live}}\|_1 + \lambda_{\text{SSIM}}\,\mathcal{L}_{\text{SSIM}}$, $\lambda_{\text{SSIM}} = 0.2$ [8]).

[2]XRAI v2.0: An open specification for generative spatial media—a scene-level document for intent, provenance, and edit history, complementing emerging geometry-delivery standards (glTF 2.0 / Khronos `KHR_gaussian_splatting`)—developed by H3M and The IMC Lab.

6. Creator Interfaces for 4D Scene Authoring

6.1. The Creation Gap

A practical 4D world-model system must support content creation by non-specialists. In Portals, creator-facing interfaces layer on top of the reconstruction pipeline, reducing the path from capture to publication to three steps: photograph $\rightarrow$ upload $\rightarrow$ publish.

6.2. AI-Assisted Reconstruction

Drawing on generative-reconstruction bridges such as ReconFusion [15], which uses diffusion priors to complete high-quality 3D models from as few as 3–5 images, our pipeline can produce publishable-quality Gaussian Splats from a 20-second phone video. The diffusion prior compensates for missed viewpoints and occluded surfaces, dramatically reducing the capture burden. Latent Gaussian generation in the style of L3DG [16] is a promising future direction for fully synthetic spatial props; the current pipeline relies on third-party object generation services.

6.3. VFX Composition as a Creator Interface

Portals exposes VFX-oriented controls for compositing dynamic effects into reconstructed or generated scenes. Rather than manipulating low-level scene primitives directly, creators can augment spatial environments with motion, atmosphere, and event-specific overlays through reusable effect templates. In practice, this makes cinematic authoring workflows compatible with deployable 4D spatial content and provides a practical surface for integrating the sparse depth-to-VFX runtime described in Section 4. Portals' editing pipeline operates at a higher semantic abstraction than per-Gaussian editing methods such as Instruct-4DGS [38]: natural-language commands mapped to scene-graph operations, which the two approaches complement.

6.4. Production Deployment Evidence

The same persistence and rendering pipeline has been reused across substantially different deployment settings without changing the underlying scene model: live-event spatial overlays (2020), a spatially persistent virtual concert (2021), real-time AI motion-capture performances (2023), extended-reality stage integration (2023), and early web-based visionOS viewing on Apple Vision Pro (2024). We present these as representative systems evidence rather than controlled benchmarks.

6.5. Voice and No-Code Authoring

Voice interaction in Portals is a semantic action layer over the persistent scene graph. A local intent parser handles common commands (creation, movement, style) on-device, while lower-confidence requests fall through to a cloud LLM. Both paths emit the same structured scene operations used by di-

Intent Category	Tests	Local
Object creation (primitives, colors, counts)	30	100%
Object modification (scale, color, material)	12	100%
Animation (8 types + stop)	9	100%
Formation arrangement (grid, circle, line)	8	100%
Structure generation (tower, castle, forest)	11	85%
Audio wiring (source → target mapping)	7	100%
Lighting (6 named themes + custom)	7	100%
Scene management (clear, undo, redo)	5	100%
Movement (direction + distance)	3	100%
VFX, atmosphere, removal, other	51	88%
Total (14 categories)	**143**	**91%**

Table 3. Voice intent parsing: 143 target utterances across 14 semantic action categories, used as regression test cases against the local regex parser. Mean parse latency < 1 ms (synchronous regex). Low-confidence requests fall through to a cloud LLM. An automated integration-test harness is planned future work.

rect manipulation and no-code editing. A *VoiceToWire* pathway maps natural-language utterances to parametric VFX expressions, enabling audio-reactive 4D behaviors through voice commands.

This layered architecture—fast on-device parsing, response cache, and cloud LLM fallback—enables fully offline-capable creation. Compared with cloud-routed interfaces such as LLMR [29], Portals keeps the common path responsive on the order of milliseconds and at zero per-request cost, with the cloud path reserved for ambiguous or open-ended utterances.

Complementing voice, no-code authoring interfaces expose the same scene operations through composer controls. Taken together, VFX tooling, voice interaction, and no-code authoring expand who can build 4D content, shorten iteration cycles, and create pathways for collecting real-world usage data across deployment settings.

7. Dynamic 4D Content: Human Avatars and Live Events

Our avatar pipeline uses AR depth segmentation and skeletal tracking rather than feed-forward Gaussian synthesis. ARKit's human-occlusion subsystem provides a per-frame depth map filtered by a body stencil mask; a compute shader (`DepthToWorld`) converts this into a world-space position map that drives VFX Graph particle effects (sparks, embers, volumetric trails) anchored to the user's silhouette. A 26-joint skeletal pose stream from ARKit `HumanBodyTracker` retargets onto rigged meshes or particle emitters, enabling real-time motion-captured interactions developed by Justin Kasowski. Generalizable Human Gaussians [14] represent a promising path toward feed-forward 3DGS avatars; our architecture is designed to accept such representations once mobile inference latency permits.

Together, world-anchored Gaussians, depth-driven VFX, and skeletal animation form an interactive *social 4D world* rendered at interactive frame rates on-device.

7.1. Live Holograms and Spatial Media Recording

The same shared runtime substrate also supports live and recorded holographic media. Portals adopts a UV-multiplexed spatial video format inspired by Keijiro Takahashi's MetavidoVFX[3]: a single 1920×1080 video frame packs RGB color (right half), hue-encoded depth (top-left quarter), human stencil (bottom-left quarter, red channel), and a metadata barcode co-located in the same quadrant's blue channel encoding camera pose, field of view, depth range, and projection parameters (12 floats per frame). Because the multiplexed frame is a standard video texture, it can be recorded, streamed, and replayed through commodity H.264/HEVC codecs and standard video players — no bespoke spatial-format infrastructure is required.

On playback, a two-pass GPU fragment shader demuxes the frame into color and depth textures; a compute shader decodes the barcode into camera pose; and the resulting five properties (ColorMap, DepthMap, RayParams, InverseView, DepthRange) bind to VFX graphs through the same property contract used for live AR input. VFX effects consuming only these five properties work interchangeably on live capture and decoded spatial video, enabling a record–edit–replay loop. Effects requiring additional textures (PositionMap, VelocityMap) currently operate on the live AR path only; extending the decoder is active work.

8. Discussion: Open Problems

8.1. Reconstruction Quality vs. Capture Burden

Even with ReconFusion-style priors [15], casual phone video introduces blur, illumination changes, and dynamic objects (moving people, cars) that degrade reconstruction quality. K-Planes [7] handles dynamic objects by factoring space-time planes, but training time (minutes) remains too long for in-situ AR capture. Recent single-video systems such as Shape of Motion [11] and MoSca [10] improve robustness from casual video, but the community still needs real-time or near-real-time 4D reconstruction from monocular capture at the frame rates required for live AR deployment.

8.2. Scene Composition and Editability

Once reconstructed, users need to *edit* 4D scenes: remove unwanted objects, change lighting, insert synthetic content. Current 3DGS representations are notoriously hard to edit due to the lack of explicit semantic structure. Feature-field extensions [21] begin to address this, but compositing a Gaussian avatar into a Gaussian background without seams remains an open challenge.

[3] `https://github.com/keijiro/MetavidoVFX`

Metric	Value	Axis
Metadata-only reload	<300 ms	3
Full-payload hydration (LTE)	<1 s	3
Draft save/reload (objects)	10/10	3
Voice parse latency (local regex)	<1 ms	4
Voice intent success (143 tests)	91%	4

Table 4. Preliminary persistence and editability measurements from deployed Portals iOS builds (axes 3–4). Reload times reflect Firestore metadata fetch and Firebase Storage asset hydration; draft survival measures scene-graph objects persisting across app restarts via XRAI serialization. Anchor-based spatial persistence (axis 2) requires further device-level integration and measurement.

8.3. Privacy in Persistent World Models

Persisting 3D scans at scale raises privacy concerns: bystanders captured inadvertently, interiors scanned without consent, geospatial databases aggregating personal data. Portals enforces client-side ephemeral buffers and uploads only user-selected keyframes. A principled privacy-by-design framework for persistent 4D world models remains open.

8.4. Toward Production Benchmarks for 4D World Models

Existing benchmarks (NeRF Synthetic, Tanks and Temples, MipNeRF-360, DiVa-360 [26]) evaluate reconstruction quality in controlled settings but omit metrics critical for production deployment. We propose four benchmark axes and provide initial measurements for each (Tables 2 and 4): (1) *mobile rendering efficiency* — median and p95 frame times on commodity mobile GPUs (Table 2); (2) *geospatial alignment accuracy* — registration error after session re-entry; (3) *temporal persistence fidelity* — anchor recovery rate, edit survival, and reload cost across return visits; (4) *creator editability* — semantic-action success rate and edit-to-preview latency for voice and no-code authoring. Table 4 reports preliminary measurements from deployed Portals iOS builds for axes 3–4; we invite the community to develop standardized evaluation protocols.

8.5. Closed-Loop Refinement and Extensibility

Portals records creator edits as a serialized history that supports replay and undo/redo. The full closed loop—edits as training signal for incremental Gaussian optimization—remains future work, requiring sparse-edit updates and conflict resolution across collaborators. The Khronos `KHR_gaussian_splatting` extension[4] provides the interoperability layer for cross-platform benchmark evaluation.

9. Conclusion

We have presented Portals as a deployed 4D world-model system for the edge-device era. The paper's core result is not a single isolated representation, but a deployment stack that proves three coupled requirements can be satisfied together: real-time runtime performance on commodity mobile hardware, persistent scene state across revisits, and creator-facing world editing that bridges reconstruction and generation. Deployment experience spanning clinical AR [23], large live experiences, and consumer spatial publishing suggests that these requirements can compose into a viable deployment surface for spatial computing.

As spatial computing shifts to phones and headsets, the bottleneck moves from scene quality to deployable systems architecture. Four principles emerge: *generative encoding* (procedural rules for compression/editability); *layered fallback* (fast local, intelligent cloud, safe degradation); *upstream truncation* (proactive complexity reduction); and *closed-loop refinement* (edits as training signal). The next chapter—real-time monocular 4D reconstruction, semantically-editable Gaussians, and privacy-preserving persistent world maps—requires continued academic-industry collaboration, supported by the production benchmarks we propose. As geometry delivery standardizes, the open systems problem moves from what to render to what to remember—and which intents to preserve across sessions, devices, and collaborators. If the last decade of graphics taught machines to render the world, the next must teach them to remember it—and let those memories compound.

Figure 3. Real in-app creator-authoring captures. **(a)** Brush, Hologram, and Hi-Fi tool selection with placed 3D objects and active VFX particles. **(b)** VFX composition toolbar (Glitch, Bubbles, Trails, HumanCube) over a live AR scene. **(c)** Voice command "add a pink sphere to the portal" parsed and executed locally. **(d)** Direct manipulation of scene objects with active VFX and depth-driven effects. All four interfaces expose the same persistent scene graph through the composition bridge.

Acknowledgements. This work was supported in part by the NIH/NCI Cancer Center Support Grant P30 CA008748 (COVIDvis component). Related earlier AR/AI research was also supported by Magic Leap and Memorial Sloan Kettering

[4] Khronos release candidate, Feb. 2026.

Cancer Center. The authors thank Dr. Omer Aras (Memorial Sloan Kettering Cancer Center) and Dr. Haluk B. Sayman (Istanbul University) for clinical collaboration. We also thank the open-source 3DGS community for tooling, the Apple Vision Pro developer program for early hardware access, and the H3M engineering team for production infrastructure.

References

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of ECCV*, 2020.

[2] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P. P. Srinivasan. Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of ICCV*, 2021.

[3] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Mip-NeRF 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of CVPR*, 2022.

[4] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman. Zip-NeRF: Anti-aliased grid-based neural radiance fields. In *Proceedings of ICCV*, 2023.

[5] D. Verbin, P. Hedman, B. Mildenhall, T. Zickler, J. T. Barron, and P. P. Srinivasan. Ref-NeRF: Structured view-dependent appearance for neural radiance fields. In *Proceedings of CVPR*, 2022.

[6] A. Yu, S. Fridovich-Keil, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of CVPR*, 2022.

[7] S. Fridovich-Keil, G. Meanti, F. Warburg, B. Recht, and A. Kanazawa. K-Planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of CVPR*, 2023.

[8] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (SIGGRAPH)*, 2023.

[9] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. 4D Gaussian splatting for real-time dynamic scene rendering. In *Proceedings of CVPR*, 2024.

[10] J. Lei, Y. Wang, C. Ye, J. Gu, A. Kanazawa, C. Daniilidis, P. Moulon, and G. Pavlakos. MoSca: Dynamic Gaussian fusion from casual videos via 4D motion scaffolds. *arXiv:2405.17421*, 2024.

[11] Q. Wang et al. Shape of Motion: 4D reconstruction from a single video. In *Proceedings of ICCV*, 2025.

[12] Y. Kwon, D. Kim, J. Ceylan, and H. Li. Neural human performer: Learning generalizable radiance fields for human performance rendering. In *Proceedings of NeurIPS*, 2021.

[13] Y. Kwon, L. Liu, H. Fuchs, M. Habermann, and C. Theobalt. DELIFFAS: Deformable light fields for fast avatar synthesis. In *Proceedings of NeurIPS*, 2023.

[14] Y. Kwon, B. Fang, Y. Lu, H. Dong, C. Zhang, F. V. Carrasco, A. Mosella-Montoro, J. Xu, and F. De la Torre. Generalizable human Gaussians for sparse view synthesis. In *Proceedings of ECCV*, 2024.

[15] R. Wu, B. Mildenhall, P. Henzler, K. Park, R. Gao, D. Watson, P. P. Srinivasan, D. Verbin, J. T. Barron, B. Poole, and A. Holynski. ReconFusion: 3D reconstruction with diffusion priors. In *Proceedings of CVPR*, 2024.

[16] B. Roessle, N. Müller, L. Porzi, S. R. Bulò, P. Kontschieder, A. Dai, and M. Nießner. L3DG: Latent 3D Gaussian diffusion. In *SIGGRAPH Asia Conference Papers*, 2024.

[17] S. Rota Bulò, L. Porzi, and P. Kontschieder. Revising densification in Gaussian splatting. In *Proceedings of ECCV*, 2024.

[18] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of CVPR*, 2024.

[19] Q. Wang et al. CUT3R: Continuous 3D perception model with persistent state. In *Proceedings of CVPR*, 2025.

[20] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. Mip-splatting: Alias-free 3D Gaussian splatting. In *Proceedings of CVPR*, 2024.

[21] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister. LangSplat: 3D language Gaussian splatting. In *Proceedings of CVPR*, 2024.

[22] Kulhánek et al. LODGE: Level-of-detail large-scale Gaussian splatting with efficient rendering. *arXiv:2505.23158*, 2025.

[23] H. B. Sayman, K. N. Toplutas, J. Tunick, and O. Aras. ^{99m}Tc-Vitamin C SPECT/CT imaging in SARS-CoV-2 associated pneumonia. *Nuclear Medicine Review*, 25(2):127–128, 2022. DOI: 10.5603/NMR.a2022.0026.

[24] J. Lin, Z. Li, X. Tang, J. Liu, S. Liu, J. Liu, Y. Lu, X. Wu, S. Xu, Y. Yan, and S. Zhang. VastGaussian: Vast 3D Gaussians for large scene reconstruction. In *Proceedings of CVPR*, 2024.

[25] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan. Dynamic 3D Gaussians: Tracking by persistent dynamic view synthesis. In *Proceedings of 3DV*, 2024.

[26] A. Xing, R. Rai, S. Sridhar, et al. DiVa-360: The dynamic visual dataset for immersive neural fields. In *Proceedings of CVPR (Highlight)*, 2024.

[27] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proceedings of CVPR*, 2017.

[28] Z. Xu, S. Peng, H. Lin, G. He, J. Sun, Y. Shen, H. Bao, and X. Zhou. 4K4D: Real-time 4D view synthesis at 4K resolution. In *Proceedings of CVPR*, 2024.

[29] F. De La Torre, C. M. Fang, H. Huang, A. Banburski-Fahey, J. Amores Fernandez, and J. Lanier. LLMR: Real-time prompting of interactive worlds using large language models. In *Proceedings of ACM CHI (Honorable Mention)*, 2024.

[30] J. Kasowski, B. A. Johnson, R. Neydavood, A. Akkaraju, and M. Beyeler. A systematic review of extended reality (XR) for understanding and augmenting vision loss. *Journal of NeuroEngineering and Rehabilitation*, 19(1):1–18, 2022.

[31] J. Bruce, M. Dennis, A. Edwards, et al. Genie: Generative interactive environments. In *Proceedings of ICML*, 2024.

[32] Y. Yuan, Q. Shen, X. Yang, and X. Wang. 1000+ FPS 4D Gaussian splatting for dynamic scene rendering. *arXiv:2503.16422*, 2025.

[33] X. Du, Y. Wang, K. Zhan, and X. Yu. MobileGS: Real-time Gaussian splatting for mobile devices. *arXiv:2603.11531*, 2026.

[34] J. Wang, M. Chen, N. Karaev, A. Vedaldi, C. Rupprecht, and D. Novotny. VGGT: Visual geometry grounded transformer. In *Proceedings of CVPR (Best Paper)*, 2025. *arXiv:2503.11651*.

[35] R. Khafizov, A. Komarichev, R. Rakhimov, P. Wonka, and E. Burnaev. G-CUT3R: Guided 3D reconstruction with camera and depth prior integration. *arXiv:2508.11379*, 2025.

[36] H. Jin, R. Wu, T. Zhang, R. Gao, J. T. Barron, N. Snavely, and A. Holynski. ZipMap: Linear-time stateful 3D reconstruction via test-time training. *arXiv:2603.04385*, 2026.

[37] R. Wu, R. Gao, B. Poole, A. Trevithick, C. Zheng, J. T. Barron, and A. Holynski. CAT4D: Create anything in 4D with multi-view video diffusion models. In *Proceedings of CVPR*, 2025. *arXiv:2411.18613*.

[38] J. Kwon, H. Cho, and J. Kim. Instruct-4DGS: Efficient dynamic scene editing via 4D Gaussian-based static-dynamic separation. *arXiv:2502.02091*, 2025.

[39] D. Wang, H. Jung, T. Monnier, K. Sohn, C. Zou, X. Xiang, Y.-Y. Yeh, D. Liu, Z. Huang, T. Nguyen-Phuoc, et al. WorldGen: From text to traversable and interactive 3D worlds. *arXiv:2511.16825*, 2025.