Canvas Engineering: Declared Causal Macrostructure for Reverse-Diffusion Latent Dynamics

Jacob Valdez* ⌘ CommandAGI

commandagi.com/research/canvas-engineering

July 2026

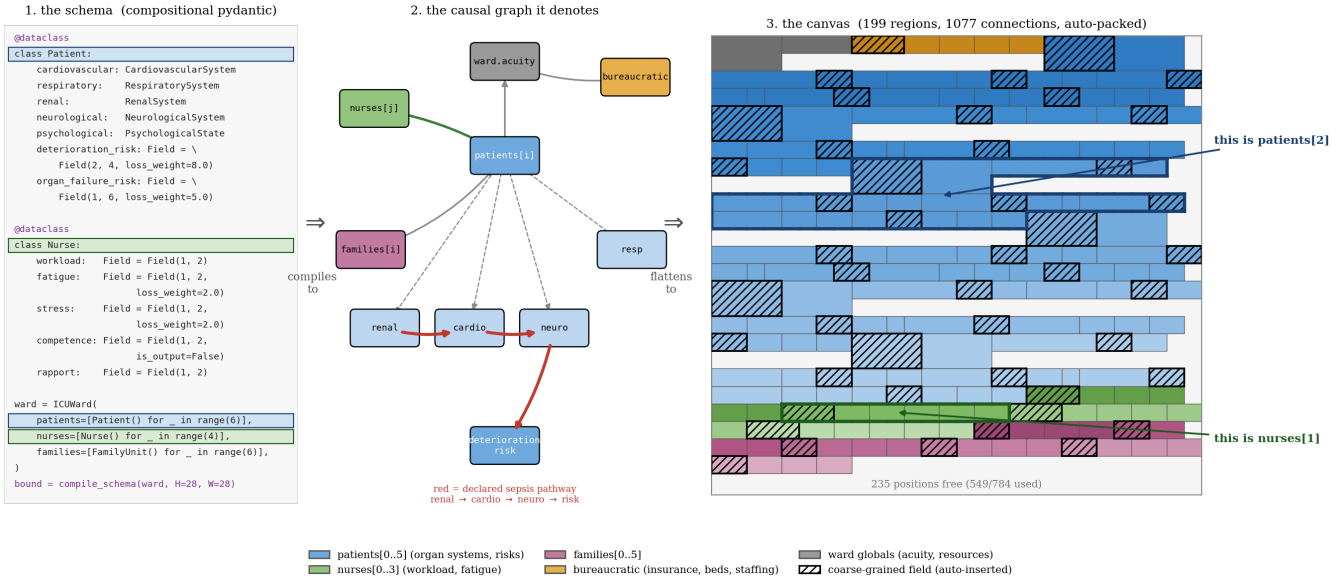

Figure 1: Canvas engineering in one figure. A typed schema compiles, end to end, into a structured latent space. **1:** the declaration—`Patient` and `Nurse` types (abridged) and the `ICUWard` instance passed to the compiler. **2:** the causal graph the schema denotes—composition and coordination edges, with the declared sepsis pathway (`renal`→`cardio`→`neuro`→`deterioration_risk`) highlighted. **3:** the compiler's output—199 typed regions and 1,077 connections auto-packed onto a 28×28 canvas (70% utilization), with no manual placement. Hatched blocks are the coarse-grained fields the compiler inserts at every nesting level; all cross-entity attention is forced through them, so the hierarchy of the schema becomes a hierarchy of bottlenecks on the canvas. Because the schema fixes what every position means, the latent tensor is *pointable*: the outlined runs of blocks are, exactly, `patients[2]` and `nurses[1]`. This is a hospital-ward example (§4); it is an architectural demonstration, not a validated clinical system.

Abstract

Prompt engineering structures what a language model *sees*; canvas engineering structures what a diffusion model *thinks in*. The practitioner declares the macrostructure of a diffusion transformer's latent space—which regions carry which modalities, their geometry, temporal update frequencies, loss participation, and, critically, a directed graph of *permitted* block-to-block attention operations—and a compiler lowers that declaration into attention masks, loss weights, and frame mappings. Because reverse diffusion is an iterated denoising map over the whole canvas, a hard connectivity constraint on attention induces an explicit, human-legible causal interaction graph *inside* the generative dynamics, while gradient descent remains free to shape all fine structure within and along the declared edges. This is a neurosymbolic architecture in a precise sense: transparent symbolic macrostructure—a type system, with offsets, signatures, calling conventions, and an ABI—hosted directly within learnable neural mechanics, with no discrete/continuous interface to cross. We give the underlying mathematics—region index sets, the mask construction, the weighted denoising objective, and the reachability semantics that make an absent edge an exact (marginal) independence—and describe the abstraction stack (regions → topology → typed programs → compiled deployment), including the compilation of nested entity schemas into hierarchically coarse-grained canvases. Worked designs span robot manipulation, a compiler-allocated hospital ICU ward with 199 regions, air-traffic control, and a 23-region cortical model that recovers brain-encoder–estimated cortical dynamics at $R^2 = 0.825$. Early experiments—26 studies, 236 training runs on CogVideoX-2B—include a $1.73\times$ parameter-efficiency result for looped attention ($p < 0.001$) and a frozen 350K-parameter configuration that beats 11.7M unfrozen parameters on action prediction. We are explicit that these are small-scale results and read them as calibration rather than narrative. We argue that *declared* latent structure is the production-ready on-ramp to intuition-guided symbolic world modeling: the symbolic model is not synthesized by the network, it is authored by the engineer, and the network learns everything else. Research and code at https://github.com/CommandAGI/canvas-engineering.

*jacob@commandagi.com

1 Position: two routes to symbolic world models

Chollet's convergence thesis [1] holds that much of AI trends toward *intuition-guided symbolic world modeling*: deep-learning-guided program synthesis, because a symbolic model constructs a compact, reusable, highly generalizable representation of a problem from minimal data [2]. This inherits a long argument that connectionism alone under-generalizes and that structure must re-enter the loop [3, 4]. The thesis names the destination but underdetermines the route.

The dominant proposed route is **synthesis**: a neural policy emits symbolic programs and is rewarded on their execution. It gains compactness and generalization, but pays for a brittle discrete search and—more corrosively—a hard boundary where continuous representations must be quantized into symbols and back. Canvas engineering takes the dual route, **declaration**: the human authors the symbolic model up front, not as a program the network emits but as the *interaction topology of latent space itself*, and the neural substrate fills it in. Both routes buy the same asset, a macrostructure acquired without data. Declaration ships today, because it compiles to nothing more exotic than attention masks and loss weights over an off-the-shelf pretrained diffusion transformer [5, 6].

The framing is economic. Neurosymbolic methods stay research curiosities as long as the marginal dollar buys more from data than from structure—a corollary of the bitter lesson [7]. Declared structure changes that calculus precisely where data is expensive: robotics, control, multi-agent coordination, world modeling [8–10]. Authoring a schema is cheap relative to collecting demonstrations. This paper answers a prior question: *how does one make structure declarable at all* inside a modern generative architecture, and what does it buy empirically? Figure 1 shows the whole pipeline in one image—a typed schema, the causal graph it denotes, and the structured canvas the compiler produces.

Contributions. (i) **Canvas engineering**, a declarative interface in which a practitioner specifies latent regions, their connectivity, temporal frequencies, and loss roles, and a compiler lowers the specification to attention masks, loss weights, and frame mappings on a stock pretrained diffusion transformer (§2). (ii) A **formal account** (§2.3) showing that absent edges yield exact marginal independence in the generative dynamics—graph reachability as the causal semantics of a masked denoiser. (iii) A **compiler** from nested typed entity schemas to hierarchically coarse-grained canvases, and a typed process layer for deployment (§3). (iv) **Worked designs** across robot manipulation, a 199-region hospital ward, air-traffic control, and a cortical model, plus a structural correspondence to cortical organization. (v) An **honest empirical read**, summarized from a companion study [11], that separates what declared structure buys (parameter efficiency, an interface) from what it does not yet (iterative reasoning, free binding).

2 Mechanism

2.1 The canvas

A **canvas** is a 3D spatiotemporal grid (T, H, W) of d-dimensional latent positions, flattened into the token sequence of a diffusion transformer (DiT). A `CanvasLayout` partitions it into named regions (Fig. 2):

```
layout = CanvasLayout(
    T=5, H=8, W=8, d_model=256,
    regions={
        "visual": (0,5, 0,6, 0,6), # 180 pos - video
        "action": (0,5, 6,7, 0,1), # 5 pos - actions
        "reward": (2,3, 7,8, 0,1), # 1 pos - scalar r
    },
    t_current=2, # t >= 2 is future (diffusion output)
)
```

Each region carries a `RegionSpec`: its bounds; a temporal `period` mapping canvas frames to real frames, so a `thought` region at `period=4` and a `perception` region at `period=1` coexist on one canvas; `is_output` and `loss_weight`, which make loss participation type-directed codegen—`loss_weight_mask()` compiles the declaration into a per-position weight tensor; an optional `semantic_type` with a frozen embedding; and a `default_attn` function family. Heterogeneous modalities enter and exit through per-region encoders and decoders; the pretrained video backbone attends over everything. The canvas is thus the omnimodal I/O layer that lets a video model also *do things*—predict actions, read proprioception, estimate reward—rather than only generate pixels.

2.2 Topology: the causal graph is the attention mask

A `CanvasTopology` is a directed multigraph of discrete cross-attention operations. Each `Connection(src, dst)` licenses `src` tokens to query `dst` keys and values; the *absence* of an edge is a hard prohibition, not a soft prior:

```
CanvasTopology(connections=[
  # action queries visual: obs →act
  # information flow (diffusion policy)
  Connection(src="action", dst="visual"),
  # action at t queries visual at t-1
  Connection(src="action", dst="visual",
             t_src=0, t_dst=-1),
  # cross-agent coordination
  Connection(src="r1_cam", dst="r2_cam",
             weight=0.5),
])
```

Temporal offsets $(t_{\text{src}}, t_{\text{dst}})$ restrict which frame pairs participate—same-frame-only, previous-frame causal, sliding window—and a `TemporalFill` mode (`HOLD`/`DROP`/`INTERPOLATE`) resolves cross-frequency queries in real-time space when a `period=1` region queries a `period=4` one. Convenience constructors—`dense` (fully connected), `isolated` (block-diagonal), `hub_spoke` (a shared coordinator), `causal_chain` ($A \to B \to C$), and `causal_temporal` (same-frame self plus previous-frame cross, no future leakage)—name the standard patterns; a standard transformer [12] is the degenerate `dense` case.

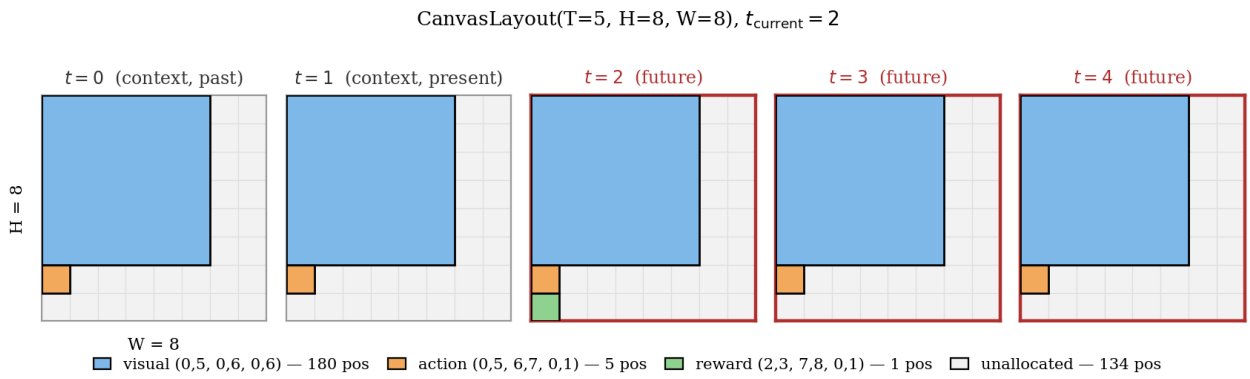

Figure 2: The exact layout declared by the code in §2.1, rendered as its five (H, W) time slices. `visual` occupies the 6×6 block at every frame (180 positions), `action` one position per frame (5), and `reward` a single position at $t = 2$. Frames $t \geq t_{\text{current}} = 2$ are future: denoised by reverse diffusion; earlier frames are clamped context. Unallocated positions (gray) carry no loss and no meaning.

This yields the paper's central claim. Reverse diffusion applies the denoiser to the full canvas at every step [13, 14], so *which regions can influence which* under iteration is exactly the topology's transitive closure. Declaring the topology therefore declares a causal interaction graph over the generative dynamics—attention-based rather than convolutional, so the graph is over *semantic regions* rather than spatial neighborhoods, and non-Euclidean by construction. Diffusion policy [15], observation→action, is the two-node base case. Multi-agent perceptual diffusion—each agent self-attending over its own observations and actions, coordinating only through declared cross-edges—is the same primitive composed.

What is fixed and what is learned splits cleanly. The engineer fixes *whether* an edge exists, its direction, temporal extent, and function family; gradients—from data or intrinsic losses—determine *what flows* along it and everything within regions. Macro is symbolic, micro is neural, and there is no interface to cross, because the symbolic layer *is* the mask.

2.3 Core mathematics

Four small pieces of arithmetic carry the whole construction.

Regions are index sets. A canvas has $N = T \cdot H \cdot W$ positions. A region r with bounds $(t_0, t_1, h_0, h_1, w_0, w_1)$ owns

$$I_r = \{\, tHW+hW+w \;:\; t_0 \leq t < t_1,\ h_0 \leq h < h_1,\ w_0 \leq w < w_1 \,\},$$

with regions pairwise disjoint. This is struct-offset arithmetic, nothing more.

The topology compiles to a mask. Edges $E = \{(r_k \to s_k, w_k, \tau_k)\}$ lower to $M \in \mathbb{R}^{N \times N}_{\geq 0}$:

$$M_{ij} \;=\; \max_k \; w_k \, \mathbf{1}[i \in I_{r_k}] \, \mathbf{1}[j \in I_{s_k}] \, A_{\tau_k}\big(t(i), t(j)\big),$$

where A_τ is the temporal alignment predicate: for offsets $(t_{\text{src}}, t_{\text{dst}})$, positions align iff $\exists \text{ref} : t(i) = \text{ref} + t_{\text{src}},\ t(j) = \text{ref} + t_{\text{dst}}$ (an unset offset is unconstrained). When periods differ, alignment is evaluated in real time $\rho_r(t) = p_r t$, and gaps resolve by fill mode: `HOLD` snaps to the most recent past anchor, `DROP` yields no edge, `INTERPOLATE` distributes weight $\propto 1/d^n$ over the $n{+}1$ nearest anchors. Execution is per-edge cross-attention, $Y_{I_r} \mathrel{+}= w \,\mathrm{softmax}\big(Q_{I_r} K_{I_s}^\top / \sqrt{d}\big) V_{I_s}$ (with the edge's `fn` substituting the kernel). For a *binary* mask ($w \in \{0, 1\}$) this coincides with the monolithic form—adding $\log M$ to dense attention logits with $\log 0 = -\infty$; with unequal edge weights the monolithic form is a distinct (approximate) reweighting, since a joint softmax normalizes across edges. The load-bearing case for the causal claim below is the binary mask.

Loss participation is a weight vector. The declarations compile to $\omega_i = \sum_r \mathbf{1}[i \in I_r] \cdot \texttt{loss\_weight}_r \cdot \mathbf{1}[\texttt{is\_output}_r]$, and training minimizes the position-weighted denoising objective

$$\mathcal{L} \;=\; \mathbb{E}_{x_0, \varepsilon, \sigma}\left[\frac{\sum_i \omega_i \left\| \hat{\varepsilon}_\theta(x_\sigma)_i - \varepsilon_i \right\|^2}{\sum_i \omega_i} \right],$$

where noise is applied only to output positions at $t \geq t_{\text{current}}$; context positions are clamped to their encodings, i.e. conditioning is inpainting-style [13, 14].

Reachability is the causal statement. Write the information-flow graph G with an arc $s \to r$ for every connection (r queries s, so content moves $s \to r$). One denoiser pass with L attention blocks moves information along paths of length $\leq L$ in G; K sampling steps compose to a horizon of KL. The load-bearing direction is exact and holds by construction, not by regularization: if G has *no* directed path $a \to b$, then a cannot influence b's denoised value, an exact *marginal* independence—

Table 1: The type-system correspondence, item by item.

Type concept	Canvas equivalent	Implementation
Struct field (offset+size)	Region bounds	`region_indices()`
Field annotation	period, loss_weight, sem. type	`RegionSpec`
Pointer / reference	Connection	`CanvasTopology`
Function signature	Topology + fn	`attention_ops()`
Type-directed codegen	Loss mask	`loss_weight_mask()`
ABI compatibility	Schema compat.	`compatible_regions()`
Coercion cost	Transfer distance	`transfer_distance()`

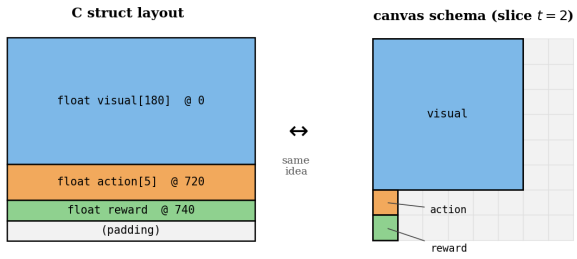

Figure 3: A C `struct` memory layout (left) and a canvas schema (right) are the same object: named fields at computed offsets, with a declared interpretation of every byte / position. The right panel is the t=2 slice of the layout in Fig. 2.

the empty-conditioning case of d-separation in a graphical model [16, 17]. The converse is bounded, not guaranteed: a path of length $\leq KL$ makes influence *possible* (the weights may still be zero), while a path longer than the horizon transmits nothing.

2.4 A type system, literally

The claim that this is a type system is not decorative (Table 1, Fig. 3). `region_indices()` computes memory offsets; the topology is a calling convention; a serialized `CanvasSchema` (layout + topology + metadata, human-readable JSON) is a complete type signature. Two models that share a schema can exchange latent state directly—no tokenization, no re-encoding—because the schema fixes what every position *means*.

Across differing schemas, a region can also declare its modality's meaning as a string plus a fixed embedding from a declared model [18], turning modality compatibility from a judgment call into a computable distance (`transfer_distance`): camera and depth should sit far nearer than camera and joint-angles. Whether that distance actually predicts transfer cost is an open calibration, contingent on the representation-stability hypothesis of §6; we make no empirical claim here.

2.5 Attention function types

Edges declare *how* information flows, not only *whether*. Seventeen registered function families span dot-product (`cross_`, `linear_`, `sigmoid_attention`), gating (`gated`), compression (`perceiver` [19], `pooling`), transfer (`copy`), state-space (`mamba` [20], `rwkv` [21]), convolution (`hyena` [22]), sparse (`local_`, `sparse_attention` [23]), and meta (`none`, `random_fixed`, `mixture`). Resolution order is `connection.fn` → `region.default_attn` → global `cross_attention`. Each choice encodes a theory of the edge—`pooling` for a 12-D proprioception summary, `perceiver` to bottleneck 864 visual tokens into a thought region, `copy` for direct latent relay between agents. The schema declares intent; the executor picks implementation.

3 Compilation: from entity schema to canvas

Hand-placing rectangles does not scale, so the compiler accepts typed entity declarations—nested Python dataclasses; a pydantic surface is the same move—and flattens the entity/relationship graph they denote into layout + topology. Every nested type automatically receives a **coarse-grained field**: a compressed representation that bottlenecks all cross-level attention.

```
@dataclass
class Vehicle:
    __coarse__ = Field(4, 4) # 4x4 summary
    camera: Field = Field(8, 8)
    plan: Field = Field(2, 4)
    action: Field = Field(1, 4, loss_weight=2.0)

@dataclass
class Fleet:
    dispatch: Field = Field(4, 4)
    vehicles: list = dc_field(default_factory=list)

bound = compile_schema(
    Fleet(vehicles=[Vehicle() for _ in range(50)]),
    d_model=256)
```

Fifty vehicles under dense cross-attention would cost $O(50^2 \times \text{fields}^2)$ connections; under coarse-graining, each interacts through its 4×4 summary—$O(50 \times 16)$. Deep nesting chains the bottlenecks: in a world-model schema, the attention path from `us.macro.gdp` to `cn.macro.inflation` is forced through `us.macro` (coarse) → `us` (coarse) → `regime` → `cn` (coarse) → `cn.macro` (coarse). Each hop compresses, so hierarchical abstraction is not an emergent hope but a topological consequence. In effect the declared schema is a probabilistic graphical model over latent factors [16, 17], compiled into the attention structure of the denoiser rather than into a message-passing runtime.

The **program layer** (v2) adds process semantics per region: a *family* (observation / state / memory / residual / action), a *carrier* (deterministic / diffusive / filter / memory / residual), a *clock* (periodic or event-triggered, with composable firing rules such as `Or(periodic(4), on("err.prediction", gt=0.5))`), and a *compile mode* (runtime / freeze / constant / export). Connection operators are auto-derived from family

pairs (observation→state = "observe"; state→action = "act"); edges can carry triggers (skipped when a predicate over region statistics is false) and learned sigmoid gates. `ProgramCompiler` materializes deploy-time semantics: frozen regions get `requires_grad=False`, constants become buffers, and export regions write state dicts to disk. The stack is thus **entity types → canvas schema → wired dispatch → compiled deployment**, each layer inspectable, serializable, and diffable.

4 Worked designs

The abstraction earns its keep on problems whose causal structure is known but whose dynamics are not.

Robot manipulation. Vision-heavy, low-latency actions. `visual` uses full `cross_attention` for spatial reasoning; `proprio` (12-D joints) uses `linear_attention`; `visual→action` asks which patches matter, while `proprio→action` is a `pooling` summary. This is the configuration behind the empirical results of §5, trained on BridgeData V2 robot video [24].

Multi-agent coordination. A vehicle fleet declares per-vehicle self-attention with cross-edges only between neighbors and a shared dispatch hub; an air-traffic conflict-routing schema declares 12 aircraft whose conflict geometry routes through a shared airspace field. Agents never read each other directly—coordination is forced through the shared region we declare between them. The topology *is* the coordination protocol, written as data.

Hospital ICU: causal structure as a type hierarchy. The deepest schema in the library (Fig. 1): six patients with organ-level physiology, four nurses with fatigue and workload dynamics, a bureaucratic state (insurance, staffing, bed pressure), and family units. One `compile_schema` call turns that nested instance into 199 regions and 1,077 connections on a 28×28 canvas—the engineer writes the schema; the compiler does all placement. Each organ system is a nested type with per-field temporal periods that mirror clinical reality (`heart_rate` at `period=1`, `creatinine` at `period=24`, `delirium_risk` at `period=12`, `loss_weight=3.0`). A side effect worth stating on its own: the latent tensor becomes *pointable*. One can circle a run of blocks in Fig. 1 and say "that is `nurses[1]`"—interpretability not as post-hoc attribution but as a legend for memory, known before training starts. Crucially, the sepsis deterioration pathway—`renal.creatinine_rise` → `cardiovascular.MAP_drop` → `neurological.altered_consciousness` → `deterioration_risk`—is *declared*. A flat model learns the correlation; *a canvas model is forced to route through the causal pathway, which is what makes the prediction interpretable and, by hypothesis, robust to distribution shift.* Every field has a physiological reason to exist; every edge has a known biological pathway. The type system is not decoration—it is the domain knowledge, made executable.

Cortical world model. Declaration taken to its limit: 23 brain regions from the Destrieux atlas [25] wired by 42 known cortical pathways (ventral visual stream, the auditory-to-language route, executive control, default mode), across five families—observation (V1, A1, somatosensory), state (association areas), memory (temporal pole), action (motor cortex), and residual (prediction error). The training target is next-timestep cortical activation as *estimated by* Meta's TRIBE v2 brain-encoding model [26] (not measured neural data): from 72 stimuli mapped to 135 regional features, the declared-topology model reaches $R^2 = 0.825$ on next-step prediction while using 19.6% of possible feature-level connections (3,579 of 18,225). A separate decoder—4-way category classification from a virtual 20-channel EEG montage sampled from the same TRIBE predictions at 10–20 electrode sites—reaches 68.8% (linear SVM 59.4%, chance 25%). The ablation is instructive: on a low-dimensional (23-scalar) version of the prediction task a small flat MLP wins, and at 135 features a fully dense model matches the cortical one—topology helps convergence and sparsity, not the ceiling. **Topology is a convergence prior, not a capacity advantage.**

The cortex as existence proof. The cortical model is not just another worked example; it is a proof that this fixed-macro / learned-micro decomposition already occurs in a working intelligence. The gross connectivity of the cortex is largely genetically specified—a wiring diagram—while synaptic weights are tuned by plasticity over a lifetime. That is exactly the split canvas engineering makes: the engineer fixes the topology, gradients learn the weights. The correspondence is not merely rhetorical. A cortical connectivity matrix and a canvas attention mask are the *same object*—a source-by-destination matrix, block-diagonal within a functional network, sparse and specific across networks, each entry typed by an operator (see the connectivity-matrix visualization in the repository, whose axes are identical to §2.3's M). Wiring the declared graph to match known neuroanatomy and recovering the TRIBE-estimated cortical dynamics at $R^2 = 0.825$—matching a fully dense model at 19.6% of the connections, and in our runs training several-fold faster because the dispatcher iterates only over the sparse connection set—is a falsifiable structural claim, and its honest reading is the convergence-prior story: the connectome accelerates development without raising the ceiling, just as declared topology accelerates training without adding capacity. One further correspondence is suggestive but we make no mechanistic claim: the cortex clamps sensory input and generates internally when thalamic gating opens, which has the same shape as clamping con-

text regions and denoising output regions—we note the resonance and explicitly do not assert that cortex implements reverse diffusion. The load-bearing point needs no such assertion: whatever the brain is doing, its *structure*—declared macro-topology hosting learned micro-dynamics—is now something an engineer can specify on an arbitrary typed process.

5 Empirical record

The looped-attention results in this section are established, with full protocols, hyperparameters, per-experiment statistical tests, and ablations, in a companion empirical study [11]; we summarize the load-bearing findings and their honest limits here. All results use a CogVideoX-2B backbone [6] on BridgeData V2 robot video [24]: 26 experiments, 236 training runs.

Looped attention. Orthogonal to the canvas, looped attention iterates frozen DiT blocks k times with zero-initialized learned iteration embeddings—so at initialization the model is *exactly* the pretrained backbone, with no distribution shift—in the spirit of universal and recurrent transformers [27, 28]. From a 12-condition grid over loops × freeze level (Table 2):

Table 2: Action loss (lower better) across the full 4×3 grid (loops × freeze level; 3 seeds each, 36 runs). Three loops wins at every freeze level; frozen 3-loop (350K trainable params) beats every unfrozen condition (11.7M). Column marginals (mean over loops): 0.110 / 0.120 / 0.108.

	Frozen (350K)	Half (3.7M)	Unfrozen (11.7M)
1 loop	0.121	0.115	0.108
2 loops	0.140	0.119	0.112
3 loops	**0.073**	0.107	0.088
4 loops	0.104	0.137	0.124

Recurrence beats depth in parameters: a 3-loop model reaches the loss of a depth-matched baseline carrying **1.73**× its trainable parameters (the estimator and paired test are given in the companion study [11]; $p < 0.001$). The freeze main effect is not significant (loop-averaged marginals 0.110 / 0.120 / 0.108 for frozen / half / unfrozen; $p = 0.72$); freezing affects video-generation quality instead (an 8–9× gap on diffusion loss). Loop representations converge toward fixed points (cosine similarity to loop 1: $0.926 \rightarrow 0.996$; token velocities decay exponentially, action tokens fastest).

Reading the data at the right altitude. These are 2B-parameter, single-domain, small-data experiments, and we deliberately avoid narratives the data cannot yet carry. Three findings calibrate what the architecture does at this scale, and each one *informs the design* rather than undermining it:

1. *Where looping's benefit comes from.* At this scale the measurable benefit of looped attention is weight-sharing regularization—fixed-point convergence, lower variance, the $1.73\times$ efficiency above—rather than detectable iterative reasoning (three independent nulls). Whether reasoning-like refinement emerges with larger canvases, longer horizons, and more compute is exactly the question the architecture is built to test; we decline to claim it early.
2. *Structure must be declared, not hoped for.* Merely co-locating modalities on a flat, densely-attending canvas did not improve joint prediction—in our runs it degraded it by 19% ($p < 0.0001$). We read this as evidence that binding, like the rest of the structure, has to be declared rather than left to emerge: the capability came from the typed multi-encoder/decoder interface, and co-location alone did not add to it. The natural follow-up—the same canvas with and without declared connectivity—is the next experiment.
3. *Canvas design is forgiving.* Loss is nearly insensitive to token allocation ($\alpha = 0.011$: doubling a region's tokens moves loss 0.8%). Geometry can therefore serve legibility and engineering convenience without materially taxing accuracy—which is convenient for an interface whose premise is that a human draws the layout.

Reproducibility and scope. The library, the runnable worked designs, and the schemas in this paper are released open-source (§Artifacts); the looped-attention experiments—seeds, hyperparameters, per-condition tests, and the full experiment archive—accompany the companion study [11]. Two scope notes. First, the worked designs (the ICU deterioration predictor, air-traffic conflict routing) are *architectural demonstrations* of how causal structure is declared, not clinically or operationally validated systems, and should not be read as such. Second, the cortical result predicts activations produced by a brain-encoding model [26], not measured neural recordings; extending it to real fMRI/EEG is future work (§6).

6 Open problems

Representation stability (the linchpin). Everything interoperable—schema-mediated latent exchange, transfer-distance calibration, plug-and-play modalities, swap-the-policy-keep-the-perception composition—rests on a platonic-representation-style claim [29]: that identical declared structure induces predictably aligned latent geometry across seeds, datasets, and backbones. Plausible, unproven. The test battery is specified: seed-stability CKA across corresponding regions [30]; topology→specialization ablations on fixed data; cross-model grafting cost; and regression of measured transfer cost against semantic-embedding distance.

Binding. The flat-canvas co-residence penalty (§5) says implicit binding is not free. Whether *declared topology*—the same canvas with authored connectivity—recovers or exceeds independent-model performance is the decisive ablation, and one on which the framework makes a concrete, testable prediction.

Scale. All evidence is at 2B parameters and robot-video scale. Whether declared macrostructure keeps paying as data grows, or is eventually washed out [7, 31], is open. The bet is asymmetric: schemas cost nothing to author, compile onto pretrained backbones, and can be relaxed edge-by-edge (`fn="none"` → dense) if they fail to bind.

Learned topology. Today the graph is authored. The natural continuation—propose-edges / prune-edges under sparsity pressure, with the declared schema as prior rather than hard constraint—would close the loop back to Chollet's synthesis route, with the canvas as the substrate both routes share.

7 Related work

Diffusion policy [15] establishes observation-conditioned action diffusion—the two-node canvas. Unified multimodal DiTs [5, 6] share one latent sequence but leave structure implicit, to be discovered from surface gradients: the exact move canvas engineering replaces with declaration. Structured-attention mechanisms—sparse patterns [23], Perceiver bottlenecks [19], state-space and long-convolution mixers [20–22]—supply the executors; the contribution here is not a mechanism but an authorable, compilable, serializable *schema layer* over them. Probabilistic graphical and structural-causal models [16, 17] are the ancestors: a canvas topology is a causal graph whose message passing is implemented by a pretrained denoiser's attention rather than by belief propagation. World models [8–10] motivate the target application. Looped and universal transformers [27, 28] study recurrence as adaptive compute; at the scales we have tested, its practical value on frozen video backbones is weight-sharing regularization.

8 Conclusion

The neurosymbolic program stalls in production because it usually demands a new runtime at the neural/symbolic boundary. Canvas engineering dissolves the boundary instead: the symbolic artifact is a schema—a type system for latent computation—and its execution is masked attention inside a stock diffusion transformer. You declare the causal macrostructure of thought; SGD fills in the rest. The empirical record is early and we present it at face value: 1.73× parameter efficiency, a 350K-parameter configuration that beats 11.7M, and calibration results that tell us where structure must be declared rather than hoped for. The decisive experiments—declared topology versus flat co-residence, representation stability across seeds and backbones, behavior at scale—are specified and awaiting compute, and we would rather run them than speculate ahead of them. What stands regardless is the abstraction. Once latent structure is *declarable*, it is versionable, diffable, compilable, and shareable—and structure you can ship is the only structure that will ever out-compete paying for more data.

Artifacts. `pip install canvas-engineering` (Apache 2.0), published under CommandAGI. Code: `github.com/commandAGI/canvas-engineering`; documentation, runnable examples, and design notes: `commandagi.com/research/canvas-engineering`. Full experiment archive and the companion empirical paper ("Looped Attention in Video Diffusion Transformers: 26 Experiments on What Works, What Doesn't, and Why") at `github.com/JacobFV/recursive-omnimodal-video-action-model`.

References

[1] François Chollet. "eventually, much of ai will converge towards intuition-guided symbolic world modeling, i.e. deep learning-guided program synthesis". Post on X, https://x.com/fchollet/status/2072779641639875048, July 2026.

[2] François Chollet. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

[3] Gary Marcus. The next decade in ai: Four steps towards robust artificial intelligence. *arXiv preprint arXiv:2002.06177*, 2020.

[4] Artur d'Avila Garcez and Luís C. Lamb. Neurosymbolic ai: The 3rd wave. *Artificial Intelligence Review*, 56:12387–12406, 2023.

[5] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[6] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[7] Richard S. Sutton. The bitter lesson. *Incomplete Ideas (blog)*, 2019.

[8] David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[9] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

[10] Yann LeCun. A path towards autonomous machine intelligence. *Open Review*, 2022.

[11] Jacob Valdez. Looped attention in video diffusion transformers: 26 experiments on what works, what doesn't, and why. Companion technical report (developed with AI assistance), https://github.com/JacobFV/recursive-omnimodal-video-action-model, 2026.

[12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[13] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[14] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[15] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[16] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

[17] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2nd edition, 2009.

[18] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, 2021.

[19] Andrew Jaegle, Felix Gimeno, Andy Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*, 2021.

[20] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[21] Bo Peng, Eric Alcaide, Quentin Anthony, et al. Rwkv: Reinventing rnns for the transformer era. *Findings of EMNLP*, 2023.

[22] Michael Poli, Stefano Massaroli, Eric Nguyen, Daniel Y. Fu, Tri Dao, Stephen Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning (ICML)*, 2023.

[23] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[24] Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. In *Conference on Robot Learning (CoRL)*, 2023.

[25] Christophe Destrieux, Bruce Fischl, Anders Dale, and Eric Halgren. Automatic parcellation of human cortical gyri and sulci using standard anatomical nomenclature. *NeuroImage*, 53(1):1–15, 2010.

[26] Meta AI Research. A foundation model of vision, audition, and language for in-silico neuroscience. Meta AI Research; model `facebook/tribev2`, 2025. https://ai.meta.com/research/publications/a-foundation-model-of-vision-audition-and-language-for-in-silico-neuroscience/.

[27] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. In *International Conference on Learning Representations (ICLR)*, 2019.

[28] Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[29] Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. In *International Conference on Machine Learning (ICML)*, 2024.

[30] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning (ICML)*, 2019.

[31] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.