CanViT: Toward Active-Vision Foundation Models

Yohaï-Eliel Berreby[1,2,4] **Sabrina Du**[1,2,4] **Audrey Durand**[2,3] **B. Suresh Krishna**[1]

[1]McGill University [2]Mila – Quebec AI Institute [3]Université Laval [4]Lamarck Labs

me@yberreby.com sabrina.du@mail.mcgill.ca

audrey.durand@ift.ulaval.ca suresh.krishna@mcgill.ca

Homepage: https://m2b3.github.io/CanViT/

Abstract

Active computer vision promises efficient, biologically plausible perception through sequential, localized glimpses, but lacks scalable general-purpose architectures and pretraining pipelines, leaving Active-Vision Foundation Models (AVFMs) underexplored. We introduce the Canvas Vision Transformer (CanViT), the first task- and policy-agnostic AVFM. CanViT uses scene-relative RoPE to bind a retinotopic Vision Transformer backbone and a spatiotopic scene-wide latent workspace, the *canvas*. Efficient interaction with this high-capacity working memory is supported by Canvas Attention, a novel asymmetric cross-attention mechanism. We decouple *thinking* (backbone-level) and *memory* (canvas-level), eliminating canvas-side self-attention and fully-connected layers to achieve fast sequential inference and scalability to high output resolutions. We propose a label-free active vision pretraining scheme, policy-agnostic passive-to-active dense latent distillation: reconstructing scene-wide DINOv3 embeddings from sequences of low-resolution glimpses with randomized locations, zoom levels, and lengths. We pretrain CanViT-B from a random initialization on 13.2 million ImageNet-21k scenes and 1 billion random glimpses, in 166 hours on a single H100. On ADE20K segmentation, a frozen CanViT-B achieves 38.5% mIoU in a single low-resolution glimpse, outperforming the best active model's 27.6% with 20× fewer inference FLOPs as well as its FLOP- or input-matched DINOv3 teacher. Given additional glimpses, CanViT-B reaches 45.9% ADE20K mIoU. On ImageNet-1k classification, CanViT-B also sets a new active-vision state of the art, with 84.5% top-1 accuracy after fine-tuning. CanViT generalizes to longer rollouts, larger scenes, and new policies. Our work narrows the wide gap between passive and active computer vision, demonstrating the potential of task- and policy-agnostic AVFM pretraining.

1 Introduction

Deep Artificial Neural Networks (ANNs) have achieved outstanding performance in a variety of computer vision tasks, and proven valuable to life sciences research as computational models of biological visual processing [1–7]. This practical and scientific success has largely hinged on vision encoders which process individual frames independently and passively, without the ability to reuse previous computation to guide further processing in a recurrent, active, human-like manner.

Unlike most ANNs, humans sample their visual environment by actively and frequently orienting their sensory apparatus towards regions of interest (ROIs), through gaze shifts. This process is inherently sequential, and involves strategic planning [8,9], integration of evidence across time in visual working memory [10,11], and top-down recurrent feedback—rich conditioning of early visual pathways by signals from higher brain areas, allowing each fixation's visual information to be processed in a contextually informed manner, based on what was seen earlier [12–14], rather than *tabula rasa*.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

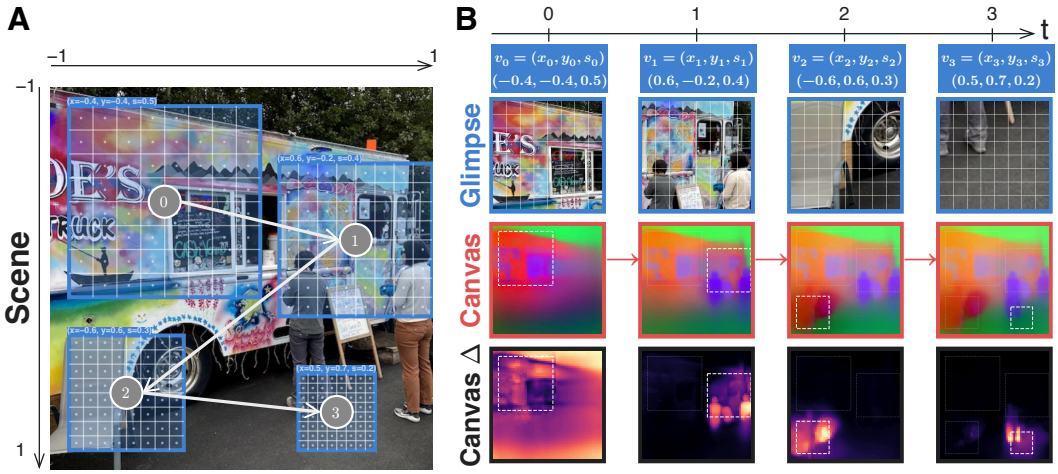

Figure 1: **A CanViT rollout.** We consider a high-resolution scene (**A**). At each timestep t, CanViT ingests a 128^2 px *glimpse* (**B**, 1st row), a crop extracted at a *viewpoint* with center $(x_t, y_t) \in [-1, +1]^2$ and scale (zoom level) $s_t \in (0, 1]$. This updates a scene-wide latent representation, the *canvas*, with which CanViT integrates broad context and fine detail from variable-scale glimpses, extrapolates to unobserved regions, and conditions visual processing on a working understanding of the scene. We visualize the canvas via Principal Component Analysis across tokens (**B**, 2nd row), and canvas updates (Δ) as cosine dissimilarity heatmaps across consecutive timesteps (**B**, 3rd row).

Drawing inspiration from human vision, various Active Computer Vision (ACV) models, which process visual scenes through sequential, localized glimpses, have been proposed [15–32]. These models often seek biological relevance and computational efficiency. Yet, despite their theoretical advantages, they have struggled to match the accuracy, efficiency, flexibility and representational richness of their passive counterparts. This disconnect is particularly striking on dense prediction, which is unsupported by most existing active vision models; the few exceptions lag dramatically behind passive models [33,34] on standard benchmarks like ADE20K segmentation [35] when evaluated on it [24,25].

An active vision model can be considered along three axes: making sense of what it sees at a given moment (instantaneous vision); updating a persistent, evolving understanding of the scene, which may in turn inform further processing (memory); and deciding where and at what zoom level to look next (action selection). The first two axes define an observer's ability to understand, persist, and recall visual inputs, while the last defines its input-selection strategy, or sensory policy. They do not play comparable roles: given perfect instantaneous vision and memory, naive and strategic policies should reach the same accuracy after enough glimpses of a static scene, but no policy can make up for a poor observer. Yet, prior ACV work has often focused on action selection [15–27,29]. Concurrently, passive-vision foundation models like DINOv3 [36] excel at general-purpose instantaneous vision but lack essential active-vision components, as they have no concept of localized glimpses or memory.

Focusing on the intersection of instantaneous vision and memory, we sought to build a task- and policy-agnostic Active-Vision Foundation Model (AVFM): a vision model capable of understanding the spatial and semantic structure of scenes across arbitrary sequences of glimpses, with rich representations that transfer across tasks and viewing policies. This policy-agnostic approach disentangles "how to see in an active-vision setting" from "where to look," thus freeing active-vision pretraining from the complexity of Reinforcement Learning (RL). Policy agnosticism may also be desirable for real-world deployment, where physical limitations (e.g., a motorized camera's range of motion and optical zoom levels) may preclude the use of a general-purpose policy.

Contributions. We introduce the Canvas Vision Transformer (CanViT, Figure 1), a recurrent architecture built around a latent scene-wide representation called the *canvas* (§4) and pretrained with a novel passive-to-active distillation scheme (§5). We evaluate CanViT-B on ADE20K semantic segmentation and ImageNet-1k [37] classification (§6). CanViT-B sets a new state of the art among active vision models on both ADE20K and ImageNet-1k while transferring across policies, temporal horizons, and canvas resolutions without retraining.

2 Related work

Deep active vision models typically process sequences of *glimpses*—fixed- or variable-scale crops extracted from a larger image or video. This line of research traces back to the Recurrent Attention Model[15], and remained largely confined to simple tasks such as digit recognition[16,18] until 2019, when Saccader[19] achieved 75% ImageNet-1k[37] top-1 accuracy by introducing an intermediate pretraining step to stabilize learning. GFNet[20] and AdaptiveNN[26] later showed that active vision could deliver computational efficiency gains on real-world tasks, although both remained structurally limited to classification tasks and fixed zoom levels.

Dense prediction in active vision has remained underexplored, as most ACV architectures cannot produce the scene-wide, spatially dense outputs required in tasks like semantic segmentation or depth estimation. Among the few exceptions, AME[24] and AdaGlimpse[25] achieve dense prediction through post-hoc expansion of encoder outputs: at each timestep, a MAE-style[38] Transformer decoder receives all encoded glimpse tokens with a full grid of learnable mask tokens and performs self-attention over the entire grid to produce scene-wide predictions. This becomes intractable at high scene resolutions, where active vision is most appealing. Despite being the state of the art on active ADE20K segmentation, these models respectively achieve only 27.6% and 25.7% mIoU.

Dense latent distillation from self-supervised ViTs allows the visuospatial intelligence acquired through extensive pretraining to be quickly transferred into randomly-initialized models. The largest DINOv2[39]/v3[36] models were distilled into smaller ViTs using the same dataset and objective as during pretraining. Proteus[40] distilled DINOv2-{g,L}/14[39] into a smaller model using 100× less data than during pretraining and a simple loss combining CLS token matching with dense feature matching. Our distillation scheme follows a similar philosophy to Proteus, but transfers across problem settings rather than across model sizes: we use a passive teacher's visual world knowledge to teach an active student how to see from arbitrary glimpse sequences.

Cross-attention for dimensionality/computation decoupling was pioneered by the Set Transformer[41], popularized by Perceiver models[42,43], then generalized to iterative bidirectional routing by Recurrent Interface Networks (RINs)[44]. Like RINs, CanViT alternates read and write cross-attention across depth and time. Unlike Perceiver and RINs, CanViT ingests external input on its few-token side (glimpse), with latents on the many-token side (canvas). Moreover, CanViT's many-token side forgoes not only self-attention, but also all fully-connected layers: canvas tokens never go through Multi-Layer Perceptrons (MLPs), QKVO projections in cross-attention, or GRU[45]/LSTM[46] recurrent gates. Instead, canvas tokens evolve solely via our Canvas Attention mechanism.

Latent-space recurrent reasoning iterates over a fixed external input in a weight-tied fashion, decoupling representational capacity from computational depth to enable improved algorithmic reasoning[47–51] and flexible test-time compute allocation[52,53]. This paradigm has recently seen renewed interest, both in the Large Language Model (LLM) space[54,55] and for its ability to produce small yet highly capable models[50,51]. Leveraging top-down recurrent feedback allows previous computation to be reused, without losing most of it to ephemeral step-wise outputs. Active vision offers an opportunity to generalize this framework by allowing each processing step to benefit from a new perspective on the scene instead of operating on a constant input. CanViT achieves latent-space recurrent reasoning with a semantically rich (rather than pixel-like) workspace, the *canvas*, which provides top-down recurrent feedback for early layers to build upon.

3 Preliminaries

Definitions: scenes, glimpses, viewpoints. For a timestep $t \in \mathbb{N}$, we consider a bounded 2D *scene*, which can be formally represented as a function $\psi_t : [-1, +1]^2 \to \mathbb{R}^3$ mapping continuous-valued (x, y) coordinates to RGB values. In our experiments, we consider static scenes, with $\psi_t = \psi_0$ for all timesteps. We define a *glimpse* as a fixed-resolution crop, extracted at a *viewpoint* $\mathbf{v}_t = (x_t, y_t, s_t)$, where $(x_t, y_t) \in [-1, +1]^2$ is the crop center in scene coordinates and $s_t \in (0, 1]$ is the crop's *scale*, or half-side-length. That is, the crop spans the scene coordinates $[x_t - s_t, x_t + s_t] \times [y_t - s_t, y_t + s_t]$, covering a fraction s_t^2 of the scene's surface area. Regardless of its scale s_t, the crop is resized to a fixed resolution of $H_g \times W_g$ pixels, which determines its information capacity; under that constraint, s_t smoothly controls the tradeoff between spatial coverage and perception of detail.

General-purpose, spatially grounded active vision. We wish for our model to build up a general-purpose understanding of the *scene* through a sequence of *glimpses* that it perceives, allowing each additional processing step to build upon and refine this understanding. This evolving latent representation should be readily decodable into predictions at every timestep, whether for non-spatial, global prediction tasks like object classification or spatially-grounded, dense tasks like semantic segmentation or depth estimation. Dense predictions require explicit architectural handling, as the active vision setting breaks the direct, connectivity-based mapping between input and output feature maps that passive Convolutional Neural Networks (CNNs) and ViTs commonly exploit: glimpses need not align with the scene from which they are extracted.

To address this problem, we introduce the **Canvas Vision Transformer**, or **CanViT** (§4), a recurrent vision architecture built around a scene-wide latent representation called the **canvas**.

4 The Canvas Vision Transformer (CanViT)

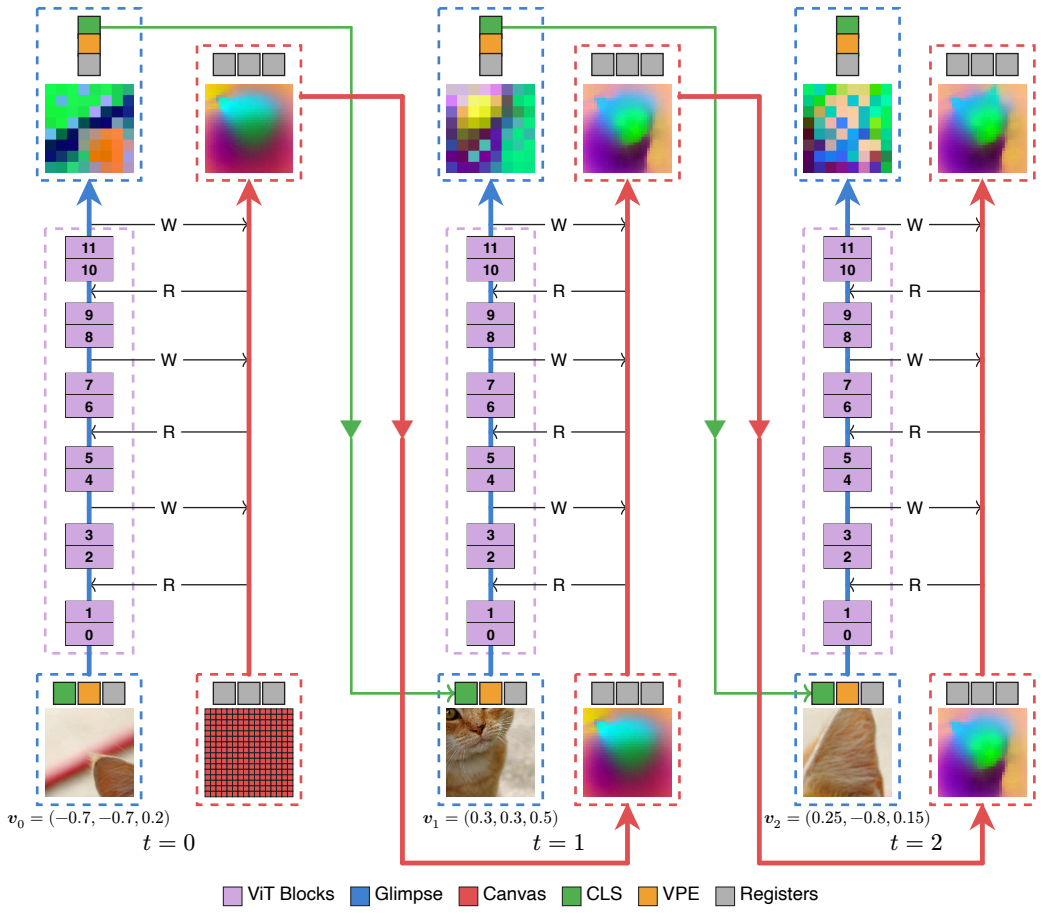

Figure 2: **CanViT architecture diagram.** We adopt a dual-stream structure, equipping a **ViT backbone** (purple, left-hand columns), which processes localized **glimpses** (blue), with a **canvas** (red, right-hand columns), a fine-grained scene-wide spatio-semantic memory. At each timestep t, a glimpse is extracted from a viewpoint $\mathbf{v}_t = (x_t, y_t, s_t)$, patchified, and processed through the backbone, alongside a recurrent CLS token (green) and a Viewpoint Encoding (VPE) token (orange). The canvas regularly interacts with the glimpse stream via **Canvas Attention** (Figure 3), alternating between Read (R) and Write (W) operations to, respectively, condition the backbone's processing on the canvas and populate the canvas. Both streams are equipped with register tokens (gray)[56].

The CanViT architecture (Figure 2) formulates active-vision processing as the interaction between a high-capacity memory stream (the *canvas*) and a ViT[57] backbone's compact glimpse processing

stream. These streams interact bidirectionally through Canvas Attention (Figure 3), a mechanism that allows the backbone to efficiently pull information from the canvas and send updates to it.

The **glimpse stream**, made up of D_{bb}-dimensional glimpse tokens, is largely ephemeral. Each glimpse is extracted from the scene at a given viewpoint, split into 16^2 px patches, and fed to the backbone alongside ephemeral register tokens[56], a recurrent CLS token, and a viewpoint encoding (VPE) token derived from the glimpse's position and zoom level. Since spatial alignment between consecutive patch grids cannot be assumed, as a glimpse may be taken from any position and at any zoom level, patch tokens cannot be directly forwarded across time without destroying their grid structure. Instead, CanViT persists relevant information from glimpses via the canvas stream.

The **canvas stream**, made up of D_{can}-dimensional canvas tokens, is fully persistent. The canvas acts as a scene-wide spatio-semantic memory, which can function as a cognitive map[58] of the scene. It comprises a few *canvas registers*, which act as a non-spatial memory, and a large $H \times W$ spatial grid of *canvas patches* tiling the $[-1, +1]^2$ scene coordinate space. At the start of each rollout, this grid is broadcasted to the desired size from a single learnable initial patch, enabling the canvas resolution to be set at inference time. Each canvas patch maps onto a fixed scene region, regardless of the current viewpoint, thus allowing direct token-wise decoding into dense predictions and unbroken gradient flow across time. After initialization, the canvas is read from and written to by consecutive Canvas Attention layers, whose outputs are residuals injected into each stream. No MLPs or self-attention layers are applied to canvas tokens, which evolve solely by interacting with the backbone's glimpse tokens via Canvas Attention Write operations. This restriction is key to CanViT's efficiency, as the canvas stream is designed to accommodate a much larger number of tokens than the glimpse stream.

Scene-Relative Rotary Position Embeddings (SR-RoPE). We compute 2D RoPE[59,60] from the centers of glimpse patches and canvas patches in the scene's $[-1, +1]^2$ coordinate system, both in the ViT backbone's self-attention and in Canvas Attention layers. The positions of glimpse patch centers depend on the current viewpoint (x_t, y_t, s_t), with their relative distances implicitly communicating the viewing scale s_t (zoom): a zoomed-in glimpse's patches are tightly clustered in a small scene region, while a zoomed-out glimpse's tokens span a wider region (Figure 1). The positions of canvas patches are constant for any given canvas grid size, since they uniformly tile the scene. Consistent use of scene-relative coordinates binds the retinotopic glimpse stream and the spatiotopic canvas stream with a shared reference frame, while RoPE enables generalization across patch grid sizes.

Canvas Attention (Figure 3**A**; pseudocode in Appendix A), a mechanism based on cross-attention, enables efficient interaction between CanViT's relatively small set of glimpse tokens and its much larger set of canvas tokens, allowing CanViT to support fine-grained scene/canvas grids without incurring the quadratic cost of self-attention (Figure 3**B**). We alternate between *Read* and *Write* operations along depth (and, implicitly, time) using a *stride* of 2 ViT blocks in CanViT-B. In a *Read*, glimpse tokens query the canvas; in a *Write*, canvas tokens query the glimpse. In both cases, the cross-attention output is added back to the querying stream via residual addition. SR-RoPE makes this process spatially aware, allowing Canvas Attention to bind the two streams. Unlike standard cross-attention implementations (e.g., multi-head attention in PyTorch[61] or Flax NNX[62,63]), Canvas Attention is *asymmetric*: it restricts Query, Key, Value and Output (QKVO) projections to one token set (glimpse-side tokens), applying only LayerNorm, RoPE (for Queries/Keys) and element-wise residual addition to the other one (canvas-side tokens).

Asymmetric projections. The glimpse token count N_g must be kept low due to the quadratic cost of the ViT backbone's self-attention, and it is desirable for the canvas to have high information capacity both *spatially*, by tiling the scene in a fine-grained manner with a large number $H \times W$ of canvas patch tokens (making up the bulk of the N_{can} canvas tokens), and *semantically* at any given position within the scene, with a large canvas embedding dimension D_{can}. This makes the asymmetric design of Canvas Attention highly advantageous, as the FLOP footprint of a single canvas-side projection relative to the Scaled Dot Product Attention (SDPA) call that it accompanies would be:

$$\frac{\text{projection FLOPs}}{\text{SDPA FLOPs}} = \frac{2N_{\text{can}}D_{\text{can}}^2}{4N_g N_{\text{can}} D_{\text{can}}} = \frac{D_{\text{can}}}{2N_g}, \tag{1}$$

which is a $7.2\times$ ratio with $D_{\text{can}} = 1024$ and $N_g = 71$ (64 glimpse patches, 5 registers, and CLS + VPE tokens). The FLOP savings from asymmetric cross-attention projections are magnified when using fewer glimpse patches and more canvas patches (Figure 3**C**). In CanViT-B, with a 64×64 canvas patch grid, adding canvas-side QKVO projections would increase the cost of each Canvas Attention Read/Write pair from 2.8 to 37.3 GFLOPs.

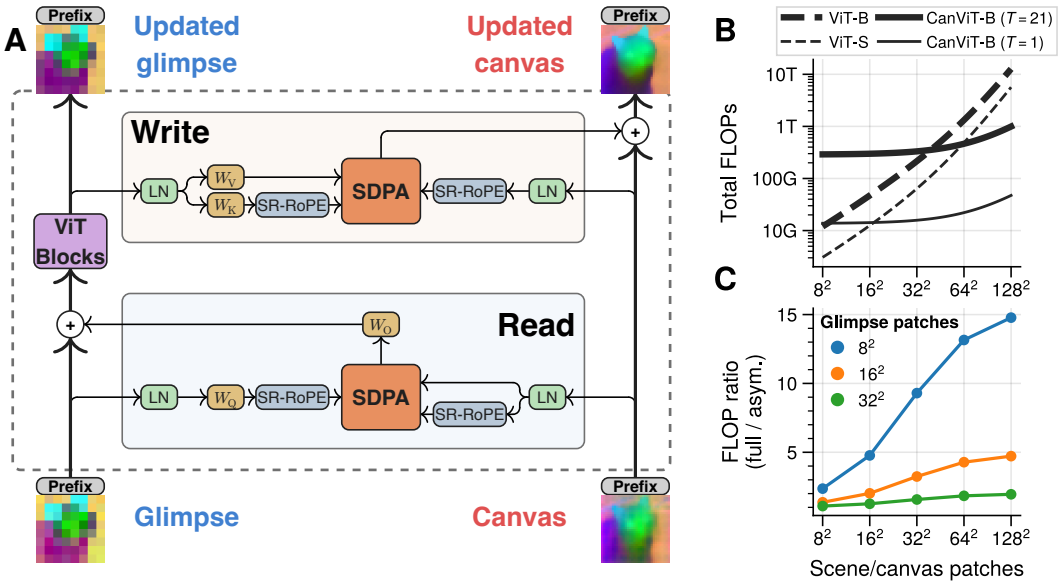

Figure 3: **A.** A Canvas Attention Read-Write pair. **B.** Total inference FLOPs vs. output patch grid. **C.** Relative cost of canvas-side QKVO projections, per Canvas Attention Read/Write pair.

Viewpoint Encoding (VPE) token. We supplement SR-RoPE, which distributes the encoding of viewpoint position and scale over glimpse patches, with a dedicated viewpoint encoding (VPE) token that concentrates the viewpoint triplet (x_t, y_t, s_t) into a single glimpse-side token. The VPE token is a non-essential architectural affordance, meant to facilitate future end-to-end policy learning by allowing the next viewpoint to be decoded from a rich transformation of the current viewpoint. We provide additional details on VPE in Appendix B and ablate it in Tables 13 and 15 and Appendix H.2.

5 Policy-agnostic passive-to-active dense latent distillation

An active vision model should (1) make sense of what it sees, (2) integrate observations into a persistent scene representation, and (3) decide where to look next. Here, we ask: how can we teach CanViT (1) and (2) in a task-agnostic, spatially aware, label-free manner, while remaining robust to the choice of policy (3)? We achieve this via *policy-agnostic passive-to-active dense latent distillation*, which combines highly informative reconstruction targets with rollout randomization.

5.1 Passive-to-active dense distillation

A natural pretext task for active-vision pretraining is *scene reconstruction*: training the model to produce a best-guess approximation of the overall scene from a sequence of glimpses. This incentivizes understanding of the spatial and semantic structure of scenes, in order to faithfully extrapolate to unseen regions and details. Since we wish for CanViT to iteratively build a semantically rich "mental image," we formulate this reconstruction objective in DINOv3[36] latent space, rather than in pixel space like passive Masked Autoencoders (MAEs)[38] and some active vision models[25].

Teacher. We build upon DINOv3[36], a self-supervised vision model family whose dense representations deliver outstanding frozen-feature transfer to classification, segmentation, depth estimation, and other downstream tasks. We use DINOv3 ViT-B as a *high-resolution scene-wide spatio-semantic teacher*, which produces patch tokens (dense features) and a global CLS token. These highly informative reference embeddings represent an idealized scene understanding for CanViT to match using sequences of cheap, partial glimpses. We apply per-position z-score standardization to reconstruction targets. High-resolution teacher inference requires more computation than a CanViT forward pass; however, since the teacher is frozen, we precompute reference features once, storing them for subsequent use across epochs and hyperparameter sweeps. This is an optional tradeoff between storage and compute cost; we estimate that computing teacher features online would increase total training compute by approximately 41%, while reducing storage from 19 TiB to zero.

Decoding. At each timestep t, CanViT produces updated canvas tokens, including canvas patches $\mathbf{C}_t \in \mathbb{R}^{H \times W \times D_{\text{can}}}$ and canvas registers, and an updated CLS token $\mathbf{h}_t \in \mathbb{R}^{D_{\text{bb}}}$. We apply layer normalization[64], then decode into DINOv3-space reconstructions via token-wise linear projections:

$$\hat{\mathbf{Z}}_t = W_C \cdot \text{LayerNorm}(\mathbf{C}_t), \quad \hat{\mathbf{z}}_t = W_h \cdot \text{LayerNorm}(\mathbf{h}_t) \tag{2}$$

Loss. Our loss combines patch-level and CLS-level reconstruction, averaged across space, teacher embedding dimensions (D), and time:

$$\mathcal{L} = \frac{1}{T} \sum_{t=0}^{T-1} \left[\frac{1}{HWD} \|\hat{\mathbf{Z}}_t - \mathbf{Z}^*\|_F^2 + \frac{1}{D} \|\hat{\mathbf{z}}_t - \mathbf{z}^*\|^2 \right] \tag{3}$$

5.2 Policy agnosticism

Dual rollouts. For each scene, we run two independent rollouts from a freshly-initialized canvas, averaging their losses. These rollouts and their viewpoint sampling policies are referred to as R-IID and F-IID, which only differ by their behavior at $t = 0$. R-IID (Random-then-IID) rollouts sample random viewpoints at all timesteps, including $t = 0$, ensuring robustness to arbitrary rollout starts. F-IID (Full-then-IID) rollouts always start with the full-scene zoomed-out viewpoint $(x_0, y_0, s_0) = (0, 0, 1)$, providing a low-resolution but high-spatial-coverage view of each scene at least once over the course of pretraining. Training with 1 F-IID + 1 R-IID rather than 2 R-IID rollouts accelerates convergence, even when evaluating on held-out data with an R-IID policy (Table 13 **h**).

Sampling of viewpoint center and scale. For all non-initial timesteps, as well as the $t = 0$ timestep of R-IID rollouts, we sample $A \sim \mathcal{U}([0, A_{\max}])$ and set $s = 1 - \sqrt{A}$. For a glimpse of scale s, valid centers form a box of half-side-length $\sqrt{A}$, within which we draw (x, y) uniformly. The resulting marginal scale density is $p(s) \propto (1 - s)$, favoring small, localized glimpses. We set $A_{\max} = (1 - s_{\min})^2$ with a minimum glimpse scale of $s_{\min} = 0.05$, or 0.25% of the scene's area.

Rollout length randomization. Our loss provides *temporally* dense supervision, with per-timestep credit assignment. This allows us to train CanViT with truncated Backpropagation Through Time (BPTT) over chunks of only $K = 2$ glimpses. At each chunk boundary, we stop the rollout with probability $p_{\text{stop}} = 0.5$, resulting in a geometric distribution of chunk counts with an *average* sequence length of $T = K/p_{\text{stop}} = 4$ glimpses while occasionally exposing the model to longer sequences. This scheme ensures sequence length robustness on a constant train-time VRAM footprint, with a modest train-time compute overhead due to the low average sequence length.

6 Experiments

To assess generalization across tasks, policies, temporal horizons, and canvas resolutions, we pretrain a general-purpose CanViT-B checkpoint and evaluate it with linear decoding on ADE20K[35] semantic segmentation and ImageNet-1k (IN1k) classification[37]. We use frozen weights to assess feature decodability on ADE20K and IN1k, and full fine-tuning to measure peak IN1k performance; we do not fine-tune on ADE20K due to its low scene count. We use 128^2 px glimpses consistently. We provide additional details on pretraining (Appendix C) and evaluation (Appendix D), as well as a detailed ablation study (Appendix E), inference latency measurements (Appendix F) and a visualization of canvas updates within and across glimpses (Appendix G). We also report additional experiments in Appendix H, covering the impact of pretraining on ImageNet-1k instead of ImageNet-21k, the relevance of the VPE token at long horizons and high canvas resolutions, ImageNet-1k accuracy against inference cost compared with AdaptiveNN and GFNet, an architecture control against AdaGlimpse with the same pretraining data and objective, and COCO[65] object detection.

Task-agnostic pretraining. Following the protocol described in §5, we pretrain CanViT-B from a random initialization in just 166 hours on a single H100, sampling approximately 1 billion glimpses from 13.2 million 512^2 px ImageNet-21k[37,66] scenes. We use an average sequence length of $T = 4$ (via $p_{\text{stop}} = 0.5$ and $K = 2$) and a canvas resolution of $32^2 = 1024$ patches during pretraining.

Task-specific probing. For ADE20K, we train linear probes to predict segmentation masks from canvas patches. We train a separate probe for each considered canvas resolution. For ImageNet-1k, we use a linear probe to predict logits from CanViT's recurrent CLS token. When fine-tuning, we start from our frozen-weights probes and unfreeze CanViT's own weights (i.e., LP-FT[67]).

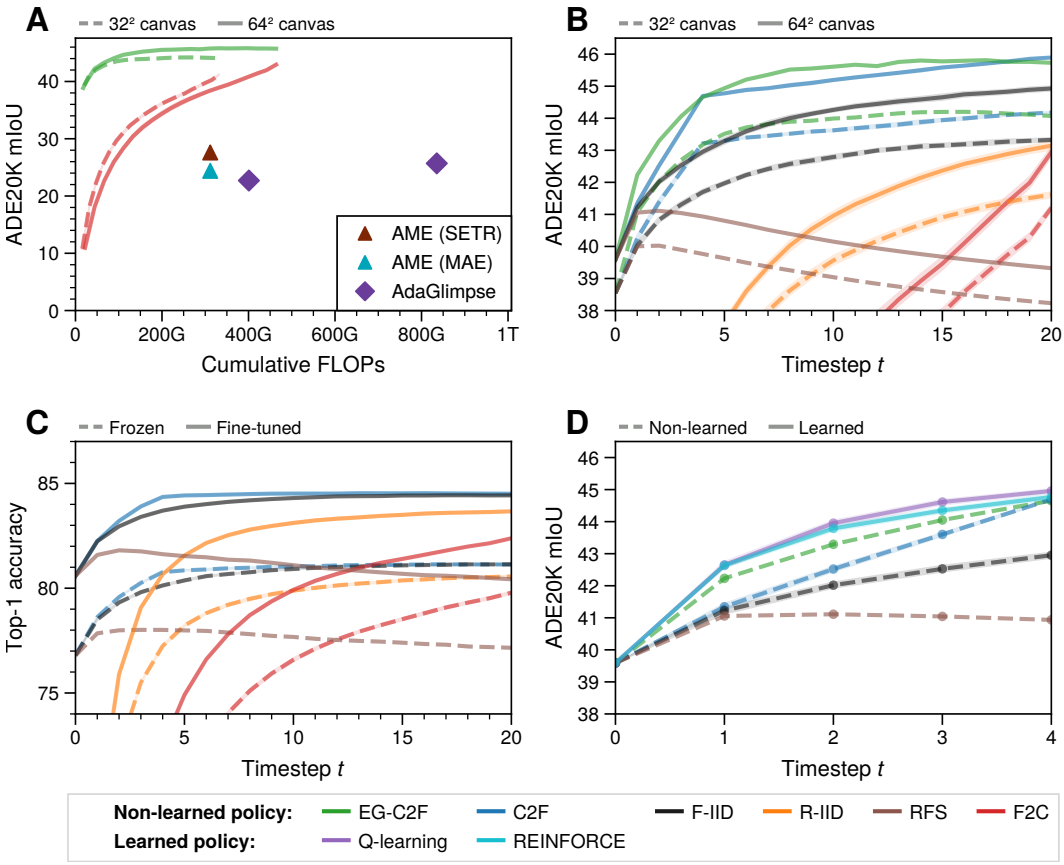

Figure 4: **Benchmark results on ADE20K and ImageNet-1k**. **A.** Accuracy–efficiency frontier on ADE20K segmentation. **B.** ADE20K segmentation mIoU by non-learned policy (32^2 or 64^2 canvas). **C.** ImageNet-1k classification accuracy by non-learned policy, frozen or fine-tuned (32^2 canvas). **D.** Learned vs. non-learned policies on short-horizon ADE20K segmentation (64^2 canvas).

Baselines. We consider leading active vision models with published results on ADE20K segmentation and/or ImageNet-1k classification: Saccader[19], AME[24], AdaGlimpse[25], and AdaptiveNN[26]. We report each model's best published results; for AME, we include both its SETR[68] and MAE[38]-initialized variants for completeness. When assessing the effects of canvas resolution and ground-truth mask area, we also compare against the DINOv3 ViT-B passive teacher.

Non-learned policies. To assess CanViT's ability to generalize across policies, we supplement the R-IID and F-IID train-time policies described in §5.2 with additional inference-time-only policies. We introduce a **Coarse-to-Fine (C2F)** policy, which traverses a quadtree over the scene, deterministically decreasing the viewpoint scale as the rollout progresses while randomizing the visitation order at each scale. To isolate the effect of processing order, we pair C2F with the **Fine-to-Coarse (F2C)** policy, which reverses C2F viewpoint sequences. For ADE20K, we also introduce an **Entropy-Guided C2F (EG-C2F)** variant, which greedily selects the highest-uncertainty tile among those that have not yet been visited at a given scale, using the segmentation probe's per-position class entropy. This image-dependent dynamic policy showcases the ability of the canvas to guide viewpoint selection, even without RL. Lastly, to disentangle the impact of additional recurrent processing from that of ingesting different inputs, we consider a **Repeated Full-Scene (RFS)** policy, which simply iterates over the $(x, y, s) = (0, 0, 1)$ zoomed-out viewpoint. We use up to $T = 21$ glimpses.

Learned policies. To test whether a task-specific viewing policy can be learned quickly on top of a frozen CanViT-B and its ADE20K probe, we train policy networks with Q-learning[69] and REINFORCE[70] to maximize segmentation quality (Appendix D.2). We train each network on 640,000 glimpses, 0.06% of CanViT-B's pretraining glimpses, in under 95 minutes on a single RTX 4090 GPU (24 GB).

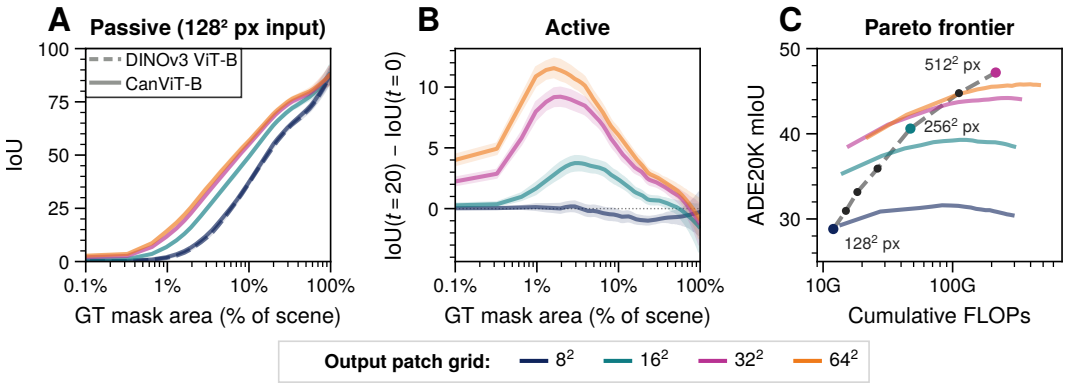

Figure 5: **Effects of canvas resolution and ground-truth mask area.** We evaluate on ADE20K segmentation with canvas resolution in $\{8^2, 16^2, 32^2, 64^2\}$, using the EG-C2F policy, frozen weights, and per-resolution linear probes. **A.** Per-mask IoU given a single full-scene 128^2 px input (8^2 input patch grid). **B.** Per-mask IoU gain given additional glimpses. **C.** Accuracy–efficiency Pareto frontier, varying FLOPs via timestep count for CanViT and via input resolution (px) for DINOv3.

6.1 Results

CanViT-B outperforms prior active vision models on both ADE20K segmentation and ImageNet-1k classification by a wide margin (Table 1), achieving up to 45.9% ADE20K mIoU (up from 27.6%) and 84.5% IN1k top-1 accuracy (up from 82.2%). On ADE20K, state-of-the-art accuracy is paired with striking efficiency: in a single glimpse, CanViT-B reaches 38.5% mIoU using 20× fewer FLOPs than AME does at 27.6% mIoU (Figure 4**A**). CanViT's advantage holds even with the F2C policy's worse-than-random viewing order, showing that perception, not viewpoint selection, was the bottleneck in this task.

Table 1: **Comparison with prior work.** †: Frozen weights with linear probing.

Model	ADE20K mIoU	IN1k top-1
Saccader[19]	—	75.0
AME[24] (MAE)	24.4	—
AME[24] (SETR)	27.6	—
AdaGlimpse[25]	25.7	77.5
AdaptiveNN[26]	—	82.2
CanViT-B	**45.9**†	81.1† / **84.5**

On both tasks, CanViT-B generalizes across policies, temporal horizons, and canvas resolutions, even with frozen weights (Figure 4**B,C**). We observe consistent ADE20K mIoU gains through $T = 21$ glimpses, well beyond the average of $T \approx 4$ used during pretraining, while IN1k accuracy quickly plateaus. The inference-time-only C2F policy dramatically outperforms the R-IID and F-IID policies on ADE20K in both peak accuracy and efficiency, and provides a modest efficiency boost early into IN1k rollouts. EG-C2F's uncertainty-based viewpoint selection enables further efficiency gains on ADE20K. ADE20K Q-learning policies outperform every non-learned policy from $t = 1$ to $t = 4$ (Figure 4**D**). On both tasks, CanViT reaches higher peak accuracies with C2F than F2C despite both policies yielding the same set of viewpoints by $t = 20$, highlighting the importance of contextually informed processing. Given a constant input (RFS), accuracy initially improves purely from recurrent processing, then subsequently declines in the absence of diverse viewpoints. Despite training exclusively with a 32^2 canvas, evaluating CanViT-B with an inference-time 64^2 canvas consistently provides an accuracy boost on ADE20K, an effect that we probe further in Figure 5.

CanViT's dual-stream design decouples glimpse (instantaneous vision) resolution from canvas (memory/output) resolution. This input-output dissociation can be advantageous even in single-timestep (passive vision) contexts: given a low-resolution full-scene glimpse, a frozen CanViT-B with a finer canvas outperforms its coarser-canvas counterpart on ADE20K segmentation, and even its input-matched DINOv3 ViT-B teacher (Figure 5**A**). Finer canvases are most beneficial when dealing with small ground-truth segmentation masks (e.g., for small, distant or occluded objects) (Figure 5**A**) or using multiple timesteps (Figure 5**B**). Together, the accuracy gains provided by finer canvases and the low computational overhead of Canvas Attention allow CanViT-B to offer an attractive accuracy-efficiency frontier on ADE20K segmentation at low FLOP budgets (Figure 5**C**).

7 Conclusion

CanViT applies the foundation model playbook to active vision, with a general-purpose active-vision model that can be used as-is across tasks and viewing policies while delivering outstanding performance. Our results show that pairing a carefully designed active-vision architecture with a highly informative learning signal is sufficient to dramatically advance the state of the art in active computer vision, without relying on complex RL pipelines. Together, CanViT's strong inference-time generalization, high computational efficiency and state-of-the-art performance on active-vision benchmarks highlight the potential of our approach and of task- and policy-agnostic AVFM pretraining.

7.1 Limitations and future work

Dynamic scenes. Like most active-vision models, CanViT was trained and evaluated on static scenes. However, its constant-memory recurrent design and high computational efficiency make it a strong candidate for future adaptation to real-time video processing and embodied active perception. In a changing environment, the addition of a lightweight gating mechanism may be desirable to facilitate forgetting, as the contents of the canvas may become stale when objects move.

Foveated sensors. CanViT accepts a different zoom level for each glimpse, but not for each patch within a glimpse. Future extensions of CanViT could incorporate a more human-like sensory front end, with a high-resolution fovea and a low-resolution periphery[71–75], making it a more faithful computational model of human active vision.

Teacher-free pretraining. Our passive-to-active distillation scheme accelerates active-vision pretraining by allowing it to remain focused on active-vision-specific considerations such as partial observability and information integration across glimpses, but requires a pretrained passive teacher. As a result, while much of CanViT's efficiency can be attributed to its architecture, its absolute performance also reflects the strength of its DINOv3 teacher. Moreover, the need for a teacher may prove limiting at very high scene resolutions due to the quadratic cost of teacher self-attention and the large storage footprint of precomputed features, especially when working with video, which would involve distinct per-timestep targets. We thus see pretraining paradigms that do not require an external pretrained teacher, such as active-to-active self-distillation and pixel-space reconstruction, as promising avenues for future research.

Policy learning. Our RL-trained viewing policies operate on top of a frozen CanViT-B and its ADE20K probe, choose among a fixed set of viewpoints in episodes of 5 glimpses, and optimize only the immediate reward. Policies that plan over longer horizons, choose from a continuous range of viewpoints, are conditioned on a goal given as input (such as a target category or a text query), or are trained jointly with CanViT would be natural extensions.

Scale. Our results use a single model size pretrained with a modest compute budget on a dataset over $100\times$ smaller than DINOv3's LVD-1689M; we expect further gains from more extensive pretraining and larger models, though the latter may be undesirable for edge deployment.

Other tasks. We evaluated CanViT-B on ADE20K segmentation and ImageNet-1k classification, with a preliminary experiment on COCO object detection (Appendix H.5). We expect CanViT-B to be applicable to other tasks, such as monocular depth estimation, without retraining, similarly to its DINOv3 teacher and passive models distilled from DINOv2/v3 features[40].

Acknowledgments and Disclosure of Funding

Funded by a Natural Sciences and Engineering Research Council of Canada (NSERC) Canada Graduate Research Scholarship – Doctoral (CGRS D-613218-2026), a Vision Sciences Research Network (VSRN) PhD Recruitment Scholarship, Unifying Neuroscience and Artificial Intelligence – Québec (UNIQUE) MSc and PhD Excellence Scholarships, a Centre for Applied Mathematics in Bioscience and Medicine (CAMBAM) Fellowship, a McGill Department of Physiology Max and Jane Childress Entrance Fellowship, and a McGill Faculty of Medicine and Health Sciences Internal Studentship to Yohaï-Eliel Berreby; an NSERC Undergraduate Student Research Award administered by the McGill Faculty of Medicine and Health Sciences to Sabrina Du; a Canada CIFAR AI Chair to Audrey Durand; and an NSERC Discovery Research program grant (RGPIN-2022-05399) and Supplement (DGECR-2022-00321) together with a computing resources grant from

Calcul Québec and the Digital Research Alliance of Canada to B. Suresh Krishna. This project received support from Lamarck Labs (Enactant, Inc.).

References

[1] Daniel L. K. Yamins, Ha Hong, Charles F. Cadieu, Ethan A. Solomon, Darren Seibert, and James J. DiCarlo. Performance-Optimized Hierarchical Models Predict Neural Responses in Higher Visual Cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014. doi: 10.1073/pnas.1403112111.

[2] Daniel L. K. Yamins and James J. DiCarlo. Using Goal-Driven Deep Learning Models to Understand Sensory Cortex. *Nature Neuroscience*, 19(3):356–365, 2016. doi: 10.1038/nn.4244.

[3] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J. Majaj, Rishi Rajalingham, Elias B. Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, Kailyn Schmidt, Daniel L. K. Yamins, and James J. DiCarlo. Brain-Score: Which Artificial Neural Network for Object Recognition Is Most Brain-Like?, 2018. bioRxiv preprint, doi: 10.1101/407007.

[4] Chengxu Zhuang, Siming Yan, Aran Nayebi, Martin Schrimpf, Michael C. Frank, James J. DiCarlo, and Daniel L. K. Yamins. Unsupervised Neural Network Models of the Ventral Visual Stream. *Proceedings of the National Academy of Sciences*, 118(3):e2014196118, 2021. doi: 10.1073/pnas.2014196118.

[5] Shahab Bakhtiari, Patrick Mineault, Timothy Lillicrap, Christopher Pack, and Blake Richards. The Functional Specialization of Visual Cortex Emerges from Training Parallel Pathways with Self-Supervised Predictive Learning. In *Advances in Neural Information Processing Systems*, volume 34, pages 25164–25178. Curran Associates, Inc., 2021.

[6] Johannes Mehrer, Courtney J. Spoerer, Emer C. Jones, Nikolaus Kriegeskorte, and Tim C. Kietzmann. An Ecologically Motivated Image Dataset for Deep Learning Yields Better Models of Human Vision. *Proceedings of the National Academy of Sciences*, 118(8):e2011417118, 2021. doi: 10.1073/pnas.2011417118.

[7] Joséphine Raugel, Marc Szafraniec, Huy V. Vo, Camille Couprie, Jérémy Rapin, Stéphane d'Ascoli, Patrick Labatut, Piotr Bojanowski, Valentin Wyart, and Jean-Remi King. Disentangling the Factors of Convergence between Brains and DINOv3. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=i99ccgfad8.

[8] Alfred L. Yarbus. *Eye Movements and Vision*. Springer US, Boston, MA, 1967. doi: 10.1007/978-1-4899-5379-7.

[9] David Hoppe and Constantin A. Rothkopf. Multi-Step Planning of Eye Movements in Visual Search. *Scientific Reports*, 9(1):144, 2019. doi: 10.1038/s41598-018-37536-0.

[10] Alan D. Baddeley and Graham Hitch. Working Memory. In Gordon H. Bower, editor, *Psychology of Learning and Motivation*, volume 8 of *Psychology of Learning and Motivation*, pages 47–89. Academic Press, 1974. doi: 10.1016/S0079-7421(08)60452-1.

[11] David Melcher. Persistence of Visual Memory for Scenes. *Nature*, 412(6845):401, 2001. doi: 10.1038/35086646.

[12] Rajesh P. N. Rao and Dana H. Ballard. Predictive Coding in the Visual Cortex: A Functional Interpretation of Some Extra-Classical Receptive-Field Effects. *Nature Neuroscience*, 2(1):79–87, 1999. doi: 10.1038/4580.

[13] Charles D. Gilbert and Wu Li. Top-down Influences on Visual Processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013. doi: 10.1038/nrn3476.

[14] Kohitij Kar, Jonas Kubilius, Kailyn Schmidt, Elias B. Issa, and James J. DiCarlo. Evidence That Recurrent Circuits Are Critical to the Ventral Stream's Execution of Core Object Recognition Behavior. *Nature Neuroscience*, 22(6):974–983, 2019. doi: 10.1038/s41593-019-0392-5.

[15] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. Recurrent Models of Visual Attention. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[16] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple Object Recognition with Visual Attention. In *International Conference on Learning Representations*, 2015.

[17] Stefan Mathe, Aleksis Pirinen, and Cristian Sminchisescu. Reinforcement Learning for Visual Object Detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2894–2902, 2016. doi: 10.1109/CVPR.2016.316.

[18] Artsiom Ablavatski, Shijian Lu, and Jianfei Cai. Enriched Deep Recurrent Visual Attention Model for Multiple Object Recognition. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 971–978. IEEE, 2017.

[19] Gamaleldin F. Elsayed, Simon Kornblith, and Quoc V. Le. Saccader: Improving Accuracy of Hard Attention Models for Vision. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[20] Yulin Wang, Kangchen Lv, Rui Huang, Shiji Song, Le Yang, and Gao Huang. Glance and Focus: A Dynamic Approach to Reducing Spatial Redundancy in Image Classification. In *Advances in Neural Information Processing Systems*, volume 33, pages 2432–2444. Curran Associates, Inc., 2020.

[21] Athanasios N. Papadopoulos, Paweł Korus, and Nasir Memon. Hard-Attention for Scalable Image Classification. In *Advances in Neural Information Processing Systems*, volume 34, pages 14694–14707. Curran Associates, Inc., 2021.

[22] Soroush Seifi, Abhishek Jha, and Tinne Tuytelaars. Glimpse-Attend-and-Explore: Self-Attention for Active Visual Exploration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16137–16146, 2021.

[23] Jiayang Liu, Yiming Bu, Daniel Ts'o, and Qinru Qiu. Improved Efficiency Based on Learned Saccade and Continuous Scene Reconstruction From Foveated Visual Sampling. In *International Conference on Learning Representations*, 2024.

[24] Adam Pardyl, Grzegorz Rypeść, Grzegorz Kurzejamski, Bartosz Zieliński, and Tomasz Trzciński. Active Visual Exploration Based on Attention-Map Entropy. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1303–1311. ijcai.org, 2023. doi: 10.24963/IJCAI.2023/145.

[25] Adam Pardyl, Michał Wronka, Maciej Wołczyk, Kamil Adamczewski, Tomasz Trzciński, and Bartosz Zieliński. AdaGlimpse: Active Visual Exploration with Arbitrary Glimpse Position and Scale. In *Computer Vision – ECCV 2024*, pages 112–129, Cham, 2025. Springer Nature Switzerland. doi: 10.1007/978-3-031-72664-4_7.

[26] Yulin Wang, Yang Yue, Yang Yue, Huanqian Wang, Haojun Jiang, Yizeng Han, Zanlin Ni, Yifan Pu, Minglei Shi, Rui Lu, Qisen Yang, Andrew Zhao, Zhuofan Xia, Shiji Song, and Gao Huang. Emulating Human-like Adaptive Vision for Efficient and Flexible Machine Visual Perception. *Nature Machine Intelligence*, 7(11):1804–1822, 2025. doi: 10.1038/s42256-025-01130-7.

[27] Motahareh Pourrahimi and Pouya Bashivan. Neural Signatures of Associational Cortex Emerge in a Goal-Directed Model of Visual Search, 2025. bioRxiv preprint, doi: 10.1101/2025.06.06.658387.

[28] Timur Ibrayev, Manish Nagaraj, Amitangshu Mukherjee, and Kaushik Roy. Exploring Foveation and Saccade for Improved Weakly-Supervised Localization. In *Proceedings of the 2nd Gaze Meets ML Workshop*, volume 226 of *Proceedings of Machine Learning Research*, pages 61–89. PMLR, 2024.

[29] Timur Ibrayev, Amitangshu Mukherjee, Sai Aparna Aketi, and Kaushik Roy. Toward Two-Stream Foveation-Based Active Vision Learning. *IEEE Transactions on Cognitive and Developmental Systems*, 16(5):1843–1860, 2024. doi: 10.1109/TCDS.2024.3390597.

[30] Mengye Ren and Richard S. Zemel. End-To-End Instance Segmentation With Recurrent Attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6656–6664, 2017.

[31] Jason Li, Nicholas Watters, Hansem Sohn, and Mehrdad Jazayeri. Modeling Human Eye Movements with Neural Networks in a Maze-Solving Task. In *Proceedings of the Gaze Meets ML Workshop*, pages 98–112. PMLR, 2023.

[32] Jan Olszewski, Dawid Rymarczyk, Piotr Wójcik, Mateusz Pach, and Bartosz Zieliński. TORE: Token Recycling in Vision Transformers for Efficient Active Visual Exploration. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 8606–8616, 2025. doi: 10.1109/WACV61041.2025.00834.

[33] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal Attention for Long-Range Interactions in Vision Transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 30008–30022. Curran Associates, Inc., 2021.

[34] Jianwei Yang, Chunyuan Li, Xiyang Dai, and Jianfeng Gao. Focal Modulation Networks. In *Advances in Neural Information Processing Systems*, volume 35, pages 4203–4217. Curran Associates, Inc., 2022. doi: 10.52202/068431-0304.

[35] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene Parsing Through ADE20K Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 633–641, 2017.

[36] Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, Francisco Massa, Daniel Haziza, Luca Wehrstedt, Jianyuan Wang, Timothée Darcet, Théo Moutakanni, Leonel Sentana, Claire Roberts, Andrea Vedaldi, Jamie Tolan, John Brandt, Camille Couprie, Julien Mairal, Hervé Jégou, Patrick Labatut, and Piotr Bojanowski. DINOv3. *Transactions on Machine Learning Research*, 2026. ISSN 2835-8856. URL https://openreview.net/forum?id=2NlGyqNjns. Featured Certification.

[37] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[38] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked Autoencoders Are Scalable Vision Learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[39] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

[40] Yitian Zhang, Xu Ma, Yue Bai, Huan Wang, and Yun Fu. Accessing Vision Foundation Models via ImageNet-1K. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=LC6ZtQV6u2.

[41] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set Transformer: A Framework for Attention-based Permutation-Invariant Neural Networks. In *Proceedings of the International Conference on Machine Learning*, pages 3744–3753. PMLR, 2019.

[42] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General Perception with Iterative Attention. In *Proceedings of the International Conference on Machine Learning*, pages 4651–4664. PMLR, 2021.

[43] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier J. Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver IO: A General Architecture for Structured Inputs & Outputs. In *International Conference on Learning Representations*, 2022.

[44] Allan Jabri, David J. Fleet, and Ting Chen. Scalable Adaptive Computation for Iterative Generation. In *Proceedings of the International Conference on Machine Learning*, pages 14569–14589. PMLR, 2023.

[45] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, 2014. arXiv preprint arXiv:1412.3555.

[46] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9 (8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

[47] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal Transformers. In *International Conference on Learning Representations*, 2019.

[48] Liu Yang, Kangwook Lee, Robert D. Nowak, and Dimitris Papailiopoulos. Looped Transformers Are Better at Learning Learning Algorithms. In *International Conference on Learning Representations*, 2024.

[49] Nikunj Saunshi, Nishanth Dikkala, Zhiyuan Li, Sanjiv Kumar, and Sashank J. Reddi. Reasoning with Latent Thoughts: On the Power of Looped Transformers. In *International Conference on Learning Representations*, 2025.

[50] Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi Yadkori. Hierarchical Reasoning Model, 2025. arXiv preprint arXiv:2506.21734.

[51] Alexia Jolicoeur-Martineau. Less Is More: Recursive Reasoning with Tiny Networks, 2025. arXiv preprint arXiv:2510.04871.

[52] Alex Graves. Adaptive Computation Time for Recurrent Neural Networks, 2017. arXiv preprint arXiv:1603.08983.

[53] Andrea Banino, Jan Balaguer, and Charles Blundell. PonderNet: Learning to Ponder. In *ICML Workshop on Automated Machine Learning (AutoML)*, 2021.

[54] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason E. Weston, and Yuandong Tian. Training Large Language Models to Reason in a Continuous Latent Space. In *Second Conference on Language Modeling*, 2025.

[55] Jonas Geiping, Sean Michael McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian R. Bartoldson, Bhavya Kailkhura, Abhinav Bhatele, and Tom Goldstein. Scaling up Test-Time Compute with Latent Reasoning: A Recurrent Depth Approach. In *Advances in Neural Information Processing Systems*, volume 38. Curran Associates, Inc., 2025. doi: 10.52202/085713-1380.

[56] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision Transformers Need Registers. In *International Conference on Learning Representations*, 2024.

[57] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021.

[58] Edward C. Tolman. Cognitive Maps in Rats and Men. *Psychological Review*, 55(4):189–208, 1948. doi: 10.1037/h0061626.

[59] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. *Neurocomputing*, 568:127063, 2024. doi: 10.1016/j.neucom.2023.127063.

[60] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary Position Embedding for Vision Transformer. In *Computer Vision – ECCV 2024*, pages 289–305, Cham, 2025. Springer Nature Switzerland. doi: 10.1007/978-3-031-72684-2_17.

[61] Jason Ansel, Edward Yang, Horace He, Natalia Gimelshein, Animesh Jain, Michael Voznesensky, Bin Bao, Peter Bell, David Berard, Evgeni Burovski, Geeta Chauhan, Anjali Chourdia, Will Constable, Alban Desmaison, Zachary DeVito, Elias Ellison, Will Feng, Jiong Gong, Michael Gschwind, Brian Hirsh, Sherlock Huang, Kshiteej Kalambarkar, Laurent Kirsch, Michael Lazos, Mario Lezcano, Yanbo Liang, Jason Liang, Yinghai Lu, C. K. Luk, Bert Maher, Yunjie Pan, Christian Puhrsch, Matthias Reso, Mark Saroufim, Marcos Yukio Siraichi, Helen Suk, Shunting Zhang, Michael Suo, Phil Tillet, Xu Zhao, Eikan Wang, Keren Zhou, Richard Zou, Xiaodong Wang, Ajit Mathews, William Wen, Gregory Chanan, Peng Wu, and Soumith Chintala. PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation. In *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, ASPLOS '24, pages 929–947, New York, NY, USA, 2024. Association for Computing Machinery. doi: 10.1145/3620665.3640366.

[62] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: Composable Transformations of Python+NumPy Programs, 2018. URL http://github.com/jax-ml/jax.

[63] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2026. URL http://github.com/google/flax.

[64] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization, 2016. arXiv preprint arXiv:1607.06450.

[65] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, pages 740–755. Springer International Publishing, 2014. doi: 10.1007/978-3-319-10602-1_48.

[66] Tal Ridnik, Emanuel Ben-Baruch, Asaf Noy, and Lihi Zelnik-Manor. ImageNet-21K Pretraining for the Masses. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

[67] Ananya Kumar, Aditi Raghunathan, Robbie Matthew Jones, Tengyu Ma, and Percy Liang. Fine-Tuning Can Distort Pretrained Features and Underperform Out-of-Distribution. In *International Conference on Learning Representations*, 2022.

[68] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking Semantic Segmentation From a Sequence-to-Sequence Perspective With Transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021.

[69] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3–4):279–292, 1992. doi: 10.1007/BF00992698.

[70] Ronald J. Williams. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8(3–4):229–256, 1992. doi: 10.1007/BF00992696.

[71] Brian Cheung, Eric Weiss, and Bruno A. Olshausen. Emergence of Foveal Image Sampling from Learning to Attend in Visual Scenes. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJJKxrsgl.

[72] George Killick, Paul Henderson, Paul Siebert, and Gerardo Aragon-Camarasa. Foveation in the Era of Deep Learning. In *British Machine Vision Conference (BMVC)*, 2023.

[73] Andrey Gizdov, Shimon Ullman, and Daniel Harari. Seeing More with Less: Human-like Representations in Vision Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4408–4417, 2025. doi: 10.1109/CVPR52734.2025.00416.

[74] Tanner Schmidt and Richard Newcombe. Segment This Thing: Foveated Tokenization for Efficient Point-Prompted Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 29428–29437, 2025. doi: 10.1109/CVPR52734.2025.02740.

[75] Nicholas Blauch, George A. Alvarez, and Talia Konkle. FOVI: A biologically-inspired foveated interface for deep vision models. In *Proceedings of the 43rd International Conference on Machine Learning*, pages 8446–8469. PMLR, 2026.

[76] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. In *Advances in Neural Information Processing Systems*, volume 33, pages 7537–7547. Curran Associates, Inc., 2020.

[77] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going Deeper With Image Transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 32–42, 2021.

[78] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28.

[79] Sanghyun Woo, Shoubhik Debnath, Ronghang Hu, Xinlei Chen, Zhuang Liu, In So Kweon, and Saining Xie. ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16133–16142, 2023. doi: 10.1109/CVPR52729.2023.01548.

[80] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft Actor-Critic Algorithms and Applications, 2018. arXiv preprint arXiv:1812.05905.

[81] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[82] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the International Conference on Machine Learning*, pages 448–456. PMLR, 2015.

[83] Skipper Seabold and Josef Perktold. statsmodels: Econometric and Statistical Modeling with Python. In *Proceedings of the Python in Science Conference*, pages 57–61, 2010.

[84] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A Next-generation Hyperparameter Optimization Framework. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, pages 2623–2631, New York, NY, USA, 2019. Association for Computing Machinery. doi: 10.1145/3292500.3330701.

[85] Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. Are We Done with ImageNet?, 2020. arXiv preprint arXiv:2006.07159.

[86] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as Points, 2019. arXiv preprint arXiv:1904.07850.

[87] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018. doi: 10.1109/TPAMI.2017.2723009.

Appendix

Contents

A Canvas Attention pseudocode **18**

B Viewpoint Encoding (VPE) **19**
B.1 Definitions: scene, viewpoint, crop 19
B.2 Finding a scale-invariant representation 19
B.3 Properties of u 19

C CanViT-B pretraining details **21**

D Evaluation details **21**
D.1 Non-learned viewing policies 21
D.2 Learned ADE20K viewing policies 22
D.3 FLOP counting 23
D.4 Task: ADE20K segmentation 24
D.5 Task: ImageNet-1k classification 24
D.6 Methodology for Figure 5 analyses 25
D.7 DINOv3 ImageNet-1k linear classification probes 27

E Pretraining ablations **29**

F Inference latency **32**

G Interpretability **33**

H Additional experiments **34**
H.1 Impact of the pretraining dataset 34
H.2 Viewpoint Encoding at long horizons and high resolution 37
H.3 ImageNet-1k accuracy against inference cost 37
H.4 Architecture control 38
H.5 COCO object detection 39

I Declarations **40**
I.1 Availability of code and data 40
I.2 Broader impact 40
I.3 Compute reporting 40
I.4 Statistical analysis 40
I.5 Licenses and attribution 41

A Canvas Attention pseudocode

```
class CanvasAttention(nn.Module):
    def __init__(self, D_q, D_kv):
        self.ln_q  = LayerNorm(D_q)
        self.ln_kv = LayerNorm(D_kv)

    # Common template for Reads and Writes
    def forward(self, x_q, x_kv, rope_q, rope_kv):
        q  = to_multihead(self.q_map(self.ln_q(x_q)))
        kv = self.ln_kv(x_kv)
        k  = to_multihead(self.k_map(kv))
        v  = to_multihead(self.v_map(kv))
        q  = apply_2d_rope(q, rope_q)
        k  = apply_2d_rope(k, rope_kv)
        return self.o_map(from_multihead(sdpa(q, k, v)))


class CanvasAttentionRead(CanvasAttention):  # backbone queries canvas
    def __init__(self, D_bb, D_can):
        super().__init__(D_q=D_bb, D_kv=D_can)
        # backbone-side: fully-connected Query and Output projections
        self.q_map = Linear(D_bb, D_can)
        self.o_map = Linear(D_can, D_bb)
        # canvas-side: no-op for Key and Value
        self.k_map = Identity()
        self.v_map = Identity()


class CanvasAttentionWrite(CanvasAttention):  # canvas queries backbone
    def __init__(self, D_bb, D_can):
        super().__init__(D_q=D_can, D_kv=D_bb)
        # backbone-side: fully-connected Key and Value projections
        self.k_map = Linear(D_bb, D_can)
        self.v_map = Linear(D_bb, D_can)
        # canvas-side: no-op for Query and Output
        self.q_map = Identity()
        self.o_map = Identity()


# cell centers of uniform R x C grid
# in [-1,+1]^2, shape [R*C, 2]
def grid(R, C):
    ys = (arange(R) + 0.5) / R * 2 - 1
    xs = (arange(C) + 0.5) / C * 2 - 1
    return stack(meshgrid(ys, xs), -1).reshape(R * C, 2)


# Scene-Relative 2D Rotary Position Embeddings
# center=(y,x) in [-1,+1]^2, scale in (0,1]: where/how zoomed-out the viewpoint is
rope_bb  = compute_2d_rope(center + scale * grid(H_g, W_g))  # dynamic
rope_can = compute_2d_rope(grid(H_c, W_c))                   # fixed

# CanViT alternates reads and writes across depth:
x_bb  = blk1(blk0(x_bb))
x_bb  = x_bb  + read(x_bb, x_can, rope_bb, rope_can)
x_bb  = blk3(blk2(x_bb))
x_can = x_can + write(x_can, x_bb, rope_can, rope_bb)
x_bb  = blk5(blk4(x_bb))
x_bb  = x_bb  + read(x_bb, x_can, rope_bb, rope_can)
# ...
```

B Viewpoint Encoding (VPE)

The VPE token is instantiated by first encoding the current viewpoint (x, y, s) as the triplet $(x/s, y/s, \log s)$, a parameterization with scale invariance, same-scale translation invariance, and planar isotropy properties (proven below). We then lift this triplet into $\mathbb{R}^{D_{\text{bb}}}$ via Random Fourier Features (RFF)[76], and apply layer normalization[64] before letting the backbone process it alongside the other glimpse tokens (patches, registers, and recurrent CLS token).

B.1 Definitions: scene, viewpoint, crop

We consider a finite 2D scene whose (x, y) coordinates span $[-1, +1]^2$.

We call a *viewpoint* a triplet $(x, y, s) \in \mathcal{V}_{\text{raw}}$ such that the corresponding square crop,

$$[x - s, x + s] \times [y - s, y + s],$$

lies inside $[-1, +1]^2$. Equivalently,

$$\mathcal{V}_{\text{raw}} = \{\, (x, y, s) \in \mathbb{R}^2 \times (0, 1] : |x| \le 1 - s,\ |y| \le 1 - s \,\}.$$

For instance, the viewpoint $(0, 0, 1)$ spans the entire scene; the viewpoint $(0.5, 0.5, 0.5)$ spans the quadrant $[0, 1]^2$; the viewpoint $(2, 2, 0.5)$ is invalid, as its center lies outside of the scene; the viewpoint $(0.5, 0.5, 1)$ is *also* invalid, even though its center lies within the scene, because its borders extend beyond the scene boundaries.

B.2 Finding a scale-invariant representation

While the above representation of a viewpoint as a triplet $(x, y, s) \in \mathcal{V}_{\text{raw}}$ is simple to understand and uniquely defines crops within the scene, it fails to represent an important property: **scale invariance**.

When considering viewpoints as vectors, we would like distances between viewpoints to be **invariant to global rescaling**. In other words, the distance between two side-by-side square crops should be identical regardless of zoom level; equivalently, a 10% zoom of the scene should leave any pairwise viewpoint distance unchanged. This is not the case in a straightforward (x, y, s) encoding, which becomes artificially insensitive to shifts in position and scale for small crops (i.e., when $s \ll 1$), thus being forced to under-represent fine detail at small scales, leading to loss of information, or over-represent it at large scales, leading to ill-conditioned representations with excessive sensitivity to small perturbations at large scales (when $s \approx 1$).

Beyond scale invariance, two further properties are natural: **same-scale translation invariance**, so that at a fixed zoom the encoding distance depends only on the relative offset between viewpoints, and **planar isotropy**, so that the encoding does not privilege a particular axis orientation. Because the scene is bounded, these properties are stated for transformations whose resulting viewpoints remain valid crops. We thus seek a smooth, injective $u : \mathcal{V}_{\text{raw}} \to \mathcal{V}$ that, equipping $\mathcal{V}$ with the Euclidean distance d inherited from $\mathbb{R}^3$, satisfies all three.

We propose the embedding $u : \mathcal{V}_{\text{raw}} \to \mathcal{V} = u(\mathcal{V}_{\text{raw}}) \subset \mathbb{R}^3$ defined by

$$\begin{aligned} u &: \mathcal{V}_{\text{raw}} \to \mathcal{V} \\ &\quad (x, y, s) \mapsto (x/s, y/s, \log s) = (u_1, u_2, u_3) \\ u^{-1} &: \mathcal{V} \to \mathcal{V}_{\text{raw}} \\ &\quad (u_1, u_2, u_3) \mapsto (\exp(u_3)u_1, \exp(u_3)u_2, \exp(u_3)) = (x, y, s). \end{aligned}$$

The component functions are smooth for $s > 0$, and the displayed inverse shows that u is injective.

B.3 Properties of u

Throughout, $q_i = (x_i, y_i, s_i) \in \mathcal{V}_{\text{raw}}$ and d is the Euclidean distance on $\mathcal{V} \subset \mathbb{R}^3$.

Lemma B.1 (Pairwise distance identity)**.** *For $q_i = (x_i, y_i, s_i) \in \mathcal{V}_{\text{raw}}$,*

$$\begin{aligned} d(u(q_1), u(q_2))^2 &= \|u(q_1) - u(q_2)\|_2^2 \\ &= \|(x_1/s_1 - x_2/s_2,\ y_1/s_1 - y_2/s_2,\ \log s_1 - \log s_2)\|_2^2 \\ &= (x_1/s_1 - x_2/s_2)^2 + (y_1/s_1 - y_2/s_2)^2 + (\log s_1 - \log s_2)^2. \end{aligned}$$

Proof. Direct computation from the definition of u and the Euclidean norm on $\mathcal{V} \subset \mathbb{R}^3$. □

Proposition B.2 (Scale invariance)**.** *For all $c > 0$ such that $cq_i = (cx_i, cy_i, cs_i) \in \mathcal{V}_{\text{raw}}$ for $i = 1, 2$,*

$$d(u(cq_1), u(cq_2)) = d(u(q_1), u(q_2)).$$

Proof. Let $c > 0$ and $q_i = (x_i, y_i, s_i)$. By the definition of u,

$$u(cq_i) = (x_i/s_i,\ y_i/s_i,\ \log s_i + \log c),$$

so the additive $\log c$ cancels in the difference:

$$u(cq_1) - u(cq_2) = \left(\frac{x_1}{s_1} - \frac{x_2}{s_2},\ \frac{y_1}{s_1} - \frac{y_2}{s_2},\ \log s_1 - \log s_2 \right).$$

Setting $c = 1$ in this identity yields $u(q_1) - u(q_2)$ component-wise; the two difference vectors are therefore equal, hence so are their norms. □

Proposition B.3 (Same-scale translation invariance)**.** *With $(x_1, y_1, s), (x_2, y_2, s) \in \mathcal{V}_{\text{raw}}$, for any planar offset $(\Delta_x, \Delta_y) \in \mathbb{R}^2$ such that $(x_1 + \Delta_x, y_1 + \Delta_y, s)$ and $(x_2 + \Delta_x, y_2 + \Delta_y, s)$ lie in $\mathcal{V}_{\text{raw}}$,*

$$d(u(x_1 + \Delta_x, y_1 + \Delta_y, s), u(x_2 + \Delta_x, y_2 + \Delta_y, s)) = d(u(x_1, y_1, s), u(x_2, y_2, s)).$$

Proof. By Lemma B.1,

$$d(u(x_1, y_1, s), u(x_2, y_2, s))^2 = \left(\frac{x_1 - x_2}{s} \right)^2 + \left(\frac{y_1 - y_2}{s} \right)^2,$$

and

$$\begin{aligned} d(u(x_1 + \Delta_x, y_1 + \Delta_y, s), u(x_2 + \Delta_x, y_2 + \Delta_y, s))^2 &= \left(\frac{(x_1 + \Delta_x) - (x_2 + \Delta_x)}{s} \right)^2 \\ &\quad + \left(\frac{(y_1 + \Delta_y) - (y_2 + \Delta_y)}{s} \right)^2 \\ &= \left(\frac{x_1 - x_2}{s} \right)^2 + \left(\frac{y_1 - y_2}{s} \right)^2. \end{aligned}$$

The two right-hand sides agree. □

Proposition B.4 (Planar isotropy)**.** *For any $Q \in \mathrm{O}(2)$, with $(x_{i,Q}, y_{i,Q}, s_i) \in \mathcal{V}_{\text{raw}}$ for $i = 1, 2$ denoting the transformed coordinates $(x_{i,Q}, y_{i,Q}) = Q(x_i, y_i)$,*

$$d(u(x_{1,Q}, y_{1,Q}, s_1), u(x_{2,Q}, y_{2,Q}, s_2)) = d(u(x_1, y_1, s_1), u(x_2, y_2, s_2)).$$

Proof. By Lemma B.1,

$$d(u(x_1, y_1, s_1), u(x_2, y_2, s_2))^2 = \|(x_1/s_1, y_1/s_1) - (x_2/s_2, y_2/s_2)\|_2^2 + (\log s_1 - \log s_2)^2.$$

For any $Q \in \mathrm{O}(2)$,

$$\begin{aligned} d(u(x_{1,Q}, y_{1,Q}, s_1), u(x_{2,Q}, y_{2,Q}, s_2))^2 &= \|Q(x_1, y_1)/s_1 - Q(x_2, y_2)/s_2\|_2^2 \\ &\quad + (\log s_1 - \log s_2)^2 \\ &= \|Q((x_1/s_1, y_1/s_1) - (x_2/s_2, y_2/s_2))\|_2^2 \\ &\quad + (\log s_1 - \log s_2)^2 \\ &= \|(x_1/s_1, y_1/s_1) - (x_2/s_2, y_2/s_2)\|_2^2 \\ &\quad + (\log s_1 - \log s_2)^2, \end{aligned}$$

where the second equality uses linearity of Q, and the third uses orthogonality $\|Qv\|_2 = \|v\|_2$ for all $v \in \mathbb{R}^2$. The result follows. □

C CanViT-B pretraining details

Hyperparameters. We list architectural hyperparameters in Table 2, and ImageNet-21k pretraining hyperparameters in Table 3. Similarly to DINOv3 ViTs, we use LayerScale [77] in the ViT backbone.

Precomputation of teacher features. To eliminate the repeated cost of the teacher's forward pass during training, we precompute DINOv3 ViT-B features for all 13.2 million ImageNet-21k images at 512^2 px input resolution, which corresponds to a 32×32 patch grid. Images are processed without augmentation (resize shortest side to 512 px, center crop). We performed this one-time export in parallel over many MIG 1g.10gb H100 instances, with a small total footprint of approximately 8 H100-equivalent hours for our entire training set. We store features in float16, totaling approximately 19 TiB, and organize them into shards of 4096 images each. Each shard contains whole-scene dense patch features and CLS tokens, both obtained after DINOv3's final LayerNorm. Per-position z-score standardization statistics (mean and variance for each spatial position and embedding dimension) are precomputed from a representative sample of 4096 images. We pre-shuffle shards during dataset export, enabling us to process the shards themselves sequentially during pretraining. This strategy enables high-bandwidth streaming from networked storage, without incurring the high cost of a random data access pattern.

Pretraining compute. Our pretraining used 174 H100-equivalent hours: 166 for training and 8 for the one-time teacher-feature export. This is comparable to or lower than AdaGlimpse's training, which we estimate at 210–410 H100-equivalent hours: its official code release[†] reports that training "takes around 1 week on 4x A100 GPUs" for ImageNet-1k, and we account for the A100's lower bfloat16 FLOP/s and memory bandwidth. Since our pretraining covers approximately 10 epochs of ImageNet-21k, computing teacher features online would only increase this total to $166 + 80 = 246$ H100-equivalent hours, while reducing storage from 19 TiB to zero. This would not be an advantageous tradeoff in a static-images setting with abundant storage, but becomes relevant under storage constraints or for future extension to video.

Numerical precision considerations. The use of mixed-precision training is critical for efficiency, but comes with numerical correctness considerations, particularly in long-horizon scenarios. Over the course of this project, we encountered several subtle correctness issues, which only led to meaningful regressions after several hundred thousand steps of pretraining. For safe CanViT pretraining with minimal performance impact, we keep the canvas in float32 as it accumulates updates across time and depth, and keep all coordinate-related components in float32: grid coordinates, RoPE computations, VPE token projection matrix and creation. Other operations, including SDPA operations and learned projection layers, can be safely performed in bfloat16, following standard autocast operator rules. We perform the backward pass outside of the AMP region, setting `backward_pass_autocast="off"` accordingly when using `torch.compile`.

D Evaluation details

D.1 Non-learned viewing policies

Full-then-IID (F-IID) and **Random-then-IID (R-IID)** are described as part of our pretraining scheme (Section 5). F-IID begins with a full-scene viewpoint ($s = 1$) then samples i.i.d. random crops, whereas R-IID samples i.i.d. random crops at all timesteps, including $t = 0$.

Coarse-to-Fine (C2F) traverses a quadtree over the scene from coarse to fine. At level $\ell \geq 0$, the viewpoint scale is $s_\ell = 2^{-\ell}$, tiling the scene into a $2^\ell \times 2^\ell$ grid of non-overlapping crops. Let V_ℓ denote the ordered set of viewpoints at level ℓ; within each level, we visit tiles in a random order $\sigma_\ell(V_\ell)$ (where σ_ℓ is an independent random permutation). The full sequence is the concatenation $V_0, \sigma_1(V_1), \sigma_2(V_2), \ldots$, truncated to the glimpse budget T.

Fine-to-Coarse (F2C) reverses C2F viewpoint sequences. For any given glimpse budget, F2C starts from the finest level and progresses toward coarser views. After identical image coverage, the C2F vs. F2C comparison isolates the effect of processing order.

Entropy-Guided C2F (EG-C2F) is a variant of C2F that prioritizes informative regions at each scale level. Rather than visiting tiles in random order, EG-C2F decodes the current canvas with

[†] https://github.com/apardyl/AdaGlimpse

Table 2: **CanViT-B architecture.**

Hyperparameter	Value
Backbone	ViT-B/16
Backbone embedding dim	768
Backbone registers (ephemeral)	5
Glimpse patch size	16^2 px
Canvas embedding dim	1024
Canvas registers (persistent)	16
Canvas Attention heads	8
Canvas Attention head dim.	128
Canvas Attention R/W stride	2
RoPE base period	100
RoPE precision	float32
VPE token	Enabled
VPE RFF[76] std. deviation σ	1

Table 3: **CanViT-B pretraining hyperparameters.**

Hyperparameter	Value
Optimizer	AdamW
Initial learning rate	1×10^{-7}
Peak learning rate	4×10^{-4}
LR schedule	Warmup $\rightarrow$ Const.
LR warmup steps	100,000
Weight decay	1×10^{-4}
Gradient clipping	1.0 max norm
AdamW β_1, β_2	0.9, 0.999
TBPTT chunk size	$K = 2$ glimpses
Stop prob.	$p_{\text{stop}} = 0.5$
Rollouts per step	1 F-IID + 1 R-IID

Hyperparameter	Value
Dataset	ImageNet-21k
IN21k version	winter21_whole
Preprocessing	Resize(512), CenterCrop
Data augmentation	None
Scene resolution	512^2 px (1024 patches)
Glimpse resolution	128^2 px (64 patches)
Batch size	64 scenes
Total training steps	2×10^6
Min scale	0.05 (0.25% of area)
Forward precision	AMP bfloat16
ViT LayerScale init.	1×10^{-5}

the ADE20K probe before each glimpse, averages the Shannon entropy of the per-position class distribution over each unvisited tile, and visits the tile of highest mean entropy.

Repeated Full-Scene (RFS) repeats the viewpoint $(x, y, s) = (0, 0, 1)$ at every timestep. It serves as a recurrence-only control: any improvement over $t = 0$ must come from iterative canvas refinement with a fixed glimpse input.

D.2 Learned ADE20K viewing policies

We trained viewing policies with RL on top of a frozen CanViT-B and its ADE20K probe, using a 64×64 canvas.

Action space. As in C2F and EG-C2F, the policy first views the full scene ($t = 0$). At each subsequent timestep t, it chooses the viewpoint $\mathbf{v}_t$ from a fixed set of 512 candidates: for each scale $s \in \{0.5, 0.25\}$, a 16×16 grid of centers spanning $[-(1-s), 1-s]^2$, which keeps each glimpse entirely within the scene.

Network. We implement the policy network as a U-Net[78] built from ConvNeXt-V2 blocks[79]. Its input combines the current canvas, the canvas before the last glimpse, and the initial canvas. At each canvas position, the input holds the layer-normalized canvas features and the entropy of the probe's prediction, the change in both over the last glimpse, and the cosine dissimilarity of the features to the canvas before the last glimpse and to the initial canvas. We average-pool these inputs to a 32×32 grid. The network outputs one score $f_t(\mathbf{v})$ for each of the 512 candidate viewpoints $\mathbf{v}$.

Reward. We formulate the reward for the viewpoint chosen at timestep t as the relative reduction in segmentation cross-entropy after its glimpse:

$$r_t = \frac{\mathrm{CE}_{t-1} - \mathrm{CE}_t}{\mathrm{CE}_{t-1}}, \tag{4}$$

where CE_t is the probe's segmentation cross-entropy against the ADE20K labels after glimpse t.

Objectives. We compare two objectives, Q-learning[69] and REINFORCE[70]. For both, we set the discount factor to zero, so the policy optimizes the immediate reward. We standardize each timestep's rewards by running estimates of their mean and standard deviation, giving $\hat{r}_t$. For Q-learning, we train $f_t(\mathbf{v})$ to predict the standardized reward of choosing $\mathbf{v}$, with the squared error $(f_t(\mathbf{v}_t) - \hat{r}_t)^2$ and ε-greedy exploration ($\varepsilon = 0.5$). For REINFORCE, the scores f_t are the logits of a softmax policy π_t, from which we sample viewpoints during training, and we minimize $-\hat{r}_t \log \pi_t(\mathbf{v}_t) - \alpha\, \mathcal{H}(\pi_t)$, where $\mathcal{H}$ is the policy's entropy. As in Soft Actor-Critic[80], the entropy coefficient α is adjusted automatically: it increases when the mean entropy over a batch falls below a target of 1 nat, and decreases otherwise. At evaluation time, both policies choose the viewpoint with the highest score, without exploration.

Optimization. We trained each network for 8,000 optimizer steps with AdamW ($\beta_1 = 0.9$, $\beta_2 = 0.95$, learning rate 2×10^{-4} held constant after a 1,000-step linear warmup, weight decay 1×10^{-2}, gradient norm clipped at 1), on batches of 16 ADE20K training images with episodes of $T = 5$ glimpses. We trained 8 networks per objective, with different seeds. Every 1,000 optimizer steps, we evaluated the policy on the ADE20K validation set. For each training run, we report the checkpoint with the lowest mean validation cross-entropy over $t = 1$ to 4 (best) and the final checkpoint (last).

Evaluation. We evaluated the learned policies with the protocol of Appendix D.4 and compared them with the non-learned policies in Figure 4**D** (best checkpoints), and with C2F and EG-C2F in Table 4.

Table 4: **ADE20K mIoU (%) with learned policies, frozen CanViT-B (64×64 canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 8$ training runs for learned policies, $n = 10$ evaluation runs for C2F). †: one evaluation run of a deterministic policy. **Bold** = best policy at each timestep.

Policy	t=1	t=2	t=3	t=4
Non-learned				
— EG-C2F†	42.23	43.29	44.05	44.65
— C2F	41.34 ± 0.10	42.52 ± 0.08	43.60 ± 0.06	44.69 ± 0.03
Learned				
— Q-learning (best)	42.65 ± 0.10	**43.95 ± 0.10**	**44.61 ± 0.08**	**44.96 ± 0.06**
— Q-learning (last)	**42.71 ± 0.06**	**43.95 ± 0.08**	44.59 ± 0.06	44.89 ± 0.08
— REINFORCE (best)	42.64 ± 0.07	43.79 ± 0.08	44.35 ± 0.07	44.77 ± 0.09
— REINFORCE (last)	42.58 ± 0.09	43.75 ± 0.10	44.27 ± 0.11	44.66 ± 0.14

D.3 FLOP counting

We count each multiply-accumulate (MAC) operation as two floating-point operations (FLOPs). We sourced architecture parameters for DINOv3[36], AME[24] and AdaGlimpse[25] from the corresponding papers and code releases.† For AdaGlimpse's published results in Figure 4**A**, we count 3×3 tokens per glimpse, as its paper describes; for the architecture control (Table 23), we count the 2×2 tokens of its released code, which the control follows. To obtain fine-grained FLOP curves, we computed FLOP counts analytically, following the same standard formulas as PyTorch's `flop_counter`

†`https://github.com/facebookresearch/dinov3`, `https://github.com/apardyl/AME`, `https://github.com/apardyl/AdaGlimpse`

module. We validated our total counts end-to-end against traced FLOP counts as well as previously-reported FLOP counts, which were published for DINOv3 but not for AdaGlimpse or AME. Among the non-learned policies (Appendix D.1), all but EG-C2F have virtually zero associated cost; EG-C2F does require entropy computation, but this leverages already-present logits output by the segmentation probe, with an additional cost of approximately 1–2 MFLOP per selection with a 32^2 canvas and approximately 5 MFLOP with a 64^2 canvas (counting each exponential and logarithm as one operation), under 0.03% of a single CanViT-B forward pass. We did not include this cost in our FLOP counts because it is insignificant, similarly to the common omission of element-wise operations in FLOP accounting.

D.4 Task: ADE20K segmentation

Probe training. Each probe consists of dropout[81] ($p = 0.1$), batch normalization[82], and a linear classifier in the form of a 1×1 convolution. For CanViT, this head is preceded by a layer normalization[64] and applied to canvas patches. For the DINOv3 baselines, the head is applied directly to the feature map, as the DINOv3 features that we consider are already layer-normalized. We train for 40,000 steps with AdamW with peak learning rate 3×10^{-4} and weight decay 10^{-3}, under a 1,500-step linear warmup followed by cosine decay, at batch size 16. Training-time augmentations follow DINOv3's segmentation recipe: random scale-jittered 512^2 crops (scale factor in $[0.5, 2.0]$), horizontal flips, and photometric distortion. Crops in which a single class covers at least 75% of the labeled pixels are rejected and redrawn, up to ten times. During CanViT probe training, glimpse viewpoints are drawn i.i.d. under the R-IID policy across $T = 10$ timesteps.

Evaluation. Similarly to our active-vision ADE20K baselines[24,25], we resize all images and masks to a fixed size (here, 512×512). We apply this preprocessing consistently when evaluating CanViT-B and DINOv3 ViT-B, ensuring a fair comparison between the models. Across up to $T = 21$ timesteps, we report ADE20K mIoU by policy and timestep at canvas grids 32×32 and 64×64 on Figure 4**B** and Table 5. Table 6 reports best-t mIoU at canvas grids $c \times c$ for $c \in \{8, 16, 32, 64\}$. Figure 6 visualizes CanViT-B mIoU at canvas grids $c \times c$ ($c \in \{8, 16, 32, 64\}$) under the C2F policy across all $T = 21$ timesteps.

D.5 Task: ImageNet-1k classification

Probe training. As the DINOv3 authors did not release ImageNet-1k classification heads for models other than their 7B-parameter flagship, we trained and released our own linear probes for all five smaller ViT variants (Appendix D.7). Since CanViT-B was pretrained to predict DINOv3 representations, it is possible to use a DINOv3-fitted probe with it. We obtain the linear classification head applied to CanViT-B's recurrent CLS token by algebraically fusing three consecutive affine transforms (CanViT's pretrained CLS projection, CLS destandardization, and our DINOv3 ViT-B IN1k linear probe) into a single LayerNorm $\rightarrow$ Linear layer.

Fine-tuning. For ImageNet-1k full fine-tuning, we start from an already-fit frozen classification head then unfreeze CanViT and fine-tune all model and head parameters end-to-end, a two-stage approach known as LP-FT (linear probing then fine-tuning)[67]. We train for 20 epochs with AdamW (peak learning rate 2.5×10^{-5}, weight decay 1×10^{-4}, gradient clipping at 1.0 max norm), batch size 256, and label smoothing 0.1. The learning rate follows a linear warmup over 25,000 steps then cosine decay to zero (100,080 total steps). Training images are augmented with RandomResizedCrop(512^2 px, scale $\in [0.2, 1.0]$) and horizontal flips. Each step processes $T = 4$ F-IID glimpses with full backpropagation through time. We performed this fine-tuning step on TPU v6e-4 hardware using `torch_xla` and SPMD for multi-chip parallelism.

Evaluation. When evaluating on the ImageNet-1k[37] validation set, we preprocess images by resizing the shortest side to 512 pixels followed by a center crop. Across up to $T = 21$ timesteps, we report frozen CanViT-B IN1k top-1 accuracy by policy and timestep at canvas grid 32×32 on Figure 4**C** (frozen curves) and Table 7. Table 8 reports best-t top-1 at canvas grids $c \times c$ for $c \in \{8, 16, 32, 64\}$. We report fine-tuned CanViT-B IN1k top-1 accuracy by policy and timestep on Figure 4**C** (fine-tuned curves) and Table 9. Canvas resolution has only a marginal effect on best-t top-1 on IN1k (Table 8), unlike on ADE20K (Table 6).

Table 5: **ADE20K mIoU (%) by policy and timestep ($T = 21$), frozen CanViT-B.** Cells: $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy and canvas size per timestep.

Policy	Canvas	t=0	t=1	t=2	t=3	t=4	t=9	t=16	t=20
— EG-C2F†	32^2	38.5	41.1	42.0	42.7	43.2	43.9	44.2	44.1
— EG-C2F†	64^2	**39.6**	**42.2**	**43.3**	**44.1**	**44.7**	**45.6**	**45.8**	45.7
— C2F	32^2	38.5	40.3 ±0.09	41.4 ±0.09	42.3 ±0.05	43.2 ±0.03	43.6 ±0.03	44.0 ±0.05	44.2 ±0.02
— C2F	64^2	**39.6**	41.3 ±0.10	42.5 ±0.08	43.6 ±0.06	**44.7** ±0.03	45.1 ±0.05	45.6 ±0.03	**45.9** ±0.02
— F-IID	32^2	38.5	40.0 ±0.10	40.8 ±0.08	41.3 ±0.09	41.7 ±0.08	42.7 ±0.08	43.2 ±0.07	43.3 ±0.06
— F-IID	64^2	**39.6**	41.2 ±0.10	42.0 ±0.09	42.5 ±0.10	43.0 ±0.09	44.1 ±0.06	44.7 ±0.08	44.9 ±0.07
— R-IID	32^2	18.1 ±0.53	26.0 ±0.34	30.5 ±0.19	33.2 ±0.14	35.1 ±0.16	39.1 ±0.14	41.1 ±0.12	41.6 ±0.09
— R-IID	64^2	18.3 ±0.30	27.1 ±0.37	31.6 ±0.29	34.3 ±0.19	36.3 ±0.16	40.5 ±0.17	42.6 ±0.12	43.1 ±0.13
— RFS†	32^2	38.5	40.0	40.0	39.9	39.8	39.1	38.5	38.2
— RFS†	64^2	**39.6**	41.1	41.1	41.0	40.9	40.3	39.6	39.3
— F2C	32^2	10.8 ±0.14	18.0 ±0.21	22.3 ±0.17	25.5 ±0.13	27.8 ±0.17	34.2 ±0.21	38.6 ±0.13	41.2 ±0.10
— F2C	64^2	11.0 ±0.38	18.1 ±0.32	22.6 ±0.29	25.9 ±0.33	28.3 ±0.29	35.4 ±0.36	40.1 ±0.19	42.9 ±0.15

Table 6: **ADE20K mIoU (%) across canvas grid sizes, frozen CanViT-B.** One probe per canvas grid c. Cells: best-t mIoU mean, with $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy.

Policy	$c = 8$	$c = 16$	$c = 32$	$c = 64$
— EG-C2F†	31.6	39.3	44.2	45.8
— C2F	31.6 ±0.03	39.1 ±0.04	44.2 ±0.02	45.9 ±0.02
— F-IID	30.7 ±0.04	38.1 ±0.09	43.3 ±0.06	44.9 ±0.07
— R-IID	29.5 ±0.10	37.2 ±0.07	41.6 ±0.09	43.1 ±0.13
— RFS†	30.3	36.8	40.0	41.1
— F2C	29.2 ±0.14	36.8 ±0.11	41.2 ±0.10	42.9 ±0.15

D.6 Methodology for Figure 5 analyses

Panels A and B. We evaluate the impact of canvas resolution on ADE20K segmentation performance, notably examining how IoU varies as a function of ground truth mask size. To do so, we collect all (scene, class) pairs from the validation set (i.e., where the ground-truth mask is non-empty). For the mask corresponding to each such pair, we measure its area as the ratio between mask and scene pixels and compute per-mask IoU with the predictions from frozen DINOv3 and frozen CanViT-B models. Unlike the class-level mIoU reported elsewhere, per-mask IoU is com-

Table 7: **ImageNet-1k top-1 accuracy (%) across timesteps, frozen CanViT-B (32×32 canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy at each timestep.

Policy	t=0	t=1	t=2	t=3	t=4	t=5	t=9	t=15	t=20
— C2F	**76.79**	**78.62** ±0.05	**79.58** ±0.02	**80.25** ±0.03	**80.77** ±0.02	**80.85** ±0.02	**81.02** ±0.03	**81.12** ±0.03	81.13 ±0.02
— F-IID	**76.79**	78.50 ±0.04	79.32 ±0.04	79.81 ±0.05	80.13 ±0.04	80.37 ±0.04	80.86 ±0.03	81.09 ±0.03	**81.14** ±0.02
— R-IID	47.73 ±0.10	65.23 ±0.08	72.23 ±0.09	75.50 ±0.06	77.24 ±0.06	78.22 ±0.06	79.71 ±0.06	80.37 ±0.04	80.56 ±0.05
— RFS†	**76.79**	77.85	77.99	78.01	78.01	77.99	77.73	77.40	77.16
— F2C	32.22 ±0.07	49.04 ±0.08	58.57 ±0.07	64.40 ±0.06	68.23 ±0.10	70.81 ±0.10	75.93 ±0.08	78.52 ±0.06	79.78 ±0.05

Table 8: **ImageNet-1k top-1 accuracy (%) across canvas grid sizes, frozen CanViT-B.** Same IN1k probe across columns, evaluated at canvas grid $c \times c$. Cells: best-t top-1 mean, with $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy.

Policy	$c = 8$	$c = 16$	$c = 32$	$c = 64$
— C2F	80.61 ±0.02	81.08 ±0.02	81.14 ±0.03	81.16 ±0.01
— F-IID	80.74 ±0.04	81.06 ±0.03	81.14 ±0.02	81.14 ±0.03
— R-IID	80.25 ±0.03	80.52 ±0.05	80.56 ±0.05	80.58 ±0.04
— RFS†	77.78	77.99	78.01	78.02
— F2C	79.37 ±0.04	79.69 ±0.02	79.78 ±0.05	79.80 ±0.03

puted for each non-zero mask instance rather than by summing pixels across the full validation set, and consequently does not consider false positive predictions from scenes where the class is absent.

We use 128^2 px input images for DINOv3 and CanViT-B models in Figure 5 A–B. To obtain a desired output resolution, we fix CanViT-B's canvas resolution to 8^2, 16^2, 32^2, or 64^2 at inference. DINOv3's output resolution is determined by its input resolution; with its 16^2 px patch size, it produces an 8^2 feature map. Each model-resolution pair under consideration is evaluated with its own probe, yielding four CanViT-B probes (one per canvas resolution) and one DINOv3 probe. This pipeline produces one (scene, class, mask area, timestep, IoU) datapoint per ground-truth mask per model. For CanViT, timesteps range from $t = 0$ to $t = 20$ under an EG-C2F policy. At $t = 0$, Table 10 reports the corresponding passive comparison against DINOv3 ViT-B/16 (our passive teacher) and DINOv3 ViT-S/16.

To compute the trends shown in Figure 5 A–B, we select the first timestep (**A**) or a difference of timesteps (**B**) and apply locally-weighted scatter-plot smoothing to the scatter of (mask area, IoU) datapoints, using the Python package statsmodels[83] with a smoothing fraction of 0.25. We disable iterative-based reweightings (i.e., we set $\mathrm{it} = 0$) to prevent outliers from being artificially down-weighted. The 95% confidence intervals were computed with 1,000 bootstrap resamples.

Panel C. For CanViT-B, we sweep the EG-C2F glimpse budget from $t = 0$ to $t = 20$ at four canvas resolutions (8^2, 16^2, 32^2, 64^2), producing one curve per grid. For DINOv3 ViT-B/16, we sweep input resolution over 128, 144, 160, 192, 256, 384, and 512 px, with one probe per resolution.

Table 9: **ImageNet-1k top-1 accuracy (%) across timesteps, fine-tuned CanViT-B** (32×32 **canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy at each timestep.

Policy	t=0	t=1	t=2	t=3	t=4	t=5	t=9	t=15	t=20
— C2F	**80.59**	**82.25** ±0.04	**83.21** ±0.03	**83.90** ±0.03	**84.35** ±0.03	**84.43** ±0.03	**84.51** ±0.01	**84.53** ±0.01	**84.51** ±0.03
— F-IID	**80.59**	82.24 ±0.03	82.95 ±0.04	83.40 ±0.03	83.70 ±0.03	83.89 ±0.04	84.25 ±0.04	84.42 ±0.03	84.43 ±0.04
— R-IID	50.99 ±0.11	68.90 ±0.07	75.90 ±0.09	79.07 ±0.08	80.66 ±0.06	81.56 ±0.04	82.97 ±0.05	83.50 ±0.05	83.67 ±0.04
— RFS†	**80.59**	81.59	81.81	81.76	81.62	81.53	81.21	80.69	80.43
— F2C	35.77 ±0.10	53.47 ±0.07	63.01 ±0.07	68.78 ±0.08	72.47 ±0.09	74.91 ±0.09	79.38 ±0.07	81.40 ±0.05	82.38 ±0.04

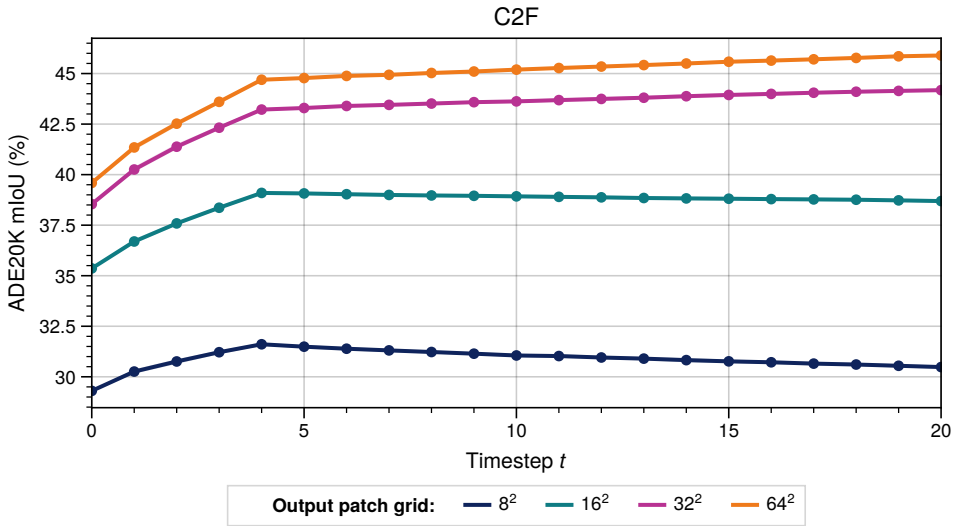

Figure 6: **Impact of canvas grid size on CanViT-B ADE20K mIoU (C2F policy,** $T = 21$**).** Mean mIoU across $n = 10$ evaluation runs, at each timestep, at canvas grids $c \times c$ ($c \in \{8, 16, 32, 64\}$).

D.7 DINOv3 ImageNet-1k linear classification probes

DINOv3[36] was released with a pretrained ImageNet-1k linear classification head only for the 7B-parameter flagship model, not for the smaller ViT checkpoints (ViT-S/16, ViT-S+/16, ViT-B/16, ViT-L/16, ViT-H+/16). Additionally, only ImageNet-ReaL top-1 accuracy, rather than standard ImageNet-1k top-1 validation accuracy, was reported for those non-flagship checkpoints. We trained linear probes for all five smaller DINOv3 ViT models, which all match or exceed the best ImageNet-ReaL top-1 validation accuracy reported by the DINOv3 authors (Table 11).

Feature extraction. We extract CLS tokens from the last transformer layer of each DINOv3 ViT model at 512×512 input resolution (1024 patch tokens with 16×16 px patches), using bfloat16 automatic mixed precision for the forward pass. CLS tokens are stored in float32. Data loading uses Inception-crop augmentation for the training split. Features are pre-computed and cached to disk before any probe training begins. This decouples the expensive backbone forward pass from the probe optimization, which iterates over cached feature vectors rather than running the backbone or loading images. This makes extensive hyperparameter search over the full training set practical. Since augmentation is applied at extraction time and baked into the cached features, each extraction run through the training split (with different random shuffling and augmentation) produces a separate

Table 10: **Passive-vision comparison: ADE20K mIoU at $t = 0$ (single full-scene glimpse).** CanViT-B vs. DINOv3 ViT-B/16 and ViT-S/16 at various input and output resolutions. All models use frozen features with a linear segmentation probe.

Model	Input	Scene grid	GFLOPs	mIoU (%)
DINOv3 ViT-S/16	128 px	8×8	3.1	25.2
	144 px	9×9	3.8	27.1
	160 px	10×10	4.7	29.4
	192 px	12×12	6.8	32.3
	256 px	16×16	12.5	36.9
	384 px	24×24	31.3	40.9
	512 px	32×32	63.9	43.3
DINOv3 ViT-B/16	128 px	8×8	12.0	28.8
	144 px	9×9	15.0	31.0
	160 px	10×10	18.4	33.2
	192 px	12×12	26.3	35.9
	256 px	16×16	47.2	40.6
	384 px	24×24	112.0	44.8
	512 px	32×32	215.3	47.2
CanViT-B (t=0, full scene)	128 px	8×8	13.8	29.3
	128 px	9×9	13.9	29.5
	128 px	10×10	13.9	31.5
	128 px	12×12	14.0	33.4
	128 px	16×16	14.2	35.4
	128 px	24×24	14.9	37.6
	128 px	32×32	15.8	38.5
	128 px	64×64	22.1	39.6

file. To train with N augmentation epochs, N extraction passes must be run and stored. During probe training, each file is treated as an independent epoch.

Hyperparameter (HP) search. For each backbone, we sweep the following HPs with Optuna[84] to maximize ImageNet-ReaL top-1 validation accuracy[85]: loss (softmax or sigmoid cross-entropy); optimizer (AdamW or SGD); reference learning rate (10^{-7} to 10^{-2}, log-uniform); batch size (one of the top three powers of 2 fitting in GPU memory for that backbone); AdamW weight decay (0 to 0.1, uniform), β_1 (0.1 to 0.99, log-uniform), and $\beta_2 = \beta_1 + g(1 - \beta_1)$ with $g \in [0.01, 1.0]$ log-uniform (ensuring $\beta_2 > \beta_1$); SGD momentum (0 to 0.99, uniform); and weight initialization (Gaussian $\mathcal{N}(0, 0.01)$ with zero bias, or PyTorch default). The peak learning rate is the reference learning rate times the batch size (linear scaling rule), with linear warmup over the first 10% of steps followed by cosine annealing to zero. We report selected HPs in Table 12.

Table 11: **DINOv3 IN1k linear probe accuracy at 512×512 input resolution.**

Model	IN-ReaL top-1 (official / ours)	IN1k top-1 (ours)
ViT-S/16	87.0% / **87.08%**	81.40%
ViT-S+/16	88.0% / **88.08%**	82.89%
ViT-B/16	89.3% / **89.54%**	85.00%
ViT-L/16	90.2% / **90.42%**	87.44%
ViT-H+/16	90.3% / **90.31%**	87.65%

Table 12: **Selected hyperparameters for each DINOv3 IN1k linear probe.**

Model	Loss	Batch size	Peak LR	Weight decay	(β_1, β_2)	DINOv3 init	Epochs (aug., total)
ViT-S/16	softmax	2048	6.6×10^{-3}	6.0×10^{-2}	$(0.22, 0.71)$	Yes	10, 20
ViT-S+/16	softmax	1024	1.1×10^{-3}	3.8×10^{-2}	$(0.10, 0.94)$	No	10, 20
ViT-B/16	softmax	1024	2.8×10^{-4}	9.2×10^{-2}	$(0.56, 0.73)$	Yes	10, 20
ViT-L/16	sigmoid	2048	3.0×10^{-3}	3.1×10^{-2}	$(0.40, 0.63)$	Yes	10, 20
ViT-H+/16	sigmoid	1024	8.2×10^{-3}	4.9×10^{-2}	$(0.78, 0.93)$	No	8, 16

E Pretraining ablations

Table 13: **Ablation study: reconstruction quality on ADE20K validation set.** We compare CanViT's predicted DINOv3 ViT-B features against reference teacher features after 10 R-IID glimpses. Params: the model alone, without readouts or probe. GFLOPs: per-timestep forward pass cost, including the ADE20K probe. Δ: % change in cosine similarity relative to baseline (mean over $n = 5$ evaluation runs per variant). Sp. = spatial (dense, patch-level); CLS = CLS-token.

Variant	Params	GFLOPs	Δ Sp. Cos	Δ CLS Cos
CanViT-B (baseline)	95.2M	15.8	ref	ref
a $D_{\text{can}} = 256$, asymmetric ($n_h = 2$)	88.1M	13.2	-12.0%	-2.1%
b $D_{\text{can}} = 256$, + QKVO ($n_h = 2$)	88.8M	14.8	-11.7%	-1.6%
c $D_{\text{can}} = 384$, + QKVO ($n_h = 3$)	91.0M	17.3	-5.8%	-1.0%
d No canvas reads	90.4M	14.2	-6.5%	-7.9%
e RW stride = 6 (1R / 1W)	88.8M	13.7	-3.9%	-6.0%
f No dense supervision	95.2M	15.8	-98.7%	-8.9%
g No F-IID, $1\times$ R-IID	95.2M	15.8	-9.4%	-16.2%
h No F-IID, $2\times$ R-IID	95.2M	15.8	-5.3%	-9.4%
i No BPTT ($K = 1$)	95.2M	15.8	-3.8%	-7.5%
j $D_{\text{bb}} = 384$ (ViT-S)	26.4M	5.9	-8.4%	-21.1%
k No VPE token	95.2M	15.6	-0.1%	-0.3%

To assess the influence of our design choices at the level of architecture and pretraining, we conducted an ablation study using short pretraining runs. For our ablation baseline and each ablated variant, we allocated approximately 215k optimizer steps (110M glimpses for the baseline, slightly over 10% of our flagship's pretraining compute), with a learning rate warmup of 20k steps.

We report training loss curves on Figure 7, disaggregated by reconstruction target (patch/CLS) and by policy (R-IID/F-IID). As our chosen step count results in slightly more than 1 epoch on the 13.2 million ImageNet-21k images in our pretraining dataset, these training curves are representative of the model's generalization capabilities.

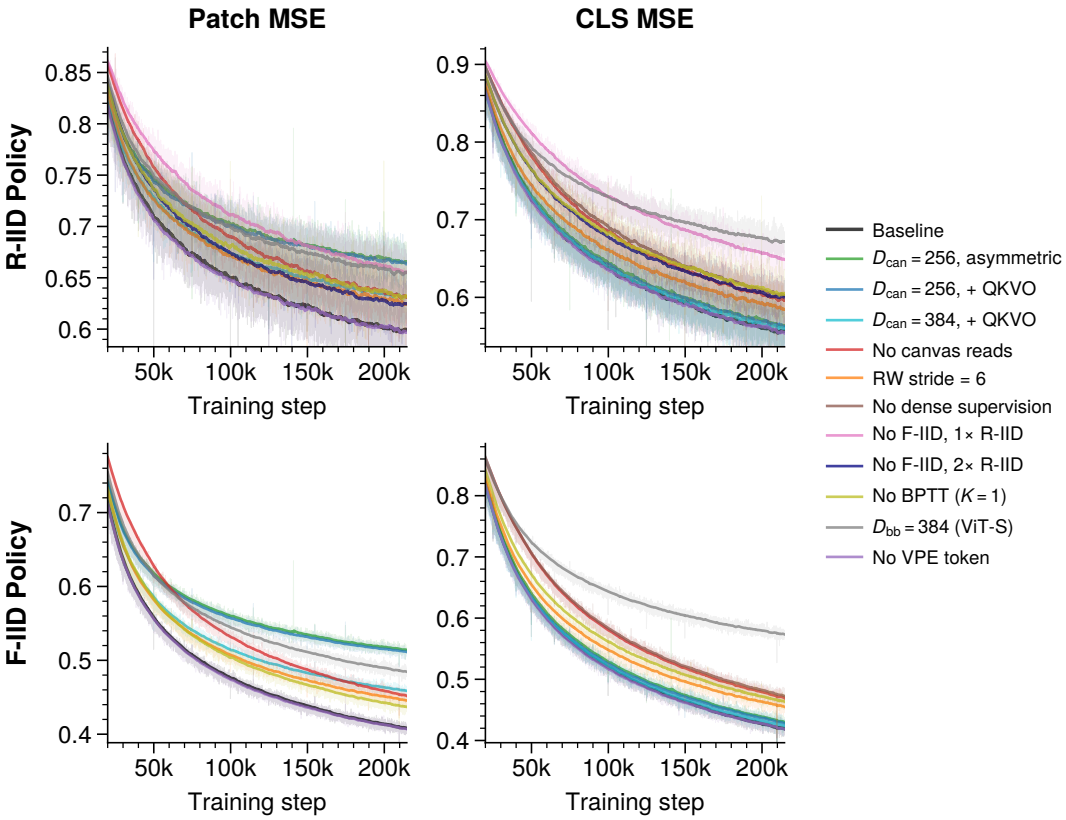

Figure 7: **Ablation study: loss curves during ImageNet-21k pretraining.** Faint lines: logged per-batch losses. Bold lines: exponential moving average of logged losses.

To further quantify generalization, we evaluated each ablation variant on the ADE20K validation set by measuring cosine similarity between the canvas reconstruction and DINOv3 ViT-B teacher features under the R-IID policy. We report the relative change compared to baseline on Table 13, alongside computational footprint and parameter count, with 95% CIs and $t = 0$ results in Table 14. To relate reconstruction quality to downstream accuracy, we also evaluated the same frozen checkpoints on ADE20K and ImageNet-1k (Table 15). For ADE20K, we trained one probe per checkpoint as in Appendix D.4. For ImageNet-1k, we fused each checkpoint's CLS projection with our DINOv3 ViT-B probe as in Appendix D.5, without further training.

Capacity–expressiveness trade-offs. The removal of canvas-side QKVO projections is key to the low overhead of Canvas Attention. On any given training or inference budget, this allows for more frequent canvas–backbone interactions, a larger canvas embedding dimension (semantic resolution), and the use of more canvas patches (spatial resolution). However, at fixed canvas dimensionality, the ablation of these projections reduces the expressiveness of each individual cross-attention operation. To assess the well-foundedness of this trade-off, we reduced canvas dimensionality from $D_{\text{can}} = 1024$ to $D_{\text{can}} = 256$, leading to a dramatic drop in patch reconstruction quality (-12.0%, Table 13 **a**). Re-introducing canvas-side QKVO projections in a FLOP-matched manner forces the use of a small canvas, resulting in a loss of per-position information capacity and a failure to rescue reconstruction quality via increased expressiveness (-11.7% and -5.8% respectively, Table 13 **b,c**).

Frequency and directionality of canvas–backbone interaction. CanViT interleaves Canvas Attention Read/Write operations along depth. In CanViT-B, this corresponds to 3 reads and 3 writes per glimpse (R/W stride of 2), evenly spread across its ViT-B backbone's 12 Transformer blocks. Write operations are required in order to update the canvas and produce dense outputs. In contrast, Read operations can be readily ablated; doing so results in a large drop in both patch-level (-6.5%) and CLS-level (-7.9%) reconstruction quality (Table 13 **d**). This highlights the benefit of canvas-to-backbone communication, which underpins top-down recurrent feedback across timesteps, indirect canvas-to-canvas interaction via the backbone, and generally allows backbone-side computation to

Table 14: **Ablation study: reconstruction quality on ADE20K validation set (supplementary).** Cosine similarity (%) with DINOv3 ViT-B teacher features; cells are 95% bootstrap CIs across $n = 5$ independent R-IID runs per variant. Sp. = spatial (patch-level), CLS = CLS-token.

Variant	$t = 0$		$t = 9$	
	Sp.	CLS	Sp.	CLS
CanViT-B (baseline)	46.9–47.4	46.4–47.2	71.3–71.4	69.5–69.7
a $D_{\text{can}} = 256$, asymmetric	42.5–42.8	45.6–46.2	62.7–62.8	68.1–68.2
b $D_{\text{can}} = 256$, + QKVO	42.9–43.2	46.4–46.6	63.0–63.1	68.4–68.6
c $D_{\text{can}} = 384$, + QKVO	44.5–45.0	45.5–46.3	67.1–67.3	68.8–69.0
d No canvas reads	45.1–45.7	44.1–44.7	66.6–66.7	64.0–64.2
e RW stride = 6	45.2–45.3	44.3–44.5	68.5–68.5	65.3–65.5
f No dense supervision	0.9–0.9	42.9–43.6	0.9–0.9	63.3–63.5
g No F-IID, 1× R-IID	43.7–44.0	40.1–40.5	64.6–64.7	58.2–58.3
h No F-IID, 2× R-IID	45.4–46.1	43.0–43.9	67.5–67.6	63.0–63.1
i No BPTT ($K = 1$)	45.9–46.3	45.1–45.3	68.5–68.7	64.4–64.5
j $D_{\text{bb}} = 384$ (ViT-S)	42.9–43.3	37.6–38.0	65.3–65.4	54.9–55.0
k No VPE token	46.4–47.3	45.9–46.8	71.2–71.3	69.4–69.5

Table 15: **Ablation study: ADE20K mIoU (%) and ImageNet-1k top-1 accuracy (%) of the frozen ablation checkpoints (**32×32 **canvas).** ADE20K: EG-C2F policy. ImageNet-1k: C2F policy, with $\pm$ 95% bootstrap CI half-width ($n = 10$ runs).

Variant	**ADE20K mIoU**		**IN1k top-1**	
	$t = 0$	$t = 20$	$t = 0$	$t = 20$
CanViT-B (baseline)	29.2	37.9	61.31	69.16 $\pm$ 0.04
a $D_{\text{can}} = 256$, asymmetric	26.3	34.1	60.03	66.97 $\pm$ 0.03
b $D_{\text{can}} = 256$, + QKVO	26.8	34.8	60.14	67.38 $\pm$ 0.03
c $D_{\text{can}} = 384$, + QKVO	27.9	36.5	61.08	67.94 $\pm$ 0.04
d No canvas reads	25.3	31.9	55.94	60.00 $\pm$ 0.03
e RW stride = 6	26.8	33.8	57.10	62.60 $\pm$ 0.04
f No dense supervision	12.3	17.4	54.29	59.85 $\pm$ 0.05
g No F-IID, 1× R-IID	18.8	29.6	39.18	50.05 $\pm$ 0.04
h No F-IID, 2× R-IID	22.2	32.3	47.80	58.97 $\pm$ 0.03
i No BPTT ($K = 1$)	27.6	34.2	57.04	61.59 $\pm$ 0.03
j $D_{\text{bb}} = 384$ (ViT-S)	24.4	32.4	44.31	52.15 $\pm$ 0.02
k No VPE token	29.3	37.0	61.39	68.71 $\pm$ 0.02

benefit from the high-capacity workspace constituted by the canvas. When increasing the R/W stride from 2 to 6 Transformer blocks, which results in just 1 read and 1 write per glimpse, we observe a similar yet slightly less pronounced effect (-3.9%, Table 13 **e**). Together, these results show the benefits of frequent, bidirectional Canvas Attention operations, and point at the importance of within-glimpse canvas refinement and contextually-aware backbone computation.

Dense latent supervision. Omitting dense supervision is contrary to the goal of using a frozen CanViT's canvas features for dense tasks. However, this objective-level intervention has no effect on the model's architecture or raw expressiveness, and could theoretically enhance CLS reconstruction by allowing the model's representations to be specialized for this purpose, rather than requiring them to support both CLS-level and patch-level reconstruction. In our ablation study, the reverse was true

(-8.9%, Table 13 **f**): removing patch-level supervision *degrades* CLS reconstruction, indicating that the patch-level objective also benefits CLS-level learning rather than competing with it for capacity.

F-IID rollouts. During pretraining, we average losses and gradients across two rollouts that start from a random (R-IID) or full-scene zoomed-out (F-IID) viewpoint for each scene. The inclusion of an F-IID rollout ensures that at least one glimpse has full spatial coverage and that the full-scene viewpoint, $(x = 0, y = 0, s = 1)$, can be seen during training. Simply removing the F-IID rollout (-9.4% spatial, Table 13 **g**) dramatically slows down the decrease *of the R-IID loss*; however, this also halves the total number of glimpses per optimizer step. To control for this, we trained a second variant that replaces the F-IID rollout with a second R-IID rollout, preserving the total number of glimpses per step (Table 13 **h**). The controlled variant still incurs a substantial degradation (-5.3% spatial, -9.4% CLS), confirming that the full-scene viewpoint itself is beneficial, beyond its contribution as an additional rollout.

Temporal credit assignment. CanViT uses the smallest possible truncated BPTT chunk size, $K = 2$, in order for gradient updates to take into account a glimpse and its successor. Setting $K = 1$ roughly halves the backward-pass memory footprint, but eliminates gradient flow across time altogether. Given temporally-dense supervision and highly-informative canvas tokens, it may still be possible to obtain meaningful results with $K = 1$, as the model learns to greedily produce a best-guess reconstruction at each timestep, which incidentally produces an informative canvas for the next timestep to reuse. In this ablation, we also decrease the stop probability from 0.5 to 0.25 to keep the expected number of glimpses per optimizer step comparable. We find that removing BPTT (Table 13 **i**) degrades both spatial (-3.8%) and CLS (-7.5%) reconstruction, indicating that even minimal temporal gradient flow ($K = 2$) contributes meaningfully to learning.

Backbone embedding dimension. Reducing the backbone embedding dimension from $D_{\text{bb}} = 768$ to $D_{\text{bb}} = 384$ is exactly equivalent to using a ViT-S backbone, rather than ViT-B. This results in the largest drop in parameter count, per-glimpse computational footprint, and CLS reconstruction quality (-21.1%) across all ablations. However, the impact of a narrower backbone on patch (spatial) reconstruction (-8.4%), while significant, remains lower than that of several other ablations.

VPE token. Among all considered ablations, the removal of the VPE token had the lowest impact across both policies and both loss types (-0.1% spatial, -0.3% CLS; Figure 7, Table 13 **k**). However, the downstream results of the same checkpoints (Table 15 and Appendix H.2) show that VPE is beneficial at long horizons and high resolution.

F Inference latency

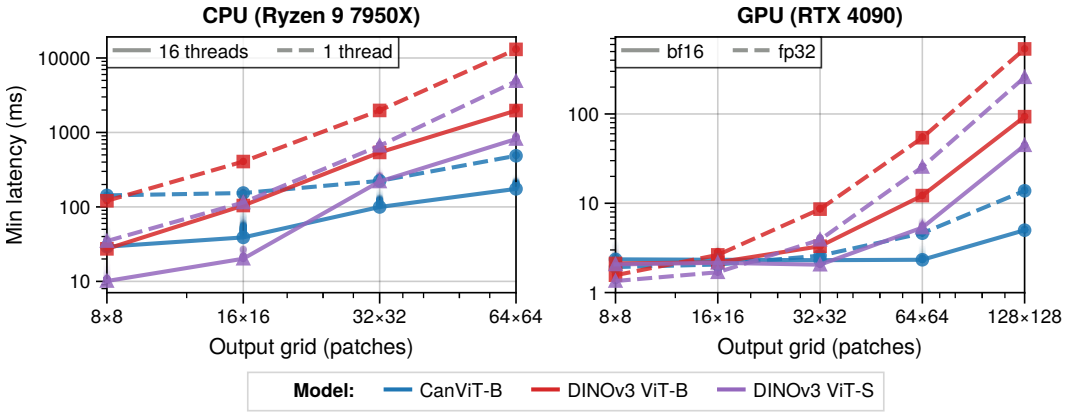

Figure 8: **Latency scaling with output resolution.** Faint scatter: individual iterations.

We report minimum latency and peak memory scaling in Figure 8 and Table 16. Benchmarks run on an NVIDIA GeForce RTX 4090 and on an AMD Ryzen 9 7950X 16-Core Processor. CUDA benchmarks use both float32 and AMP bfloat16, and `torch.compile`. CPU benchmarks use float32 in eager mode, with 1 thread or 16 threads (the physical-core count).

Table 16: **Latency and peak memory by hardware, precision, and output resolution.** **Bold** = lowest latency and peak memory per row.

Device	Scene grid	CanViT-B		DINOv3 ViT-B/16		DINOv3 ViT-S/16	
		ms	MB	ms	MB	ms	MB
CPU, 1T	8×8	143	—	121	—	**34.8**	—
CPU, 1T	16×16	154	—	407	—	**115**	—
CPU, 1T	32×32	**224**	—	1975	—	663	—
CPU, 1T	64×64	**486**	—	13049	—	4882	—
CPU, 16T	8×8	29.3	—	27.4	—	**10.1**	—
CPU, 16T	16×16	39.0	—	104	—	**20.2**	—
CPU, 16T	32×32	**99.8**	—	539	—	219	—
CPU, 16T	64×64	**175**	—	1969	—	817	—
CUDA, AMP bf16	8×8	2.36	397	2.12	359	**2.07**	**100**
CUDA, AMP bf16	16×16	2.33	402	2.18	362	**2.16**	**101**
CUDA, AMP bf16	32×32	2.31	419	3.30	388	**2.05**	**115**
CUDA, AMP bf16	64×64	**2.33**	499	12.2	416	5.34	**135**
CUDA, AMP bf16	128×128	**4.99**	794	93.3	607	44.6	**250**
CUDA, fp32	8×8	1.93	395	1.57	356	**1.35**	**99**
CUDA, fp32	16×16	2.06	408	2.64	359	**1.69**	**101**
CUDA, fp32	32×32	**2.59**	430	8.60	378	3.90	**112**
CUDA, fp32	64×64	**4.62**	544	54.3	454	25.5	**154**
CUDA, fp32	128×128	**13.8**	996	535	758	257	**326**

Each timed iteration processes a single input (batch size $B = 1$) with explicit device synchronization both before and after. After 3 warmup iterations, measurement runs at least 5 iterations and continues until either a 20-second budget or a 500-iteration limit is reached, whichever comes first. We repeat each configuration 3 times and pool the per-iteration latencies. Within the measurement window, we record peak GPU memory and minimum forward-pass latency. To vary the scene grid size during benchmarking, we adjust the canvas resolution for CanViT and the input (and thus, output) resolution for DINOv3, as in Figure 5 and Appendix D.6.

G Interpretability

PCA visualizations. To visualize grids of high-dimensional glimpse or canvas patch tokens as RGB images (Figure 1, Figure 2, Figure 3, Figure 9), we adopt a similar approach to that of DINOv3[36], by performing Principal Component Analysis (PCA) across tokens then mapping groups of three consecutive PCs to RGB. In order to prevent global variance from washing out local detail when visualizing a subset of the canvas, we apply min-max scaling to the resulting RGB channels across the visible region. We apply this protocol to layer-normalized[64] tokens, such that each individual token has unit variance and zero mean across its dimensions.

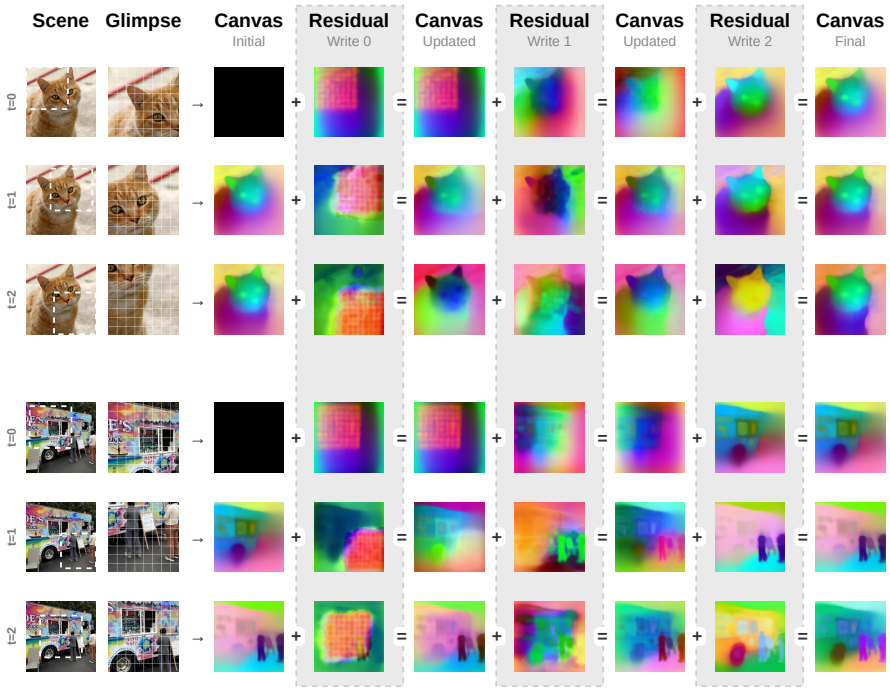

Figure 9: **Canvas updates within and across glimpses.** CanViT performs multiple Canvas Attention Write operations per glimpse, each producing a residual that is summed with the canvas (Figure 2). To isolate the contribution of individual Write operations, we capture intermediate residuals and canvases after each Write operation within a 128^2 px glimpse, and visualize these snapshots with Principal Component Analysis (PCA) across tokens. We compute PCA bases independently for all snapshots rather than using a shared basis, in order to visualize the spatial structure of each individual snapshot. We observe an emergent progression in the spatial structure of residuals as within-glimpse processing unfolds from Write 0 to Write 2, from localized and feature-level content to scene-wide and semantically grounded content. Write-0 residuals reflect the structure of the glimpse's 8×8 patch grid. Write-1 residuals begin to reflect the structure of the objects within the glimpse. Write-2 residuals extrapolate beyond the confines of the glimpse, and delineate sharper object boundaries.

H Additional experiments

We ran additional experiments during the review and rebuttal period. Most rebuttal-period experiments were run on Google Cloud using Spot and DWS Flex TPU v5p and TPU v6e instances, with code written in JAX[62] and Flax NNX[63].

H.1 Impact of the pretraining dataset

We pretrained our flagship on ImageNet-21k (IN21k) to maximize the quality of CanViT-B's representations through scene diversity, rather than by relying on a complex augmentation pipeline on top of the smaller IN1k. To measure the effect of this choice, we pretrained a new CanViT-B checkpoint on IN1k instead of IN21k, using the same compute budget and hyperparameters as in our flagship results. We found that, unlike IN21k pretraining, IN1k-only pretraining exhibited late instability, and thus retained the checkpoint that best reconstructed the teacher's features on ImageNet-1k validation images before the instability, at 92% of training (step 1,837,056 of 2,000,000). We then fit ADE20K probes to this new pretrained checkpoint. We computed IN1k classification performance using frozen features, as well as via LP-FT fine-tuning. For LP-FT, we fine-tuned both checkpoints anew in JAX, with the fine-tuning hyperparameters of Appendix D.5.

Here, we observe that pretraining on IN1k instead of IN21k results in a minor drop in LP-FT top-1 IN1k classification accuracy, a slight *increase* in frozen-probe IN1k accuracy, and a drop in ADE20K

Table 17: **ADE20K mIoU (%) and ImageNet-1k top-1 accuracy (%) by pretraining dataset, CanViT-B (C2F policy, $t = 20$).** ADE20K with a 64×64 canvas, IN1k with a 32×32 canvas. Cells: $\pm$ 95% bootstrap CI half-width ($n \geq 10$ runs).

	ADE20K mIoU, frozen	**IN1k top-1, frozen**	**IN1k top-1, LP-FT**
CanViT-B, IN21k-pretrained	45.89 $\pm$ 0.02	81.13 $\pm$ 0.02	84.44 $\pm$ 0.02
CanViT-B, IN1k-pretrained	43.40 $\pm$ 0.03	81.49 $\pm$ 0.02	84.19 $\pm$ 0.02

segmentation accuracy. These opposite effects may be due to a potential in-domain advantage when pretraining on IN1k (train) then evaluating on IN1k (val), which vanishes given cross-dataset transfer (when evaluating on ADE20K) or given the increased risk of overfitting presented by fine-tuning (when using LP-FT on IN1k). Regardless, with IN1k pretraining alone and C2F evaluation, CanViT-B still reaches 84.2% top-1 IN1k accuracy after LP-FT and 43.4% ADE20K mIoU, ahead of every prior model in Table 1. Tables 18, 19 and 20 report each policy's results for both checkpoints.

Table 18: **ADE20K mIoU (%) by policy and timestep, frozen CanViT-B pretrained on IN21k or IN1k (64×64 canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy at each timestep, for each checkpoint.

Policy	t=0	t=1	t=2	t=3	t=4	t=9	t=16	t=20
CanViT-B, IN21k-pretrained								
— EG-C2F†	**39.6**	**42.2**	**43.3**	**44.1**	**44.7**	**45.6**	**45.8**	45.7
— C2F	**39.6**	41.3 ±0.10	42.5 ±0.08	43.6 ±0.06	**44.7** ±0.03	45.1 ±0.05	45.6 ±0.03	**45.9** ±0.02
— F-IID	**39.6**	41.2 ±0.10	42.0 ±0.09	42.5 ±0.10	43.0 ±0.09	44.1 ±0.06	44.7 ±0.08	44.9 ±0.07
— R-IID	18.3 ±0.30	27.1 ±0.37	31.6 ±0.29	34.3 ±0.19	36.3 ±0.16	40.5 ±0.17	42.6 ±0.12	43.1 ±0.13
— RFS†	**39.6**	41.1	41.1	41.0	40.9	40.3	39.6	39.3
— F2C	11.0 ±0.38	18.1 ±0.32	22.6 ±0.29	25.9 ±0.33	28.3 ±0.29	35.4 ±0.36	40.1 ±0.19	42.9 ±0.15
CanViT-B, IN1k-pretrained								
— EG-C2F†	**36.4**	**38.7**	**40.2**	**41.0**	**41.7**	**43.0**	**43.3**	43.3
— C2F	**36.4**	38.1 ±0.03	39.4 ±0.07	40.7 ±0.04	**41.7** ±0.05	42.4 ±0.05	43.1 ±0.04	**43.4** ±0.03
— F-IID	**36.4**	37.9 ±0.03	38.7 ±0.06	39.3 ±0.08	39.8 ±0.07	41.2 ±0.07	42.0 ±0.07	42.2 ±0.07
— R-IID	16.9 ±0.30	25.1 ±0.27	29.6 ±0.24	32.5 ±0.30	34.3 ±0.31	38.5 ±0.23	40.5 ±0.26	41.1 ±0.25
— RFS†	**36.4**	37.5	37.6	37.5	37.4	36.8	36.2	36.0
— F2C	9.4 ±0.16	16.1 ±0.17	20.7 ±0.34	24.0 ±0.26	26.5 ±0.22	33.5 ±0.14	38.2 ±0.13	40.9 ±0.14

Table 19: **ImageNet-1k top-1 accuracy (%) by policy and timestep, frozen CanViT-B pretrained on IN21k or IN1k** (32×32 **canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 10$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy at each timestep, for each checkpoint.

Policy	t=0	t=1	t=2	t=3	t=4	t=5	t=9	t=15	t=20
CanViT-B, IN21k-pretrained									
— C2F	**76.79**	**78.62** ±0.05	**79.58** ±0.02	**80.25** ±0.03	**80.77** ±0.02	**80.85** ±0.02	**81.02** ±0.03	**81.12** ±0.03	81.13 ±0.02
— F-IID	**76.79**	78.50 ±0.04	79.32 ±0.04	79.81 ±0.05	80.13 ±0.04	80.37 ±0.04	80.86 ±0.03	81.09 ±0.03	**81.14** ±0.02
— R-IID	47.73 ±0.10	65.23 ±0.08	72.23 ±0.09	75.50 ±0.06	77.24 ±0.06	78.22 ±0.06	79.71 ±0.06	80.37 ±0.04	80.56 ±0.05
— RFS†	**76.79**	77.85	77.99	78.01	78.01	77.99	77.73	77.40	77.16
— F2C	32.22 ±0.07	49.04 ±0.08	58.57 ±0.07	64.40 ±0.06	68.23 ±0.10	70.81 ±0.10	75.93 ±0.08	78.52 ±0.06	79.78 ±0.05
CanViT-B, IN1k-pretrained									
— C2F	**77.75**	**79.01** ±0.02	**79.74** ±0.04	**80.28** ±0.03	**80.68** ±0.02	**80.82** ±0.02	**81.15** ±0.01	**81.39** ±0.02	**81.49** ±0.02
— F-IID	**77.75**	78.99 ±0.03	79.62 ±0.03	80.04 ±0.02	80.31 ±0.04	80.53 ±0.03	80.98 ±0.05	81.27 ±0.04	81.36 ±0.03
— R-IID	51.66 ±0.16	68.03 ±0.12	74.19 ±0.10	76.94 ±0.08	78.32 ±0.07	79.10 ±0.06	80.36 ±0.04	80.90 ±0.03	81.06 ±0.04
— RFS†	**77.75**	78.46	78.71	78.76	78.79	78.76	78.56	78.23	78.08
— F2C	37.84 ±0.10	54.25 ±0.06	62.93 ±0.09	67.99 ±0.07	71.26 ±0.07	73.43 ±0.06	77.50 ±0.05	79.44 ±0.04	80.38 ±0.03

Table 20: **ImageNet-1k top-1 accuracy (%) by policy and timestep, CanViT-B fine-tuned (LP-FT) from the IN21k or IN1k checkpoint** (32×32 **canvas).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 11$ runs) for stochastic policies. †: deterministic policy. **Bold** = best policy at each timestep, for each checkpoint.

Policy	t=0	t=1	t=2	t=3	t=4	t=5	t=9	t=15	t=20
CanViT-B, IN21k-pretrained									
— C2F	**80.54**	**82.27** ±0.02	**83.19** ±0.03	**83.78** ±0.03	**84.24** ±0.03	**84.31** ±0.03	**84.39** ±0.02	**84.44** ±0.02	**84.44** ±0.02
— F-IID	**80.54**	82.18 ±0.02	82.87 ±0.03	83.29 ±0.03	83.57 ±0.02	83.77 ±0.02	84.19 ±0.03	84.34 ±0.03	84.37 ±0.02
— R-IID	50.99 ±0.08	68.85 ±0.07	75.83 ±0.08	78.97 ±0.06	80.62 ±0.05	81.53 ±0.07	82.91 ±0.05	83.41 ±0.05	83.56 ±0.05
— RFS†	**80.54**	81.72	81.83	81.74	81.61	81.45	81.05	80.67	80.39
— F2C	35.70 ±0.10	53.27 ±0.10	62.89 ±0.11	68.62 ±0.09	72.22 ±0.07	74.70 ±0.08	79.26 ±0.07	81.29 ±0.06	82.28 ±0.04
CanViT-B, IN1k-pretrained									
— C2F	**79.93**	**81.75** ±0.03	**82.66** ±0.02	**83.29** ±0.02	**83.80** ±0.02	**83.90** ±0.02	**84.07** ±0.02	**84.19** ±0.02	**84.19** ±0.02
— F-IID	**79.93**	81.67 ±0.03	82.42 ±0.05	82.88 ±0.04	83.18 ±0.04	83.37 ±0.04	83.83 ±0.04	84.06 ±0.04	84.11 ±0.03
— R-IID	52.92 ±0.12	69.74 ±0.11	76.13 ±0.09	79.01 ±0.07	80.49 ±0.05	81.26 ±0.05	82.53 ±0.03	83.06 ±0.03	83.23 ±0.03
— RFS†	**79.93**	81.06	81.22	81.19	81.07	80.92	80.59	80.19	80.05
— F2C	38.61 ±0.09	55.47 ±0.11	64.37 ±0.08	69.61 ±0.06	72.97 ±0.05	75.23 ±0.05	79.18 ±0.03	80.93 ±0.05	81.93 ±0.05

H.2 Viewpoint Encoding at long horizons and high resolution

We hypothesized that the VPE token's additional location information could help in long-horizon and high-resolution settings, where integration of fine detail matters most. To test this, we evaluated the VPE and no-VPE ablation checkpoints (Table 13, baseline and row **k**) with the EG-C2F policy, using ADE20K probes for 32×32 (C32) and 64×64 (C64) canvases, on 512^2 and 1024^2 px scenes respectively. We report the results in Table 21, which show that VPE is beneficial at long horizons and high resolution.

Table 21: **Ablation study: ADE20K mIoU (%) with and without the VPE token, EG-C2F policy.** C32: 32×32 canvas, 512^2 px scenes. C64: 64×64 canvas, 1024^2 px scenes.

	t=0	t=3	t=10	t=15	t=20
C32 (VPE)	29.24	35.63	37.62	37.88	37.89
C32 (no VPE)	29.34	35.47	37.17	37.20	37.03
C32 (VPE) − C32 (no VPE)	**−0.10**	**+0.17**	**+0.45**	**+0.69**	**+0.86**
C64 (VPE)	29.89	36.71	38.76	39.04	39.00
C64 (no VPE)	29.59	35.80	37.75	37.95	37.72
C64 (VPE) − C64 (no VPE)	**+0.30**	**+0.91**	**+1.01**	**+1.09**	**+1.28**

H.3 ImageNet-1k accuracy against inference cost

While our core efficiency claims are made within the scope of dense scene-wide prediction, our IN1k results can be further contextualized in two respects: on accuracy against inference cost, and against GFNet[20] in addition to AdaptiveNN[26] (Table 22). We use the "anytime prediction" inference setting, which fixes the number of glimpses per image. We recomputed every AdaptiveNN and GFNet cell from their released code and checkpoints, ensuring that all three models are measured the same way. All FLOP counts follow the convention of two FLOPs per MAC, and include policy networks and classification heads. For AdaptiveNN and GFNet, the multiple-glimpse rows use the largest number of glimpses their released models take. Parameter counts include every module that runs up to the evaluated glimpse. AdaptiveNN and GFNet run a policy network and a second encoder after the first glimpse, and a classifier at every glimpse, so their parameter counts grow with the number of glimpses. For the 8^2-canvas LP-FT rows, we fine-tuned the pretrained flagship with an 8^2 canvas.

Table 22: **ImageNet-1k top-1 accuracy (%), parameters and cumulative inference GFLOPs of CanViT-B (C2F policy), AdaptiveNN and GFNet at a fixed number of glimpses.** Cells: $\pm$ 95% bootstrap CI half-width over evaluation runs ($n = 10$ runs, or 11 for LP-FT at canvas 8^2).

Model	Input	Glimpses	Params (M)	GFLOPs	Top-1
Single glimpse					
CanViT-B, LP-FT (canvas 32^2)	128^2 px	1	95.9	15.48	80.59
CanViT-B, LP-FT (canvas 8^2)	128^2 px	1	95.9	13.80	80.54
CanViT-B, frozen (canvas 32^2)	128^2 px	1	95.9	15.48	76.79
CanViT-B, frozen (canvas 8^2)	128^2 px	1	95.9	13.80	76.73
AdaptiveNN (DeiT-S)	112^2 px	1	22.0	2.15	73.96
GFNet (EfficientNet-B3)	144^2 px	1	12.2	0.86	78.04
GFNet (MobileNet-V3-L 1.00)	128^2 px	1	5.5	0.15	70.72
Multiple glimpses					
CanViT-B, LP-FT (canvas 32^2)	128^2 px	3	95.9	46.44	83.21 ± 0.03
CanViT-B, LP-FT (canvas 8^2)	128^2 px	3	95.9	41.41	83.09 ± 0.04
CanViT-B, frozen (canvas 32^2)	128^2 px	3	95.9	46.44	79.58 ± 0.02
CanViT-B, frozen (canvas 8^2)	128^2 px	3	95.9	41.41	79.29 ± 0.04
AdaptiveNN (DeiT-S)	112^2 px	5	82.4	14.48	82.29
GFNet (EfficientNet-B3)	144^2 px	4	37.4	3.49	79.92
GFNet (MobileNet-V3-L 1.00)	128^2 px	3	16.6	0.45	73.67

H.4 Architecture control

To compare our architecture with AdaGlimpse's under the same DINOv3 teacher and data, we ran a small-scale controlled experiment, pitting an AdaGlimpse baseline (112.3M parameters, including its Transformer decoder to DINOv3 space) against CanViT-B (96.5M parameters, including its linear projections to DINOv3 space), with the same glimpse budget as our ablation baseline (approximately 110M glimpses; Appendix E), from a random initialization. We trained both models with the targets and loss of Section 5.1. As we controlled for IN1k-vs-IN21k effects above, we used IN1k images here. We used 512 scenes per step, one F-IID and one R-IID rollout per scene, with dense DINOv3 supervision across $T = 4$ timesteps. We implemented both models in JAX, porting AdaGlimpse's architecture from its PyTorch release, and trained them for 26,875 optimizer steps. Following AdaGlimpse's released code, the baseline divides each glimpse into 2×2 crops and downsamples each crop to a single patch, representing a glimpse with 4 tokens. CanViT-B instead splits the glimpse into 8×8 patches of 16^2 px, representing it with 64 tokens. We ran this experiment on TPU v6e chips, with a pretraining cost of 22 v6e chip-hours for CanViT-B and 212 v6e chip-hours for the AdaGlimpse baseline. We fit one ADE20K linear probe to each model's frozen features. We trained these probes over $T = 4$ timesteps, each model's training horizon. We report the results in Table 23 with an EG-C2F policy and a 32×32 output feature map / canvas resolution.

Here, CanViT-B is ahead at every glimpse count, by 18.1 mIoU at $t = 3$. It reaches its fourth glimpse for less compute than the baseline spends on its first, while requiring considerably less training compute under the same harness. This highlights our architecture's efficiency, both at training and at inference; CanViT's ability to inexpensively output scene-wide dense features allows it to reallocate a given budget to *more* glimpses and *higher-resolution* glimpses, resulting in performance improvements across the board.

Table 23: **Architecture control: ADE20K mIoU (%) against cumulative forward inference FLOPs.** EG-C2F policy, 32×32 canvas / output feature map. GFLOPs include each model's ADE20K probe.

t	mIoU, CanViT-B	mIoU, AdaGlimpse	GFLOPs, CanViT-B	GFLOPs, AdaGlimpse
0	26.86	12.00	15.8	70.2
1	30.52	13.73	31.6	141.3
2	32.77	14.88	47.4	213.5
3	33.69	15.61	63.2	286.8

H.5 COCO object detection

While object detection is a challenging and interesting setting, we did not pursue it in the main experiments, as none of our baselines report results on standard detection benchmarks. Building on the architecture control above (AdaGlimpse vs. CanViT under the same teacher distillation training scheme), we ran a preliminary object detection experiment on COCO[65]. We fine-tuned the architecture-control CanViT-B and AdaGlimpse checkpoints with an identical CenterNet[86] detection head, for approximately 35 epochs, with $T = 5$ glimpses per scene. We also fine-tuned the flagship checkpoint for 140 epochs, with a longer training, BPTT and evaluation horizon ($T = 10$ glimpses in training, evaluation up to $t = 20$). Table 24 lists the training settings.

Table 24: **COCO detection training settings** of the two architecture-control models and of the flagship detector.

Hyperparameter	**Architecture control**	**Flagship**
Peak learning rate, detection head	1×10^{-3}	4×10^{-4}
Peak learning rate, pretrained model	1×10^{-4}	4×10^{-4}
Weight decay	1×10^{-3}	1×10^{-1}
Batch size	256	1,024
Glimpses per scene	$T = 5$	$T = 10$
Optimizer steps at evaluation	16,000 of 92,411	16,172
Optimizer	AdamW, gradient norm clipped at 1.0	
Schedule	Warmup (20% of steps) $\rightarrow$ cosine decay to zero	
Training rollouts	1 F-IID + 1 R-IID, min. scale 0.25, BPTT through all glimpses	
Head	3×3 conv (256 channels); 1×1 convs for heatmap, size, offset	
Loss	focal ($\alpha = 2$, $\beta = 4$) + $0.4\times$ size L1 + offset L1	
Augmentation	horizontal flip, photometric distortion	

Table 25: **COCO val2017 AP by timestep, C2F policy (32×32 canvas / output feature map, 512^2 px scenes).** Cells: $\pm$ 95% bootstrap CI half-width ($n = 5$ runs). —: not evaluated.

Model (initialized from)	t=0	t=1	t=2	t=3	t=4	t=9	t=20
AdaGlimpse (architecture control)	5.55	6.37 $\pm$0.040	7.02 $\pm$0.034	7.59 $\pm$0.023	8.12 $\pm$0.006	—	—
CanViT-B (architecture control)	12.01	14.41 $\pm$0.091	16.16 $\pm$0.059	17.67 $\pm$0.059	19.02 $\pm$0.025	—	—
CanViT-B (flagship)	19.31	22.08 $\pm$0.039	24.37 $\pm$0.079	26.43 $\pm$0.057	28.39 $\pm$0.026	29.78 $\pm$0.043	32.42 $\pm$0.014

We observe that the architecture-control CanViT-B more than doubles the baseline's AP at every timestep (Table 25). Fine-tuned from the flagship checkpoint, CanViT-B improves with every glimpse up to $t = 20$. We see this limited, simple experiment as encouraging evidence that CanViT can be straightforwardly trained to perform detection tasks.

I Declarations

I.1 Availability of code and data

We release our code (https://github.com/m2b3/CanViT) and trained checkpoints (https://huggingface.co/canvit) under the MIT license, as well as the data needed to regenerate our figures and tables (https://doi.org/10.6084/m9.figshare.32148076).

I.2 Broader impact

CanViT's constant-memory recurrent architecture, fast sequential inference and scalability to large scenes make it an attractive candidate for future extension to real-time, high-FPS video processing, embodied active perception, and high-resolution static image processing. In turn, such extensions may support applications in a variety of domains, including edge/physical AI, medical imaging, and camera- or satellite-based environmental monitoring.

As with nearly all general-purpose computer vision models, derivatives of this work might be incorporated into surveillance and security technology. As such, we invite researchers building upon active-vision foundation models to engage with the societal implications of any specific application.

I.3 Compute reporting

Before submission, we used approximately 2,500 H100-equivalent hours on our SLURM-based compute platform. Our pretraining runs required under 24 GB of GPU memory (18.2 GB for the 2M-step CanViT-B pretraining run described in Appendix C). For interactive development, probe training, and downstream evaluations, we additionally used a single NVIDIA RTX 4090 workstation; we did not have fine-grained usage tracking in place there, but estimate a total of approximately 2,000 RTX 4090 GPU-hours.

The ImageNet-1k fine-tuning experiments of Appendix D.5 were run on Google Cloud TPU v6e-4 spot instances using `torch_xla`. The total TPU cost of these experiments was under 800 USD (1,318 v6e chip-hours). The total runtime of our reported IN1k fine-tuning run was under 15 wall-clock hours on TPU v6e-4 using SPMD data parallelism. We did not perform extensive optimization of our `torch_xla` code.

During the review and rebuttal period, we used a further 285 H100-equivalent hours on our SLURM-based compute platform and at least 369 GPU-hours on the RTX 4090 workstation. On Google Cloud, we used 3,431 TPU v6e and 200 TPU v5p chip-hours, at a list price of 4,796 USD.

The numbers above include failed runs, preliminary experiments, and hyperparameter sweeps.

I.4 Statistical analysis

All confidence intervals reported in this paper are 95% bootstrap CIs. For the four stochastic viewing policies (F-IID, R-IID, C2F, F2C), each *evaluation run* is one full pass over the evaluation set under a fresh policy seed. We bootstrap the mean across runs (percentile method, 10,000 resamples). We plot these CIs as shaded bands (often too tight to be visible) in Figure 4 and Figure 6. The other two non-learned policies (EG-C2F, RFS) are deterministic and yield single-run point estimates. For the learned viewing policies (Appendix D.2), each training run yields two deterministic policies, its best and last checkpoints. For each objective and checkpoint, we bootstrap the mean across its 8 training runs in the same way (Figure 4**D**, Table 4). For Figure 5**A,B**, we bootstrap a LOWESS regression across ground-truth masks (1,000 resamples; full methodology in Appendix D.6).

I.5 Licenses and attribution

Pretraining and evaluation. CanViT-B was pretrained on ImageNet-21k's `winter21_whole` split[66], and evaluated on the ImageNet-1k[37] and ADE20K[35] datasets. In Appendix H, we also pretrained CanViT-B on ImageNet-1k, and fine-tuned and evaluated detectors on COCO[65]. CanViT's DINOv3 ViT-B teacher is distributed under the DINOv3 License.

Example images. The cat image (Cat03.jpg) used as an example in figures throughout the paper is sourced from Wikimedia Commons and is attributed to Fir0002/Flagstaffotos under the CC BY-NC 3.0 license. Other example images were sourced from the Places365 dataset[87] and used solely for illustration purposes.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our key contributions are stated in the abstract and Section 1, and substantiated in the architecture description (Section 4), pretraining recipe (Section 5), and main experimental results (Section 6).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations in Section 7.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.

- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: While most of the paper is empirical, theoretical results about Viewpoint Encoding are described and supported by proofs in Appendix B.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: To reproduce our experimental results, we describe the CanViT architecture and Canvas Attention mechanism in detail (Section 4 and Appendix A), the pretraining protocol (Section 5 and Appendix C), including all necessary hyperparameters, the evaluation details (Section 6 and Appendix D), the ablation study (Appendix E), and the additional experiments (Appendix H). This complements our code release.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.

- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We release code and checkpoints under the MIT license, as well as the data necessary to regenerate figures and tables (Appendix I.1). We document the code in its repository, each checkpoint in its model card, and the data in the README of its release.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We describe training and evaluation details in Section 6, Appendix C, Appendix D, Appendix E and Appendix H.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.

- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: For experiments involving stochastic viewing policies, we report 95% bootstrap CIs over independent evaluation runs, shown as shaded bands in Figure 4 and Figure 6 and as ± half-widths or lo–hi ranges in Table 5, Table 6, Table 7, Table 8, Table 9, Table 14, Table 15, Table 17, Table 18, Table 19, Table 20, Table 22, and Table 25. For the learned viewing policies, the CIs are over independent training runs (Figure 4**D**, Table 4). See Appendix I.4 for more details.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We discuss compute workers, GPU memory, and time of execution in Appendix I.3, and storage (precomputed teacher features) in Appendix C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research we present here fully conforms with the NeurIPS Code of Ethics.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We discuss potential positive and negative impacts in Appendix I.2.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The work presented in this paper does not pose any such direct risk.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.

- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite and credit existing assets throughout the paper and in Appendix I.5.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release our code, checkpoints and data with their documentation (Appendix I.1).

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: Our paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Our methodology does not involve LLMs as any important, original, or non-standard components. However, we have made extensive use of agentic AI tools, such as Claude Code with Claude Opus 4.7 to 5.5, when developing, debugging and refactoring our code.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.