DREAMING DEEPER: STABLE RECURRENT MEDICAL IMAGE SEGMENTATION BEYOND TRAINING ROLLOUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Medical image segmentation in resource-constrained settings requires compact models that preserve accuracy under practical inference budgets. Recurrent refinement allows such models to increase test-time computation without adding parameters, but refiners trained on short rollouts can lose accuracy at greater depths. We attribute this limitation to a mismatch between recurrent states encountered during training and extended inference. We introduce **VD-Refine**, a compact U-Net-style architecture with virtual-depth (VD) supervision to bridge this gap. VD geometrically extrapolates late feature states from a short training rollout, avoiding the forward cost of their long prefixes. Initial-state-conditioned FiLM normalization controls channel-moment drift, while multi-start training supervises corrections at several real depths. VD-Refine outperforms compact baselines across BraTS, LiTS, and DRIVE. A single checkpoint supports multiple inference depths. On BraTS, VD reduces mean case-level late degradation, while a separate patch-level diagnostic shows stable accuracy far beyond the training rollout. Our analytic inference directly decodes an extrapolated state, approaching 32-step BraTS accuracy at about half the FLOPs. Our anonymous project page: `https://anonymous-vd-refine.github.io/VD-Refine/`.

1 INTRODUCTION

Medical image segmentation in resource-constrained settings requires compact models that remain accurate within practical inference budgets. In tasks such as tumor and retinal vessel segmentation, these models must still delineate small target regions and fine structures accurately. Encoder–decoder architectures have achieved strong segmentation performance (Isensee et al., 2020; Roy et al., 2023; Ma et al., 2024), while lightweight designs seek to retain this accuracy with fewer parameters (Valanarasu & Patel, 2022; Liao et al., 2024). However, their computational depth is fixed during training, limiting their ability to use additional computation at inference time. We therefore aim to build a compact model whose inference depth can be adjusted to the available compute budget without retraining.

Recurrent refinement offers a way to achieve this goal: a shared module repeatedly updates an image-conditioned representation, adding computation without adding parameters. It is a form of iterative refinement, a strategy widely used in medical image segmentation (Alom et al., 2019; Li et al., 2020; Tomar et al., 2023; Tang et al., 2021). More broadly, weight-tied networks have explored the benefits of repeated computation (McIntosh et al., 2018; Dehghani et al., 2019; Schwarzschild et al., 2021; Wang et al., 2025). However, training over long rollouts is costly, as each additional iteration requires more computation and increases the cost of storing activations for backpropagation. Refiners are therefore typically trained on short rollouts, with limited supervision of the states encountered at greater depths. Additional inference iterations may then over-adjust accurate boundaries or erase small structures, reducing segmentation accuracy. To make deeper inference useful, the refiner must preserve segmentation accuracy beyond the training rollout.

We view this loss of accuracy as a consequence of the mismatch between states seen during training and those reached during extended inference. As refinement continues, feature statistics may drift, while the remaining segmentation errors may require different corrections from those learned at ear-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

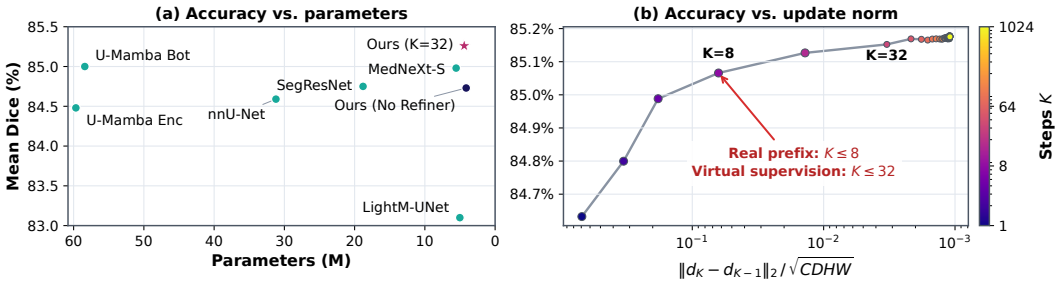


Figure 1: BraTS accuracy/parameter comparison and test-time depth diagnostic. **(a)** Average Dice versus parameter count on 74 held-out volumes. VD-Refine (labeled Ours) at $K = 32$ achieves the highest average Dice among the methods shown, with fewer parameters than the full-size baselines and only a small increase over the no-Refiner backbone. **(b)** Patch-level mechanism diagnostic on 500 fixed held-out BraTS patches. Complete real rollouts from $K = 1$ to $K = 1024$ are plotted against the step-to-step residual $\|d_K - d_{K-1}\|_2/\sqrt{CDHW}$. Training rollouts stop at $K = 8$, with VD supervision to $K = 32$ ($4\times$ the training rollout length). Patch Dice remains stable through $K = 1024$ ($128\times$), while update magnitudes decrease.

lier depths. Updates that improve early predictions can therefore become redundant or harmful later. Stabilizing feature statistics alone is insufficient: even a bounded feature state can decode to a mask that misses a tumor region. This motivates both controlling feature drift and supervising corrections at later states. We assess stability through feature dynamics and the retention of segmentation accuracy over the evaluated depths.

Obtaining late states for this supervision is costly, as reaching each state requires executing all preceding updates. Truncated backpropagation reduces activation storage and shortens the gradient horizon, but does not eliminate the forward cost of generating these states. Supervising several local windows along a short trajectory covers a wider range of early states, but still leaves later states unseen. This raises our central question: *can we estimate late states from a short training rollout and use them to supervise refinement beyond that rollout?*

We propose **VD-Refine**, a compact recurrent segmentation model built around virtual-depth (VD) supervision. The core idea is to "dream deeper" by geometrically extrapolating late feature states from a short training rollout. These estimated states initialize label-supervised two-step correction windows, providing supervision at greater depths without executing the long prefixes needed to reach them. Initial-state-conditioned FiLM normalization gives real and estimated states a common reference for their channel statistics while allowing spatial content to evolve. Multi-start training complements VD supervision by supervising corrections at several depths along the real rollout. We implement these components in a compact U-Net-style architecture, where a weight-tied Refiner repeatedly updates a decoder feature state using fixed image context.

Experiments on BraTS, LiTS, and DRIVE yield three main findings, using a single checkpoint per run across inference depths. First, late-state supervision supports refinement beyond the training rollout. On BraTS, the model is trained with eight-step rollouts yet retains its accuracy gains at 32 steps; a separate patch-level diagnostic shows stable accuracy through 1,024 steps. Second, VD-Refine outperforms nnU-Net on all three datasets, using approximately one-seventh as many parameters in the BraTS configuration. Third, the same checkpoint supports *analytic inference*, which directly decodes an extrapolated state and approaches 32-step BraTS accuracy at about half the FLOPs.

Our contributions are threefold: (1) We introduce virtual-depth supervision to train refiners on estimated late states. Geometric extrapolation supplies the starting states for label-supervised two-step correction windows, extending supervision beyond short training rollouts without executing long prefixes. (2) We develop VD-Refine, a compact U-Net-style model that combines this supervision with a weight-tied Refiner, initial-state-conditioned FiLM, and multi-start training to support adjustable inference depth at a fixed parameter count. (3) We demonstrate accurate segmentation across BraTS, LiTS, and DRIVE, with one checkpoint per run supporting both multiple refinement depths and analytic inference. BraTS experiments show improved mean Dice beyond the training rollout and stable accuracy over extended rollouts in a separate patch-level diagnostic.

2 Related Work

Efficient medical image segmentation. U-Net and nnU-Net established strong encoder–decoder approaches (Ronneberger et al., 2015; Isensee et al., 2020). UNeXt and MedNeXt improve efficiency through lightweight mixing and convolutional design (Valanarasu & Patel, 2022; Roy et al., 2023), while U-Mamba and LightM-UNet incorporate state-space modeling (Ma et al., 2024; Liao et al., 2024). VD-Refine complements backbone design by inserting a shared Refiner into the decoder and training it to support adjustable inference depth.

Recurrent and iterative segmentation. R2U-Net accumulates features through recurrent residual convolutions within U-Net (Alom et al., 2019); IterNet repeatedly applies mini-U-Nets with shared weights for retinal vessel refinement (Li et al., 2020). Recurrent mask refinement also improves few-shot medical segmentation (Tang et al., 2021). Beyond medical imaging, McIntosh et al. (2018) explicitly vary recurrent inference iterations to accommodate different computational budgets. Building on these uses of recurrence, VD-Refine focuses on supervising corrections at estimated states beyond the training rollout.

Training and scaling recurrent models. Weight-tied networks separate execution depth from parameter count (Dehghani et al., 2019; Schwarzschild et al., 2021); recurrent language models scale test-time computation through latent updates (Geiping et al., 2025). Deep equilibrium models learn implicit fixed-point representations (Bai et al., 2019; 2020), including for segmentation (Zhang et al., 2022; Chau et al., 2022). LoopFormer trains variable-length trajectories with shortcut consistency and time/step-size conditioning to support different compute budgets (Jeddi et al., 2026). VD-Refine targets the forward cost of obtaining late states for explicit recurrent training: gradient truncation shortens backward history, but a real late start still requires its full prefix. Geometric extrapolation estimates late starts from a short trajectory for label-supervised local corrections, extending training-state coverage without executing those prefixes.

3 Method

VD-Refine refines a decoder feature state with a weight-tied transition. FiLM controls channel-moment drift, multi-start training supervises real states, and virtual-depth supervision extends local correction training to estimated late states beyond the training rollout (Figure 2).

3.1 Recurrent Segmentation Architecture

We decompose a U-shaped encoder–decoder into a Prelude, Refiner, and Coda (Geiping et al., 2025; McLeish et al., 2025). Given a 2D or 3D image x, the encoder and initial decoder stages (Prelude) produce d_0 and fixed context s. A weight-tied Refiner updates this state; the Coda comprises the remaining decoder stages and prediction head:

$$(d_0, s) = \mathcal{P}_\phi(x), \quad d_{t+1} = R_\theta(d_t; d_0, s), \quad \hat{y}_K = \mathcal{C}_\psi(d_K, s_D), \tag{1}$$

where $s_D \subseteq s$ contains the Coda's skip features. The Prelude and Coda execute once; the Refiner executes K times without changing the parameter count. At $K = 0$, we decode the same checkpoint's initial proposal, distinct from a separately trained no-Refiner backbone.

We instantiate Eq. 1 with LiteMB-UNet, a U-Net backbone using MobileNetV2-style inverted residual blocks (Sandler et al., 2018), depthwise spatial convolutions, and efficient channel attention (Wang et al., 2020). Shifted-MLP mixing is restricted to coarse resolutions. In BraTS, the

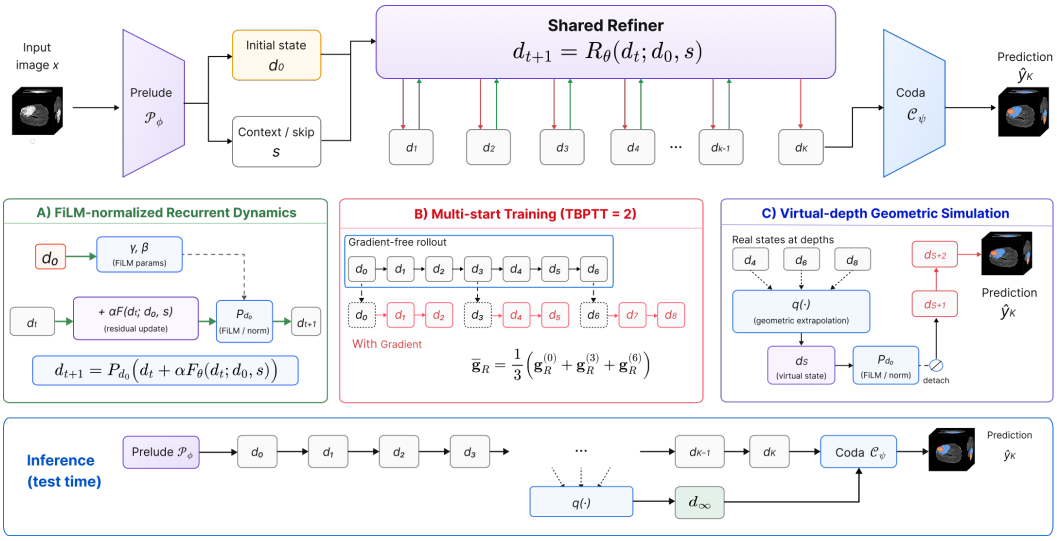


Figure 2: **VD-Refine Overview.** A weight-tied Refiner updates a selected decoder feature state while image context remains fixed. Training combines **(A)** d_0-conditioned FiLM normalization, **(B)** two-step windows from multiple real starts, and **(C)** geometrically constructed late starts followed by two real corrections. The bottom panel shows real-rollout inference and optional analytic inference.

Refiner acts on the third of five decoder stages, counting from the bottleneck. Lightweight adapters map the state and fixed context into a shared correction core and project its output back to the state space. In the BraTS configuration, the backbone has 4.12M parameters and the complete conditioned Refiner adds 0.28M; architecture details are in Appendix A.1.

3.2 Initial-State-Conditioned FiLM Normalization

Repeated residual updates can cause state-scale drift (Yang et al., 2026). We normalize each updated state and condition its affine parameters on d_0, following FiLM (Perez et al., 2018):

$$R_\theta(d_t; d_0, s) = P_{d_0}(d_t + \alpha F_\theta(d_t; d_0, s)), \qquad P_{d_0}(z) = \gamma \odot \mathrm{IN}(z) + \beta, \tag{2}$$

where $(\gamma, \beta) = h(\mathrm{GAP}(d_0))$. Here, GAP averages spatial dimensions, and the learned MLP h maps the resulting channel descriptor to FiLM scales γ and shifts β. IN normalizes each channel over spatial positions without its own affine transform, and α is the residual step size. The same sample-dependent γ, β are reused at every step. Thus, for $t \geq 1$, the spatial channel mean is $\mu_c(d_t) = \beta_c$, and $\sigma_c(d_t) \approx |\gamma_c|$ when the pre-normalization channel variance is large relative to the numerical stabilizer ϵ. The shared affine parameters place recurrent states on a common channel-moment reference while allowing spatial content to evolve. Local supervision then teaches the Refiner which spatial corrections are useful at different depths. Throughout the paper, P_{d_0} denotes this conditional normalization map.

3.3 Local Supervision Across Recurrent Depths

Full backpropagation through a long recurrent trajectory requires storing intermediate activations across all steps. Truncated backpropagation (Werbos, 1990) reduces this memory cost by restricting the gradient horizon; a two-step window starting at depth ρ still requires ρ forward updates to obtain its real initial state. We address this remaining cost through the choice of starting states. Multi-start supervision trains corrections from several real states within a short prefix, and virtual-depth supervision adds estimated late starts derived from that same prefix. Both use two-step gradient windows, separating the depth of supervised states from the length of backpropagation.

Real-state multi-start supervision. We generate a gradient-free training rollout $\bar{d}_{0:8}$ and sample one starting depth τ_b from each of three depth bands (Algorithm 1). From each detached state, we

execute two gradient-tracked updates and apply the segmentation loss ℓ:

$$\mathcal{L}_{\mathrm{MS}} = \frac{1}{3}\sum_{b=1}^{3} \ell\big(\mathcal{C}_\psi\big(R_\theta^2(\mathrm{stopgrad}(\bar{d}_{\tau_b})), s_D\big), y\big)\,. \tag{3}$$

Here R_θ^2 denotes two successive Refiner applications with the fixed conditioning d_0, s omitted for readability. The averaged losses supervise corrections to both initial and refined states, with detachment limiting each gradient window to two steps.

Virtual-depth supervision via geometric extrapolation. Real-state multi-start supervision covers only the short training rollout, leaving deeper inference states outside its training coverage. We extend this coverage by estimating a starting state at $\rho = S - 2$ for a virtual supervised endpoint S. Every estimate uses the same eight-step training rollout; increasing the virtual target depth requires no additional prefix updates. We use the real states d_4, d_6, d_8, writing d_t for the gradient-free reference states in this construction. Shared FiLM conditioning provides a common channel-moment reference for these displacements.

With $u_6 = d_6 - d_4$ and $u_8 = d_8 - d_6$, we estimate a per-sample two-step decay coefficient:

$$q = \mathrm{clip}\left(\frac{\langle u_8, u_6\rangle}{\|u_6\|_2^2 + \epsilon}, 0, 0.95\right), \qquad \tilde{d}_\rho = d_8 + q\frac{1 - q^{(\rho-8)/2}}{1-q}\,u_8. \tag{4}$$

The scalar q fits the approximate relation $u_8 \approx q u_6$ and is clipped to keep the geometric series bounded. Continuing this local model assumes that subsequent two-step displacements retain approximately the direction of u_8 and decay by a factor of q. The geometric sum accumulates these diminishing increments; half-integer exponents interpolate the construction to odd ρ. This is a local approximation to the trajectory, whose fidelity we assess empirically in Section 4.4. We restore the conditioned channel moments, detach the result, and execute two real corrections:

$$z_0 = \mathrm{stopgrad}\Big(P_{d_0}(\tilde{d}_\rho)\Big), \quad z_2 = R_\theta^2(z_0), \quad \mathcal{L}_{\mathrm{VD}} = \ell(\mathcal{C}_\psi(z_2, s_D), y)\,. \tag{5}$$

Segmentation labels train the two real corrections while the estimated start remains detached. Geometric extrapolation captures diminishing motion, unlike copying d_8 or repeating a constant displacement, and avoids generating exact late training states.

Unified real- and virtual-start objective. Under the BraTS schedule, the Refiner-only phases minimize

$$\mathcal{L}_B = (1 - w_v)\mathcal{L}_{\mathrm{MS}} + w_v\mathcal{L}_{\mathrm{VD}}. \tag{6}$$

The virtual weight and target range increase during these phases. Training rollouts stop at eight steps, while virtual endpoints extend to depth 32, four times the training rollout length; all local gradient windows have length two. In this schedule, the surrounding network and FiLM mapping are fixed during Refiner-only updates and recalibrated in separate phases. Algorithm 1 details the 100-epoch schedule: shallow joint training, Refiner-only training, surrounding-network realignment, a second Refiner-only phase, and decoder-tail calibration.

3.4 Inference Procedures

The same checkpoint supports two ways to obtain the state decoded by the Coda.

Real-rollout inference. For a preset depth K, we compute d_0, execute K shared Refiner updates, and decode d_K once. We use this procedure for the finite-K segmentation results and long-rollout diagnostics.

Analytic inference. For *analytic inference*, we execute an eight-step real prefix, fit q from d_4, d_6, d_8 using Eq. 4, and form the extrapolated state $\hat{d}_\infty$ from the limit of the fitted geometric series:

$$\hat{d}_\infty = P_{d_0}\left(d_8 + \frac{q}{1-q}(d_8 - d_6)\right). \tag{7}$$

We then decode $\mathcal{C}_\psi(\hat{d}_\infty, s_D)$ directly, without further Refiner steps. The additional cost beyond eight real steps is extrapolation and normalization. The subscript ∞ denotes the limit of the fitted series, rather than a guaranteed fixed point of the learned recurrence. We evaluate its decoding accuracy and separately test its compatibility with the Refiner by continuing real updates from the estimated state.

Table 1: **Segmentation accuracy and model cost.** Dice (%) is evaluated on the full held-out cohorts. VD-Refine scores report means across runs, with standard deviations (SD) in the rows below. Both inference modes share a checkpoint within each run; $K = \hat{\infty}$ denotes analytic inference from eight real steps. Costs use the BraTS four-channel 128^3 configuration on an H100 at batch size one.

Method	BraTS ↑				LiTS ↑			DRIVE ↑	Model cost ↓		
	Avg	WT	TC	ET	**Avg**	Liver	Tumor	**Dice**	Params (M)	FLOPs (G)	Runtime (ms)
Larger baselines											
nnU-Net (2020)	84.59	90.82	85.90	76.92	77.30	91.54	63.06	82.21	31.20	969.04	23.40 ± 0.06
U-Mamba Enc (2024)	84.48	**91.77**	86.06	76.20	79.27	**94.47**	64.07	82.26	59.68	2081.31	106.86 ± 0.11
U-Mamba Bot (2024)	85.00	91.57	86.70	76.74	77.78	92.24	63.33	82.20	58.41	1990.50	51.94 ± 0.15
SegResNet (2018)	84.75	91.14	86.77	76.34	76.75	91.71	61.80	82.21	18.80	1173.86	70.80 ± 0.04
Compact baselines											
UNeXt (2022)	82.61	89.68	83.81	74.34	69.39	89.93	48.85	80.64	3.87	29.61	5.37 ± 0.01
MedNeXt-S (2023)	84.98	91.14	85.63	**78.18**	76.64	92.60	60.67	82.19	5.55	260.44	67.03 ± 1.57
LightM-UNet (2024)	83.10	90.50	82.78	76.02	69.27	89.98	48.56	81.09	5.02	412.66	254.74 ± 0.13
Our backbone and recurrent variants											
LiteMB-UNet (no Refiner)	84.73	90.96	86.94	76.30	77.27	93.93	60.61	82.26	4.12	163.42	46.66 ± 0.02
VD-Refine ($K = 32$)	**85.258**	**91.259**	**87.061**	77.299	**79.734**	**94.286**	**65.183**	**82.364**	4.40	494.26	68.44 ± 1.59
SD	0.185	0.172	0.155	0.515	1.514	0.278	2.751	0.187			
VD-Refine ($K = \hat{\infty}$)	**85.239**	**91.245**	**86.999**	77.318	**79.946**	**94.310**	**65.582**	**82.337**	4.40	251.09	51.82 ± 0.03
SD	0.161	0.161	0.127	0.507	1.084	0.403	1.774	0.212			

Notation. Each VD-Refine mode is compared independently with baselines: **bold**, above all baselines; **bold underlined**, above all compact baselines. Bold in baseline rows marks the best Dice across all methods.

4 Experiments

We evaluate segmentation accuracy and computational cost on three datasets, then examine how performance changes with recurrent depth. Component ablations and state-level analyses assess how conditioned normalization, multi-start training, and virtual-depth supervision affect refinement, with all depth comparisons using a fixed checkpoint for each configuration.

4.1 Experimental Setup

Datasets and metrics. We use BraTS 2020 (Bakas et al., 2018), LiTS (Bilic et al., 2023), and DRIVE (Staal et al., 2004). BraTS and LiTS provide 3D MRI and CT volumes, respectively, while DRIVE contains 2D retinal fundus images. Training/validation/test splits are $236/59/74$ volumes, $92/13/26$ volumes, and $16/4/20$ images, respectively. We report Dice in percent and differences in percentage points (pp). BraTS mDice is the per-case mean of valid whole tumor (WT), tumor core (TC), and enhancing tumor (ET) Dice, omitting NaNs; reported Avg averages mDice across cases. LiTS averages liver and tumor; DRIVE uses vessel Dice. BraTS and DRIVE use test-time augmentation (TTA); LiTS does not.

Training and evaluation. Our base schedule alternates three types of phases: A jointly trains the model, B optimizes the shared Refiner core, and C adapts the surrounding network with the core frozen. After warm-up, we alternate B and C for two rounds, for a total of 100 training epochs. Algorithm 1 specifies the BraTS schedule; dataset-specific adaptations and optimization settings are detailed in Appendix A.3. Case-paired diagnostic protocols appear in Appendix A.4.

4.2 Segmentation Accuracy and Computational Cost

Segmentation accuracy and parameter efficiency. Table 1 shows that VD-Refine improves mean Dice over the no-Refiner backbone with a modest parameter increase. VD-Refine surpasses all evaluated lightweight baselines in mean Dice across the three datasets and also outperforms larger models, including both U-Mamba variants with over ten times as many parameters in the BraTS configuration. Relative to the lightweight MedNeXt-S (Roy et al., 2023), analytic inference (Section 3.4) uses 20.7% fewer parameters in the BraTS configuration, with mean Dice gains of 0.259, 3.306, and 0.147 pp on BraTS, LiTS, and DRIVE (Table 1).

Model cost. VD-Refine retains a compact parameter budget while requiring substantially fewer FLOPs than nnU-Net, both U-Mamba variants, and SegResNet in the BraTS configuration (Table 1).

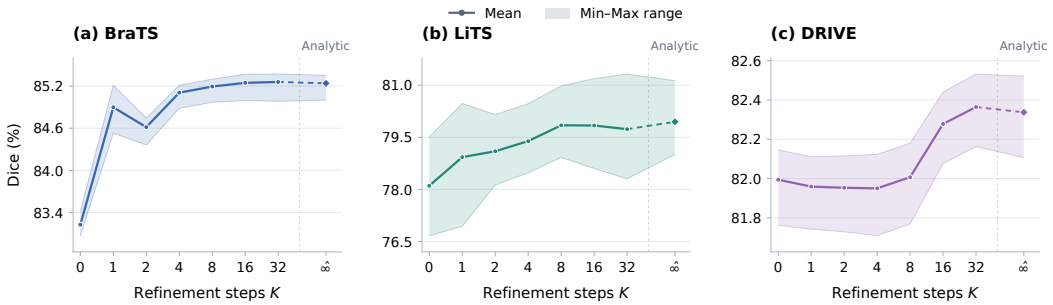


Figure 3: **Test-time depth scaling.** Complete held-out cohorts: 74 BraTS volumes, 26 LiTS volumes, and 20 DRIVE images. Solid curves show mean Dice across runs, and shading spans the minimum–maximum range at each real depth. The diamond and range at $K = \hat{\infty}$ show analytic inference from eight real steps.

With analytic inference, its FLOPs are also slightly below MedNeXt-S and lower than LightM-UNet, while achieving higher mean Dice than both methods on all three datasets. Additional runtime measurements are provided in Appendix D.

4.3 Test-Time Depth Scaling

Beyond training rollouts. Figure 3 compares real-step depth scaling and analytic inference across all three datasets. On BraTS, mean Dice improves from 83.23% at $K = 0$ to 85.19% at $K = 8$ and 85.26% at $K = 32$, retaining the early gains beyond the training rollout. LiTS benefits further from analytic inference, reaching its highest mean Dice of 79.95% among the evaluated modes. On DRIVE, mean Dice increases from 82.01% at $K = 8$ to 82.36% at $K = 32$; analytic inference achieves 82.34%. Both $K = 32$ refinement and analytic inference outperform the no-Refiner and nnU-Net baselines on all three datasets. On 500 fixed held-out BraTS patches, mean Dice stays near 85.2% (range 85.15–85.18%) at sampled depths from $K = 32$ to 1024, reaching $128\times$ the training rollout length and $32\times$ the deepest virtual supervised depth (Figure 1b).

Case-level reliability at greater depth. We quantify case-level late degradation as $D_i = \max\{0, \mathrm{Dice}_8(i) - \min(\mathrm{Dice}_{16}(i), \mathrm{Dice}_{32}(i))\}$. On BraTS, virtual-depth supervision (VD) reduces mean D_i by 75.6%, from 0.234 to 0.057 pp. Across the 74 held-out cases, its 95th percentile falls from 1.426 to 0.306 pp and its maximum from 3.010 to 0.733 pp, indicating lower average and tail degradation. Appendix B.3 reports distributions for all three datasets.

4.4 Ablation and Mechanism Analysis

Component ablations. Table 2 evaluates three parts of VD-Refine: the alternating schedule with real-state multi-start supervision, conditioned FiLM normalization, and virtual-depth supervision (VD). The shallow and tail-TBPTT configurations (A1–A6) all lose accuracy from $K = 8$ to 32. Alternating multi-start training achieves higher $K = 32$ Dice with either no normalization or FiLM (A7 and A9), but late-depth losses remain. Within this schedule, plain InstanceNorm (A8) gives a positive late-depth gain at the cost of lower absolute Dice than A7 and A9. Adding VD to the FiLM model (A9–A10) yields the highest $K = 32$ Dice, improving it by 0.53 pp and changing the $K = 8 \to 32$ gain from -0.14 to $+0.07$ pp.

Real-state multi-start supervision. Different real starting depths provide aligned but distinct supervision signals. At the checkpoint after the initial joint phase, we independently compute Refiner gradients from the two-step windows $d_1 \to d_3$, $d_3 \to d_5$, and $d_6 \to d_8$. Across 500 held-out patches sampled with training-time foreground oversampling, the median gradient cosine is 0.887 for the early–late pair and $0.962/0.966$ for neighboring pairs. The shared direction and depth-dependent differences support combining several real starting depths within an update; Figure 12 in the appendix gives the distributions.

Table 2: **BraTS component ablations on 74 full volumes.** Joint trains all parameters; $A \to B$ uses 50 joint warm-up epochs followed by 50 Refiner-only epochs; Alternating follows the five-stage schedule. Shallow supervises $K \in \{1, 2\}$; Tail TBPTT-2 backpropagates through at most the last two updates at random $K \in \{1, \ldots, 8\}$; Multi-start combines two-step windows from several starting depths. Norm denotes state normalization, and VD denotes virtual-depth supervision. $\Delta_{8\to32}$ is the within-checkpoint mean Dice change from $K = 8$ to 32 (pp). Full results are in Table 3.

ID	Schedule	Supervision	Norm	VD	mDice$_{32}$ ↑	$\Delta_{8\to32}$ ↑
Training schedules and short-gradient strategies						
A1	Joint	Shallow $K \in \{1, 2\}$	None	No	84.33	-0.36
A2	Joint	Shallow $K \in \{1, 2\}$	FiLM	No	83.89	-0.34
A3	Joint	Tail TBPTT-2	None	No	84.34	-0.29
A4	Joint	Tail TBPTT-2	FiLM	No	84.57	-0.14
A5	$A \to B$	Tail TBPTT-2	None	No	83.65	-0.80
A6	$A \to B$	Tail TBPTT-2	FiLM	No	84.37	-0.55
Normalization and virtual depth with alternating multi-start training						
A7	Alternating	Multi-start	None	No	84.73	-0.16
A8	Alternating	Multi-start	InstanceNorm	No	84.45	+0.05
A9	Alternating	Multi-start	FiLM	No	84.73	-0.14
A10	Alternating	Multi-start	FiLM	Yes	**85.26**	**+0.07**

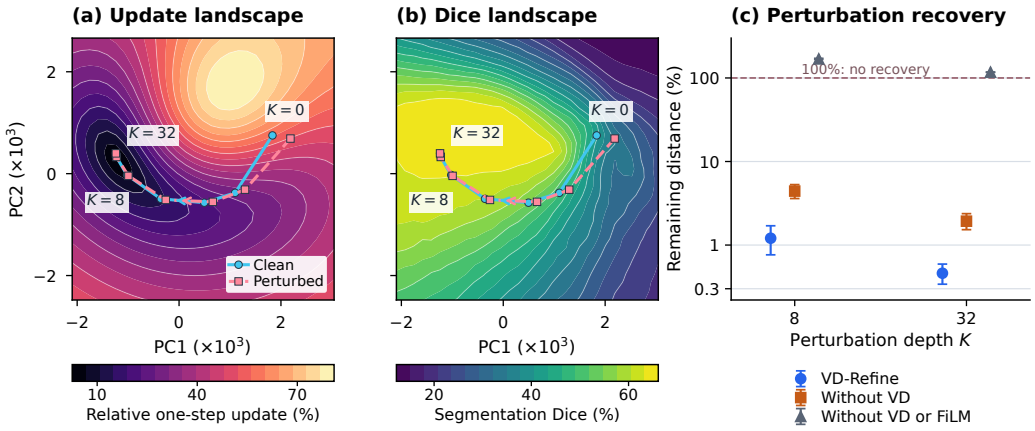


Figure 4: **Local geometry and perturbation recovery. (a,b)** Relative one-step update magnitude and Dice on a two-dimensional PCA slice for one held-out BraTS patch, with projected clean and perturbed rollouts. **(c)** Percentage of injected feature distance remaining after 64 updates (lower is better), with moment-restored noise at $K = 8, 32$. Points and bars show means and 95% case-bootstrap intervals over 74 case patches.

Conditioned normalization across depth. FiLM keeps the state norm near its initial scale through $K = 1024$, whereas the unnormalized norm grows by roughly $146\times$. With FiLM, update magnitudes decrease by nearly three orders of magnitude and prediction changes approach zero (Figure 10, Appendix C.1).

Local geometry and perturbation recovery. For the illustrated patch, both projected rollouts approach a lower-update, higher-Dice region (Figure 4a,b). Across 74 case patches, the mean distance remaining after 64 updates from moment-restored perturbations at $K = 8, 32$ is $1.207\%/0.460\%$ with VD, versus $4.424\%/1.931\%$ without VD (Figure 4c). VD therefore improves recovery toward the clean feature trajectory; the small initial Dice changes limit this result to feature recovery. Protocols and paired task measurements are in Appendices C.2–C.3.

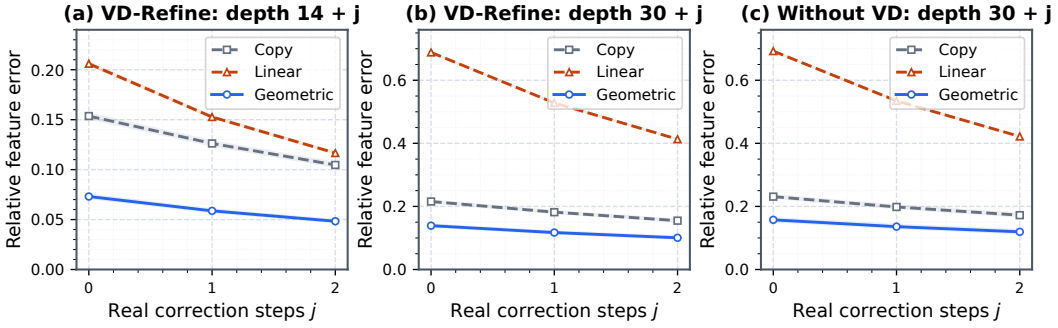


Figure 5: **Late-state approximation.** Relative feature error against the matched real state at depth $\rho + j$, after j real corrections from copied, linear, or geometric starts. Copy reuses the real state d_8 without extrapolation. Panels compare $\rho = 14, 30$ for VD-Refine and $\rho = 30$ without VD. The same fixed patch from each of 74 cases is used across conditions; shaded bands show 95% case-bootstrap CIs (Appendix A.4).

Accuracy of virtual late states. Geometric extrapolation supplies more accurate virtual starts than copying or linear extrapolation under matched conditioning and FiLM normalization (Figure 5). At depth 30, the complete model's mean relative feature error is 0.139 for geometric extrapolation, compared with 0.216 for copying and 0.688 for linear extrapolation. Two real corrections further reduce these errors against d_{32} to 0.101, 0.155, and 0.413, respectively. The same ordering holds without VD, indicating that short trajectories already contain useful information for estimating late training states.

Further refinement after analytic inference. After 64 real updates from $\hat{d}_\infty$, relative feature drift reaches 0.151, yet mean patch Dice changes by only $+0.073$ pp. This distinguishes stable decoding from a feature fixed point; residuals, full trajectories, and the d_{1024} reference appear in Appendix C.8.

5 Discussion

Virtual-depth supervision expands training-state coverage and improves BraTS accuracy and case-level reliability. Geometric estimates are useful with and without VD, suggesting that short rollouts contain information about later states. The InstanceNorm ablation shows that late-depth gains alone do not ensure high absolute Dice. Extrapolation relies on approximately aligned, decaying displacements; LiTS retains heavier case-level degradation tails, and the $K = 1024$ evidence is patch-level. Our stability claims are empirical and restricted to the evaluated trajectories.

6 Conclusion

We presented VD-Refine, a compact recurrent U-Net-style model trained on real states and geometric estimates of late states beyond its training rollout. FiLM controls channel moments, while local supervision trains the shared Refiner across depths. On BraTS, the model achieves 85.26% mean Dice with 4.40M parameters at four times the training rollout length. VD improves case-level reliability over training without VD, and a separate patch-level diagnostic preserves stable accuracy through $K = 1024$. Fixed checkpoints support multiple depths across datasets, with analytic inference reducing FLOPs in the BraTS configuration.

AI USE STATEMENT

Generative AI tools assisted with literature retrieval, drafting and revising manuscript text, checking mathematical expressions and consistency, writing statistical-analysis and plotting code, and interpreting experimental results. The authors take responsibility for verifying all AI-assisted material and for the final text, code, results, and claims.

ETHICS STATEMENT

This study uses retrospective medical segmentation benchmarks. Its experiments assess segmentation performance rather than clinical decision-making; clinical deployment requires external validation across patient populations and acquisition settings.

REPRODUCIBILITY STATEMENT

Section 3 gives the update and supervision objectives; Algorithm 1 specifies the training schedule. Appendix A documents the architecture, preprocessing, and evaluation protocols. Appendices B and C report additional results and mechanism diagnostics; Appendix D specifies the timing protocols. Anonymous code, with dataset-specific protocols and verification notes, is available at `https://anonymous-vd-refine.github.io/VD-Refine/`.

REFERENCES

Md Zahangir Alom, Chris Yakopcic, Mahmudul Hasan, Tarek M. Taha, and Vijayan K. Asari. Recurrent residual u-net for medical image segmentation. *Journal of Medical Imaging*, 6(01):1, March 2019. ISSN 2329-4302. doi: 10.1117/1.jmi.6.1.014006. URL `http://dx.doi.org/10.1117/1.jmi.6.1.014006`.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep equilibrium models. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 688–699, 2019. URL `https://proceedings.neurips.cc/paper/2019/hash/01386bd6d8e091c2ab4c7c7de644d37b-Abstract.html`.

Shaojie Bai, Vladlen Koltun, and J. Zico Kolter. Multiscale deep equilibrium models. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin (eds.), *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/3812f9a59b634c2a9c574610eaba5bed-Abstract.html`.

Spyridon Bakas, Mauricio Reyes, Andras Jakab, Stefan Bauer, Markus Rempfler, Alessandro Crimi, Russell Takeshi Shinohara, Christoph Berger, Sung Min Ha, Martin Rozycki, et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge. *arXiv preprint arXiv:1811.02629*, 2018. doi: 10.48550/arXiv.1811.02629.

Patrick Bilic, Patrick Christ, Hongwei Bran Li, Eugene Vorontsov, Avi Ben-Cohen, Georgios Kaissis, Adi Szeskin, Colin Jacobs, Gabriel Efrain Humpire Mamani, Gabriel Chartrand, Fabian Lohöfer, Julian Walter Holch, et al. The liver tumor segmentation benchmark (LiTS). *Medical Image Analysis*, 84:102680, 2023. doi: 10.1016/j.media.2022.102680.

Nguyen Minh Chau, Le Truong Giang, and Dinh Viet Sang. PolypDEQ: Towards effective transformer-based deep equilibrium models for colon polyp segmentation. In *Advances in Visual Computing*, volume 13598 of *Lecture Notes in Computer Science*, pp. 456–467. Springer, 2022. doi: 10.1007/978-3-031-20713-6_35.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Lukasz Kaiser. Universal transformers. In *7th International Conference on Learning Representations, ICLR 2019, New*

Orleans, LA, USA, May 6-9, 2019. OpenReview.net, 2019. URL https://openreview.net/forum?id=HyzdRiR9Y7.

Jonas Geiping, Sean McLeish, Neel Jain, John Kirchenbauer, Siddharth Singh, Brian 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 38*, NeurIPS 2025, pp. 46242–46293. Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2025. doi: 10.52202/085713-1380. URL http://dx.doi.org/10.52202/085713-1380.

Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature Methods*, 18(2):203–211, December 2020. ISSN 1548-7105. doi: 10.1038/s41592-020-01008-z. URL http://dx.doi.org/10.1038/s41592-020-01008-z.

Ahmadreza Jeddi, Marco Ciccone, and Babak Taati. LoopFormer: Elastic-depth looped transformers for latent reasoning via shortcut modulation. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=RzYXb5YWBs.

Liangzhi Li, Manisha Verma, Yuta Nakashima, Hajime Nagahara, and Ryo Kawasaki. Iternet: Retinal image segmentation utilizing structural redundancy in vessel networks. In *IEEE Winter Conference on Applications of Computer Vision, WACV 2020, Snowmass Village, CO, USA, March 1-5, 2020*, pp. 3645–3654. IEEE, 2020. doi: 10.1109/WACV45572.2020.9093621. URL https://doi.org/10.1109/WACV45572.2020.9093621.

Weibin Liao, Yinghao Zhu, Xinyuan Wang, Chengwei Pan, Yasha Wang, and Liantao Ma. Lightm-unet: Mamba assists in lightweight unet for medical image segmentation. *CoRR*, abs/2403.05246, 2024. doi: 10.48550/ARXIV.2403.05246. URL https://doi.org/10.48550/arXiv.2403.05246.

Jun Ma, Feifei Li, and Bo Wang. U-mamba: Enhancing long-range dependency for biomedical image segmentation. *CoRR*, abs/2401.04722, 2024. doi: 10.48550/ARXIV.2401.04722. URL https://doi.org/10.48550/arXiv.2401.04722.

Lane McIntosh, Niru Maheswaranathan, David Sussillo, and Jonathon Shlens. Recurrent segmentation for variable computational budgets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1648–1657, June 2018. doi: 10.1109/CVPRW.2018.00216. URL https://arxiv.org/abs/1711.10151.

Sean McLeish, Ang Li, John Kirchenbauer, Dayal Singh Kalra, Brian R. Bartoldson, Bhavya Kailkhura, Avi Schwarzschild, Jonas Geiping, Tom Goldstein, and Micah Goldblum. Teaching pretrained language models to think deeper with retrofitted recurrence. *CoRR*, abs/2511.07384, 2025. doi: 10.48550/ARXIV.2511.07384. URL https://doi.org/10.48550/arXiv.2511.07384.

Andriy Myronenko. 3d MRI brain tumor segmentation using autoencoder regularization. In Alessandro Crimi, Spyridon Bakas, Hugo J. Kuijf, Farahani Keyvan, Mauricio Reyes, and Theo van Walsum (eds.), *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries - 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II*, volume 11384 of *Lecture Notes in Computer Science*, pp. 311–320. Springer, 2018. doi: 10.1007/978-3-030-11726-9_28. URL https://doi.org/10.1007/978-3-030-11726-9_28.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville. Film: Visual reasoning with a general conditioning layer. In Sheila A. McIlraith and Kilian Q. Weinberger (eds.), *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 3942–3951. AAAI Press, 2018. doi: 10.1609/AAAI.V32I1.11671. URL https://doi.org/10.1609/aaai.v32i1.11671.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells III, and Alejandro F. Frangi (eds.), *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015 - 18th International Conference Munich, Germany, October 5 - 9, 2015, Proceedings, Part III*, volume 9351 of *Lecture Notes in Computer Science*, pp. 234–241. Springer, 2015. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

Saikat Roy, Gregor Köhler, Constantin Ulrich, Michael Baumgartner, Jens Petersen, Fabian Isensee, Paul F. Jäger, and Klaus H. Maier-Hein. Mednext: Transformer-driven scaling of convnets for medical image segmentation. In Hayit Greenspan, Anant Madabhushi, Parvin Mousavi, Septimiu E. Salcudean, James Duncan, Tanveer F. Syeda-Mahmood, and Russell H. Taylor (eds.), *Medical Image Computing and Computer Assisted Intervention - MICCAI 2023 - 26th International Conference, Vancouver, BC, Canada, October 8-12, 2023, Proceedings, Part IV*, volume 14223 of *Lecture Notes in Computer Science*, pp. 405–415. Springer, 2023. doi: 10.1007/978-3-031-43901-8_39. URL https://doi.org/10.1007/978-3-031-43901-8_39.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Avi Schwarzschild, Eitan Borgnia, Arjun Gupta, Furong Huang, Uzi Vishkin, Micah Goldblum, and Tom Goldstein. Can you learn an algorithm? generalizing from easy to hard problems with recurrent networks. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pp. 6695–6706, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/3501672ebc68a5524629080e3ef60aef-Abstract.html.

Joes Staal, Michael D. Abràmoff, Meindert Niemeijer, Max A. Viergever, and Bram van Ginneken. Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509, 2004. doi: 10.1109/TMI.2004.825627.

Hao Tang, Xingwei Liu, Shanlin Sun, Xiangyi Yan, and Xiaohui Xie. Recurrent mask refinement for few-shot medical image segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3918–3928, October 2021. doi: 10.1109/ICCV48922.2021.00389.

Nikhil Kumar Tomar, Debesh Jha, Michael A. Riegler, Håvard D. Johansen, Dag Johansen, Jens Rittscher, Pål Halvorsen, and Sharib Ali. Fanet: A feedback attention network for improved biomedical image segmentation. *IEEE Trans. Neural Networks Learn. Syst.*, 34(11):9375–9388, 2023. doi: 10.1109/TNNLS.2022.3159394. URL https://doi.org/10.1109/TNNLS.2022.3159394.

Jeya Maria Jose Valanarasu and Vishal M. Patel. Unext: Mlp-based rapid medical image segmentation network. In Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li (eds.), *Medical Image Computing and Computer Assisted Intervention - MICCAI 2022 - 25th International Conference, Singapore, September 18-22, 2022, Proceedings, Part V*, volume 13435 of *Lecture Notes in Computer Science*, pp. 23–33. Springer, 2022. doi: 10.1007/978-3-031-16443-9_3. URL https://doi.org/10.1007/978-3-031-16443-9_3.

Guan Wang, Jin Li, Yuhao Sun, Xing Chen, Changling Liu, Yue Wu, Meng Lu, Sen Song, and Yasin Abbasi-Yadkori. Hierarchical reasoning model. *CoRR*, abs/2506.21734, 2025. doi: 10.48550/ARXIV.2506.21734. URL https://doi.org/10.48550/arXiv.2506.21734.

Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Paul J. Werbos. Backpropagation through time: what it does and how to do it. *Proc. IEEE*, 78(10): 1550–1560, 1990. doi: 10.1109/5.58337. URL https://doi.org/10.1109/5.58337.

Xiao-Wen Yang, Zi-Yu Han, Xi-Hua Zhang, Wen-Da Wei, Jie-Jing Shao, Lan-Zhe Guo, and Yu-Feng Li. Stabilizing recurrent dynamics for test-time scalable latent reasoning in looped language models. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2605.26733.

Sai Zhang, Liangjia Zhu, and Yi Gao. An efficient deep equilibrium model for medical image segmentation. *Computers in Biology and Medicine*, 148:105831, 2022. doi: 10.1016/j.compbiomed.2022.105831.

A Implementation and Evaluation Details

A.1 LiteMB-UNet Architecture Details

LiteMB-UNet is the lightweight encoder/decoder backbone proposed in this work and shared by both our no-Refiner control and recurrent model. As shown in Figure 6, the BraTS configuration uses six encoder resolutions with channel widths $[32, 64, 128, 256, 320, 320]$. Capping the deepest width at 320, rather than continuing to double it, prevents low-resolution stages from dominating the parameter count. The decoder mirrors this hierarchy using nearest-neighbor upsampling, $1 \times 1 \times 1$ projections, concatenated encoder skips, and a single lightweight refinement block at each resolution; the final mask is produced by a $1 \times 1 \times 1$ prediction head.

The main spatial primitive is LiteMBConv3D. It first expands channels with a pointwise convolution, applies a $3 \times 3 \times 3$ depthwise convolution, uses ECA-3D for inexpensive channel reweighting, and projects back with another pointwise convolution, with a residual connection whenever the input and output shapes match. Thus, costly dense 3D spatial mixing is replaced by a depthwise operation, while channel mixing is confined to pointwise projections. Specifically, DWConv3D sets the number of groups to the expanded channel count C_h, so every channel is convolved independently by one $3 \times 3 \times 3$ kernel. For this spatial operation, the number of kernel weights decreases from $27C_h^2$ for a dense same-width 3D convolution to $27C_h$, with the same C_h-fold reduction in spatial multiply-add operations at a fixed feature-map size. ECA-3D adds only global average pooling and a short one-dimensional convolution. The three highest-resolution stages use only these local blocks; residual 3D shifted-MLP blocks are introduced at the three coarsest resolutions, where spatial tensors are small. They split channels into seven groups, shift them along the positive and negative depth, height, and width directions (plus one unshifted group), and combine a channel MLP with a depthwise 3D convolution. This allocation preserves cross-axis feature mixing without applying a heavy token mixer to the largest feature maps.

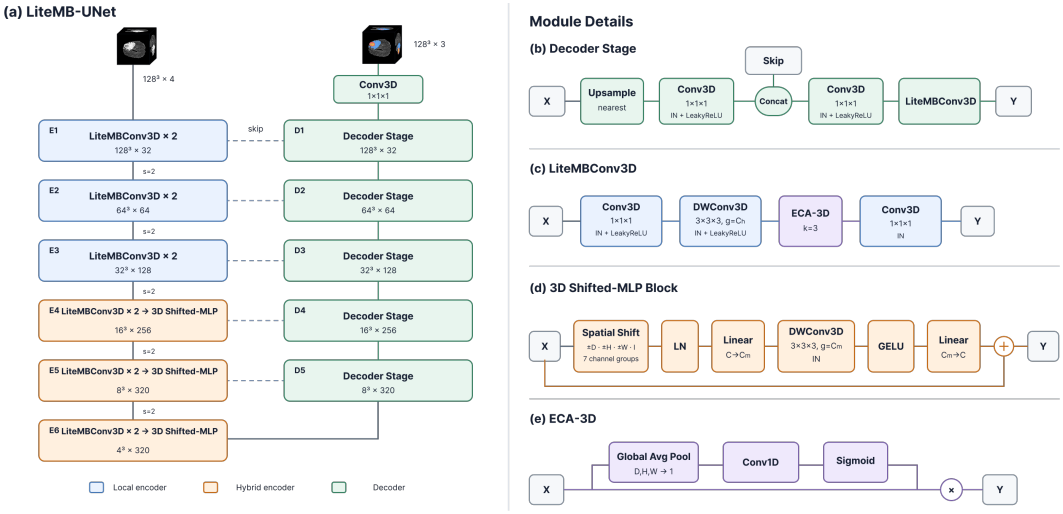


Figure 6: The proposed LiteMB-UNet backbone in the BraTS 3D configuration. **(a)** A six-resolution encoder/decoder with lightweight skip fusion. LiteMBConv3D blocks are used throughout, while shifted-MLP mixing is restricted to the three coarsest encoder resolutions. **(b)** Each decoder stage performs nearest-neighbor upsampling, pointwise projection, skip concatenation, and one LiteMBConv3D refinement. **(c)** LiteMBConv3D factorizes spatial and channel mixing into pointwise expansion, depthwise 3D convolution, ECA-3D, and pointwise projection; shape-compatible residual connections are omitted for clarity. **(d)** The residual 3D shifted-MLP block mixes seven directionally shifted channel groups and adds a lightweight depthwise spatial path. **(e)** ECA-3D obtains channel weights from global average pooled features using a short one-dimensional convolution.

A.2 Training Schedule and Inference Algorithms

Algorithm 1 BraTS training schedule (100 total epochs) and two inference modes. Conditioning on d_0, s, s_D is omitted inside local losses for readability; only the explicitly unfrozen parameters are updated.

Require: Training set $\mathcal{D}$; Prelude $\mathcal{P}_\phi$, shared Refiner R_θ, Coda $\mathcal{C}_\psi$; phase lengths $(50, 20, 10, 10, 10)$ epochs.
Multi-start bands $\mathcal{B} = \{[0,1],[2,4],[4,6]\}$ and two-step TBPTT window length $L = 2$.
$Q_1 = \mathcal{U}\{8,\dots,14\}$; $Q_2 = 0.6Q_1 + 0.4\mathcal{U}\{15,\dots,30\}$ (virtual start depth).
procedure VirtualDepthStep(x, y, Q, w_v)
    Compute $(d_0, s) \leftarrow \mathcal{P}_\phi(x)$ and the training rollout $d_{0:8}$ without gradients.
    For each band $b \in \mathcal{B}$, sample $\tau_b \in b$, detach d_{τ_b}, and take two gradient-tracked Refiner steps.
    $\mathcal{L}_{\mathrm{MS}} \leftarrow \frac{1}{|\mathcal{B}|}\sum_b \ell(\mathcal{C}(R^2(\mathrm{stopgrad}(d_{\tau_b}))), y)$.
    Sample $\rho \sim Q$; construct and detach $z_0 \approx d_\rho$ by geometric extrapolation followed by FiLM normalization.
    $z_2 \leftarrow R^2(z_0) \approx d_{\rho+2}$; $\mathcal{L}_{\mathrm{VD}} \leftarrow \ell(\mathcal{C}(z_2), y)$.
    Update only the shared correction core within R_θ with $(1 - w_v)\mathcal{L}_{\mathrm{MS}} + w_v\mathcal{L}_{\mathrm{VD}}$.
end procedure
Phase A (epochs 1 to 50): joint training.
Unfreeze $\mathcal{P}, R, \mathcal{C}$ (including the FiLM conditioning parameters).
for each minibatch (x, y) **do**
    Sample $K = 1$ with probability 0.6, otherwise $K = 2$.
    Compute $(d_0, s) \leftarrow \mathcal{P}_\phi(x)$ and $d_K = R^K(d_0; d_0, s)$; update (ϕ, θ, ψ) with $\ell(\mathcal{C}(d_K), y)$.
end for
Phase B_1 (epochs 51 to 70): Refiner optimization with fixed context and decoder.
Freeze all parameters; unfreeze only the shared Refiner core.
for each minibatch (x, y) **do**
    VirtualDepthStep($x, y, Q_1, w_v : 0.05 \to 0.20$)
end for
Phase C_1 (epochs 71 to 80): context and decoder realignment.
Unfreeze the Prelude, Coda, Refiner adapters, and FiLM; keep the shared Refiner core frozen.
for each minibatch (x, y) **do**
    Sample $K_{\mathrm{live}} \in \{1, 2\}$ with probabilities $(0.8, 0.2)$ and three depths $\mathcal{K}_d \subset \{4,\dots,8\}$.
    $\mathcal{L}_{C_1} \leftarrow 0.625\,\ell(\mathcal{C}(d_{K_{\mathrm{live}}}), y) + \frac{0.375}{3}\sum_{K\in\mathcal{K}_d}\ell(\mathcal{C}(d_K), y)$.
    Backpropagate the shallow term through its gradient-tracked trajectory; detach deep states and context for the deep Coda losses.
    Update the unfrozen surrounding-network parameters with $\mathcal{L}_{C_1}$.
end for
Phase B_2 (epochs 81 to 90): re-train the transition after C_1.
Freeze all parameters; unfreeze only the shared Refiner core.
for each minibatch (x, y) **do**
    VirtualDepthStep($x, y, Q_2, w_v : 0.20 \to 0.25$)
end for
Phase C_2 (epochs 91 to 100): final decoder-tail calibration.
Freeze all parameters; unfreeze only the Coda (decoder tail and prediction head).
for each minibatch (x, y) **do**
    Generate $d_{0:8}$ without gradients; update the unfrozen tail/head with $\frac{1}{5}\sum_{K=4}^{8}\ell(\mathcal{C}(d_K), y)$.
end for
Inference: input x, mode $m \in \{\mathrm{real}, \mathrm{analytic}\}$; depth $K \geq 0$ for real rollout.
$(d_0, s) \leftarrow \mathcal{P}_\phi(x)$. ▷ fixed checkpoint; no gradients
if $m = \mathrm{real}$ **then** ▷ Mode 1: real-rollout inference
    $d_K \leftarrow R_\theta^K(d_0; d_0, s)$. ▷ K real Refiner steps; $K = 0$ uses d_0
    return $\mathcal{C}_\psi(d_K, s_D)$.
else ▷ Mode 2: analytic inference
    Generate $d_{1:8}$ by $d_{t+1} = R_\theta(d_t; d_0, s)$; retain d_4, d_6, d_8.
    $u_6 \leftarrow d_6 - d_4$; $u_8 \leftarrow d_8 - d_6$.
    $q \leftarrow \mathrm{clip}\big(\langle u_8, u_6\rangle/(\|u_6\|_2^2 + \epsilon), 0, 0.95\big)$.
    $\hat{d}_\infty \leftarrow P_{d_0}\Big(d_8 + \frac{q}{1-q}\,u_8\Big)$. ▷ conditioned normalization
    return $\mathcal{C}_\psi(\hat{d}_\infty, s_D)$. ▷ no further Refiner steps
end if

A.3 Data Preprocessing and Training Hyperparameters

BraTS uses fold 0 of a five-fold split of the 295-case development cohort, yielding 236 training and 59 validation cases; the 74 test cases are held out separately. BraTS and LiTS use 128^3 patches, and DRIVE uses 512×512 image patches. BraTS modalities and DRIVE color channels are z-score normalized; LiTS uses percentile clipping and the dataset-level CT normalization.

Optimization settings. All three VD-Refine configurations use AdamW with a base learning rate of 10^{-4}, weight decay 10^{-5}, $(\beta_1, \beta_2) = (0.9, 0.999)$, and $\epsilon = 10^{-5}$. The training batch size is 2 for all three datasets. The segmentation objective combines cross-entropy and soft Dice with equal weights: BraTS uses binary cross-entropy over tumor regions and per-case Dice, while LiTS and DRIVE use multiclass cross-entropy and batch-pooled foreground Dice. Dice smoothing is 10^{-5}, and deep supervision is disabled. The residual step size is fixed at $\alpha = 0.5$; instance normalization uses $\epsilon = 10^{-5}$. Each epoch comprises 250 training minibatches and 50 validation minibatches. The 100-epoch schedule allocates $50/20/10/10/10$ epochs to $A/B_1/C_1/B_2/C_2$, respectively. Training uses bfloat16 autocasting, and gradients are clipped to a maximum ℓ_2 norm of 12.

Learning-rate schedule. Learning rates are constant within each phase. All groups use 10^{-4} in A; the shared Refiner core uses 10^{-4} and 2×10^{-5} in B_1 and B_2. In C_1, the encoder, decoder/adapters, FiLM, and prediction head use 5×10^{-6}, 10^{-5}, 10^{-5}, and 10^{-4}, respectively. In C_2, only the decoder tail and head are updated, at 5×10^{-6} and 5×10^{-5}. The datasets share the five-phase schedule, but differ in which parameters are updated during B. On BraTS, only the shared Refiner core is updated; the Prelude, Coda, Refiner adapters, and FiLM mapping remain frozen. For LiTS and DRIVE, the proposal-preserving B phases additionally update the encoder, decoder, and prediction head at 10^{-4} in B_1 and 5×10^{-5} in B_2, while FiLM remains frozen. These two datasets also use a $K = 0$ auxiliary loss with weight 0.8 in A; in B, the separate backbone objective combines a depth sampled from $\{1, 2, 4\}$ with a $K = 0$ auxiliary loss of weight 0.1.

A.4 Case-Paired Evaluation Protocol

The finite-state and continuation analyses use one prespecified patch from each of the 74 held-out BraTS cases for both the complete and no-VD checkpoints. Patch selection is fixed across conditions. State diagnostics use FP32 without TTA. The diagnostic implementation reproduces the native $K = 8$ logits exactly on the initial verification patch. Full-case Dice uses the complete test cohorts and the inference conventions in Section 4.1. BraTS and DRIVE use the checkpoint with the best exponentially averaged validation pseudo-Dice at $K = 8$; LiTS uses the final checkpoint. DRIVE results use training seed 2029. Each checkpoint is held fixed across inference modes and depths. For these case-paired analyses, 95% confidence intervals (CIs) use 10,000 bootstrap resamples of the held-out test cases, preserving pairing across conditions. Multiple regions or states from one case are not treated as independent cases.

B Additional Segmentation Results

B.1 Parameter Efficiency and Cross-Dataset Accuracy

The LiteMB-UNet backbone has 4.12M trainable parameters for the BraTS no-Refiner backbone. Adding the FiLM-conditioned shared Refiner increases the model to 4.40M parameters (approximately 0.28M additional parameters). The Refiner weights are shared across iterations, so increasing K changes computation without adding parameters. Parameter efficiency combines the lightweight backbone with recurrent weight sharing.

Figure 7 compares the two VD-Refine inference modes with MedNeXt-S across datasets. Parameter counts refer to the BraTS configuration; Dice gains use each dataset's full test cohort. The two VD-Refine modes share a checkpoint within each run. The plotted gains are differences between the mean Dice values reported in Table 1, rather than significance estimates.

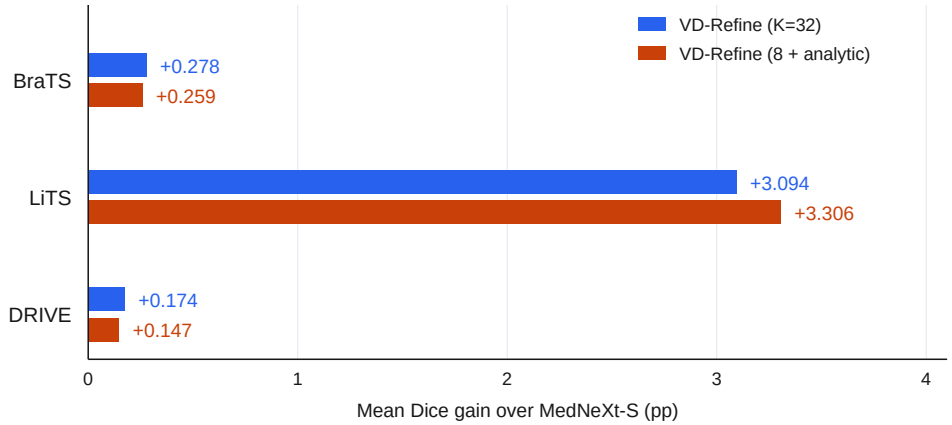


Figure 7: **Cross-dataset accuracy gains over MedNeXt-S.** Mean Dice differences use the full held-out cohorts: 74 BraTS volumes, 26 LiTS volumes, and 20 DRIVE images. Gains are computed directly from Table 1, using its case-averaged BraTS scores and the reported LiTS and DRIVE scores. Both VD-Refine modes share a checkpoint within each run. In the BraTS configuration, VD-Refine uses 4.40M parameters versus MedNeXt-S's 5.55M, a 20.7% reduction (Table 1).

B.2 Complete BraTS Ablation Results

Table 3 gives regional Dice and early-depth gains for all configurations in Table 2. All values use the same 74 held-out volumes; differences are computed within each checkpoint from unrounded scores. Tail TBPTT-2 samples $K \sim \mathcal{U}\{1, \ldots, 8\}$ and uses at most two gradient-tracked steps. Alternating denotes the five-stage schedule. Plain InstanceNorm and FiLM differ in whether the affine parameters are conditioned on d_0.

Table 3: **Complete quantitative BraTS ablations.** IDs identify the training configurations in Table 2. WT, TC, ET, and mDice are evaluated on 74 held-out volumes at $K = 32$. $\Delta_{0\to8}$ and $\Delta_{8\to32}$ are within-checkpoint mean Dice changes between the indicated depths, in percentage points. Bold marks the highest value in each column.

ID	WT	TC	ET	mDice	$\Delta_{0\to8}$	$\Delta_{8\to32}$
A1	90.73	85.14	77.12	84.33	+1.23	-0.36
A2	90.44	83.76	77.41	83.89	+1.21	-0.34
A3	90.60	85.52	76.71	84.34	+1.15	-0.29
A4	90.35	85.18	**78.14**	84.57	+1.16	-0.14
A5	89.75	83.68	77.37	83.65	+1.35	-0.80
A6	90.68	84.99	77.40	84.37	**+2.01**	-0.55
A7	90.97	86.47	76.76	84.73	+1.33	-0.16
A8	**91.28**	86.66	75.42	84.45	+1.99	+0.05
A9	91.20	86.18	76.81	84.73	+1.96	-0.14
A10	91.26	**87.06**	77.30	**85.26**	+1.97	**+0.07**

B.3 Case-Level Reliability at Greater Depth

Table 4 summarizes D_i from Section 4.3; Figure 9 displays its distribution. The complete BraTS model reduces both average and tail degradation compared with the no-VD checkpoint. The paired mean difference (VD-Refine minus no VD) is -0.177 pp (95% CI $[-0.318, -0.062]$). LiTS has a larger tail, illustrating the patient-level variation behind its average depth curve.

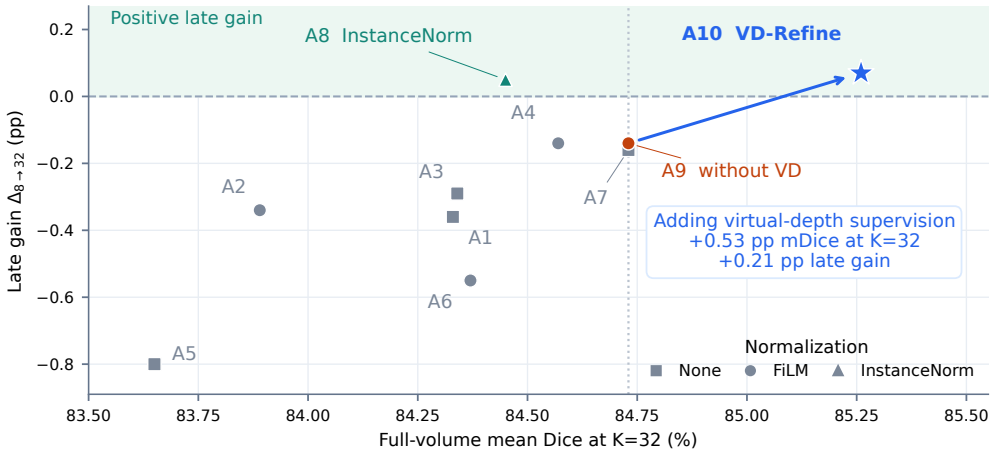


Figure 8: **Accuracy and late-depth gain for the configurations in Table 2.** The arrow connects the FiLM model without VD (A9) to the full model (A10), isolating the addition of virtual-depth supervision. Moving right indicates higher $K = 32$ mean Dice; moving up indicates a larger gain from $K = 8$ to 32. Horizontal and vertical lines mark zero late gain and the no-Refiner backbone's accuracy, respectively. VD improves both measures.

Table 4: Case-level late degradation in Dice percentage points (lower is better). For each case, degradation is the maximum drop from $K = 8$ to $K \in \{16, 32\}$, clipped at zero. N is the number of held-out cases. The interval is a 95% case-bootstrap CI for the mean; the final columns give the empirical 95th percentile and maximum. LiTS and DRIVE use VD-Refine; the BraTS comparison isolates VD.

Model / dataset	N	Mean	95% CI	95th pct.	Maximum
BraTS: VD-Refine	74	0.057	[0.027, 0.093]	0.306	0.733
BraTS w/o VD	74	0.234	[0.114, 0.378]	1.426	3.010
LiTS	26	0.673	[0.233, 1.312]	2.411	7.153
DRIVE	20	0.005	[0.000, 0.015]	0.013	0.092

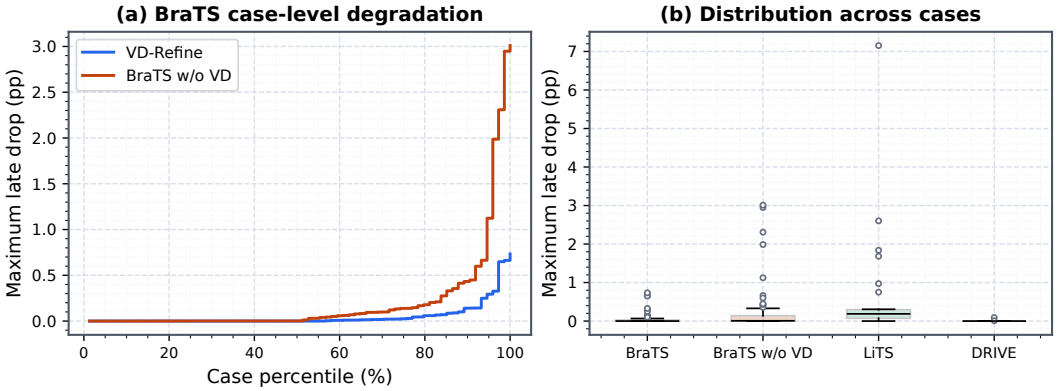


Figure 9: Case-level late degradation. (a) Empirical case-percentile curves on BraTS. (b) Case distributions across datasets and the BraTS no-VD checkpoint. Values are the nonnegative maximum drop from $K = 8$ to $K \in \{16, 32\}$. Boxes span the interquartile range, whiskers extend to 1.5 times that range, and circles show remaining cases. Cohorts contain 74 BraTS, 26 LiTS, and 20 DRIVE cases; lower values indicate better preservation of accuracy. VD reduces both mean and tail degradation on BraTS.

B.4 Paired Accuracy Comparisons Between Inference Modes

Table 5 gives paired complete-case Dice differences. On DRIVE, analytic inference improves over eight real steps by 0.341 pp (95% CI $[0.225, 0.487]$); all other paired intervals include zero. Table 8 reports the corresponding inference costs. For the BraTS no-VD checkpoint, analytic inference yields 84.804% Dice versus 84.866% for the eight-step baseline. The full model's analytic-minus-eight-step gain exceeds the no-VD gain by 0.108 pp. Evaluating all 74 no-VD eight-step cases on the same A100 environment used for analytic predictions yields a paired gain difference of $+0.108$ pp (95% CI $[0.033, 0.197]$). This comparison uses matched hardware within each checkpoint. All latency plots use H100 measurements.

Table 5: Paired full-case Dice differences (pp) for one fixed checkpoint per dataset. Analytic inference decodes a geometric state estimate after eight real steps; real 8 and real 32 decode the corresponding real rollouts. Positive differences favor analytic inference. N is the test-cohort size; intervals are 95% paired case-bootstrap CIs for the mean difference.

Dataset	Comparison	N	Mean	95% CI
BraTS	Analytic − real 8	74	+0.046	[-0.006, +0.108]
BraTS	Analytic − real 32	74	-0.020	[-0.086, +0.051]
LiTS	Analytic − real 8	26	+0.101	[-0.136, +0.508]
LiTS	Analytic − real 32	26	+0.212	[-0.544, +1.017]
DRIVE	Analytic − real 8	20	+0.341	[+0.225, +0.487]
DRIVE	Analytic − real 32	20	-0.011	[-0.074, +0.049]

C Mechanism Analysis

This section complements Section 4.4 with local trajectory geometry, perturbation recovery, training-stage progression, gradient diversity, and analyses of virtual training states and the extrapolated inference state. The case-paired protocol is described in Appendix A.4.

C.1 Conditioned Normalization Across Depth

Figure 10 reports the complete scale, update, and prediction-change diagnostics underlying the conditioned-normalization results in Section 4.4. These long-rollout measurements complement the local geometry and perturbation-recovery analyses in Figure 4.

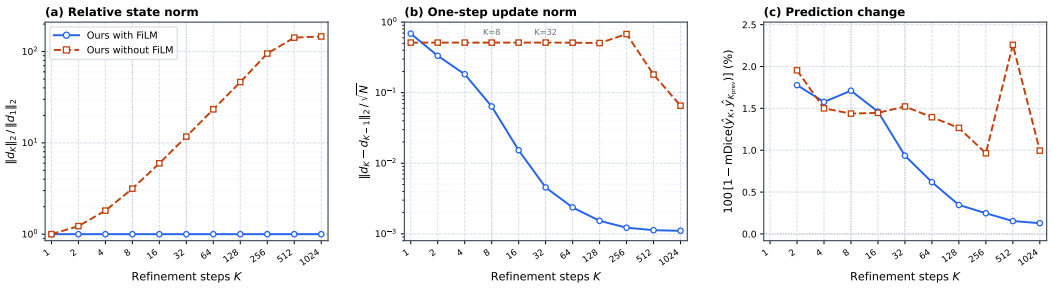


Figure 10: Effect of FiLM normalization on recurrent dynamics. The alternating multi-start models with FiLM (A9) and without normalization (A7) both omit VD and are evaluated on held-out BraTS patches. The summaries use 500 patches per sampled depth and show the available real-rollout measurements through $K = 1024$. **(a)** FiLM keeps $\|d_K\|_2/\|d_1\|_2$ near one, whereas the unnormalized state grows by roughly $146\times$ over the tested range. **(b)** With FiLM, the per-element update $\|d_K - d_{K-1}\|_2/\sqrt{CDHW}$ decreases by nearly three orders of magnitude; without FiLM it remains nearly constant through $K = 256$. **(c)** The prediction change $100[1 - \mathrm{mDice}(\hat{y}_K, \hat{y}_{K^-})]$, where K^- is the preceding sampled depth, progressively vanishes with FiLM but remains unstable without it.

C.2 Local Trajectory Geometry

Figure 4a,b visualizes both the relative one-step update magnitude $100\|R(d) - d\|_2/\|d\|_2$ and decoded Dice in the same local feature plane for the illustrative held-out patch `BraTS20_Training_086_155`. The first two principal components of the sampled clean and perturbed trajectories explain 76.52% and 11.29% of their variance, respectively. The background evaluates reconstructed states on a 31×31 grid in this plane; the overlaid curves are projections of the real high-dimensional trajectories at $K \in \{0, 1, 2, 4, 8, 16, 32\}$. Thus, a background value at a projected trajectory coordinate need not equal the corresponding real state's measurement. The clean rollout's Dice increases from 35.15% at $K = 0$ to 64.93% at $K = 32$. This single-patch visualization illustrates local refinement behavior rather than establishing global convergence or isolating the effect of VD. The separate case-paired perturbation diagnostic below quantifies feature recovery across held-out cases.

C.3 Recovery from Feature Perturbations

To complement the illustrative landscape in Figure 4a,b, we quantify local perturbation recovery on one prespecified patch from each of the 74 held-out BraTS cases. At $K \in \{8, 32\}$, we inject spatially centered Gaussian noise into the recurrent state and continue both the clean and perturbed trajectories for 64 real updates, using two paired random directions per case. Let $\widetilde{d}_{K+t}$ and d_{K+t} denote the perturbed and continuing clean states. We measure the remaining perturbation $q_t = 100\|\widetilde{d}_{K+t} - d_{K+t}\|_2/\|\widetilde{d}_K - d_K\|_2$ and the paired task difference $g_t = 100[\mathrm{mDice}(\mathcal{C}(\widetilde{d}_{K+t}), y) - \mathrm{mDice}(\mathcal{C}(d_{K+t}), y)]$, excluding undefined regions from each paired Dice comparison. Table 6 reports both raw and channel-moment-restored perturbations at amplitude 0.3, the larger of the two tested amplitudes (0.1 and 0.3); Figure 4c displays the moment-restored feature-recovery results. After moment restoration, VD-Refine retains 1.207% and 0.460% of the injected feature distance at $K = 8$ and $K = 32$, respectively; without virtual-depth supervision, the corresponding residuals are 4.424% and 1.931%. Removing FiLM as well increases them to 169.025% and 117.220%. Unnormalized perturbations show the same ordering. Both FiLM-equipped models approach their clean trajectories, with smaller residuals for VD-Refine. Mean initial Dice changes are within 0.011 pp in magnitude. The experiment therefore measures feature recovery from perturbations that initially have little effect on segmentation.

C.4 Training-Stage Progression

We evaluate the checkpoint after every phase of the $A \to B_1 \to C_1 \to B_2 \to C_2$ schedule on the same 500 held-out BraTS patches and sweep the real rollout depth from $K = 1$ to $K = 32$. Figure 11 shows that the initial joint phase degrades outside its shallow training range, whereas the first Refiner-only phase (B_1) produces the dominant and increasingly large gain at deeper K. Realignment in C_1 adds a smaller, approximately depth-uniform improvement, while the second alternation yields diminishing returns. These checkpoints track the combined progression of optimization, parameter freezing, and depth supervision.

C.5 Gradient Diversity Across Depths

Using the checkpoint after the initial joint phase, we analyze 500 held-out BraTS patches sampled with the training loader's foreground-oversampling policy. For each patch, we first compute a gradient-free reference trajectory to d_8 and then independently backpropagate through the two-step windows $d_1 \to d_3$, $d_3 \to d_5$, and $d_6 \to d_8$. The resulting Refiner gradients retain a dominant common direction, but the early and late pair is less aligned than neighboring pairs: its median cosine similarity is 0.887, compared with 0.962 and 0.966. Figure 12 shows that the windows provide compatible gradients with a measurable depth-dependent component.

C.6 Virtual-State Accuracy Across Depths

For finite depths, we write $\hat{d}_\rho = P_{d_0}(\tilde{d}_\rho)$ for the normalized geometric estimate in Eq. 4. For finite starts, the linear comparator is $P_{d_0}[d_8 + \frac{\rho-8}{2}(d_8 - d_6)]$, and the copy comparator reuses d_8. Each

Table 6: Local perturbation recovery on one fixed patch from each of 74 BraTS cases. Spatially centered Gaussian noise has norm 0.3 times the centered state norm at depth K. Moment restoration rescales the perturbed state to the clean channel means and standard deviations. q_{64} is the feature distance after 64 updates as a percentage of the injected distance, with 95% case-bootstrap intervals; lower values indicate stronger recovery, and 100% means no reduction. g_0 and g_{64} are perturbed-minus-clean Dice differences (pp); both trajectories continue to the same depth. Evaluation uses fixed best checkpoints, FP32, and no TTA. Two noise directions are averaged per case before 10,000 bootstrap resamples.

Model	FiLM	VD	K	q_{64} (%) [95% CI]	g_0 (pp)	g_{64} (pp)
Raw perturbation						
VD-Refine	Yes	Yes	8	1.168 [0.736, 1.648]	+0.0047	+0.0014
Without VD	Yes	No	8	4.215 [3.436, 5.038]	+0.0020	+0.0022
Without VD or FiLM	No	No	8	167.187 [165.016, 169.821]	-0.0026	+0.0025
VD-Refine	Yes	Yes	32	0.439 [0.325, 0.564]	+0.0011	+0.0011
Without VD	Yes	No	32	1.849 [1.466, 2.273]	-0.0023	-0.0037
Without VD or FiLM	No	No	32	116.399 [116.023, 116.843]	-0.0007	+0.0136
Moment-restored perturbation						
VD-Refine	Yes	Yes	8	1.207 [0.764, 1.701]	-0.0016	+0.0012
Without VD	Yes	No	8	4.424 [3.604, 5.292]	-0.0022	+0.0014
Without VD or FiLM	No	No	8	169.025 [166.777, 171.769]	-0.0032	-0.0050
VD-Refine	Yes	Yes	32	0.460 [0.340, 0.592]	-0.0092	+0.0012
Without VD	Yes	No	32	1.931 [1.528, 2.376]	-0.0106	-0.0027
Without VD or FiLM	No	No	32	117.220 [116.823, 117.689]	-0.0006	+0.0061

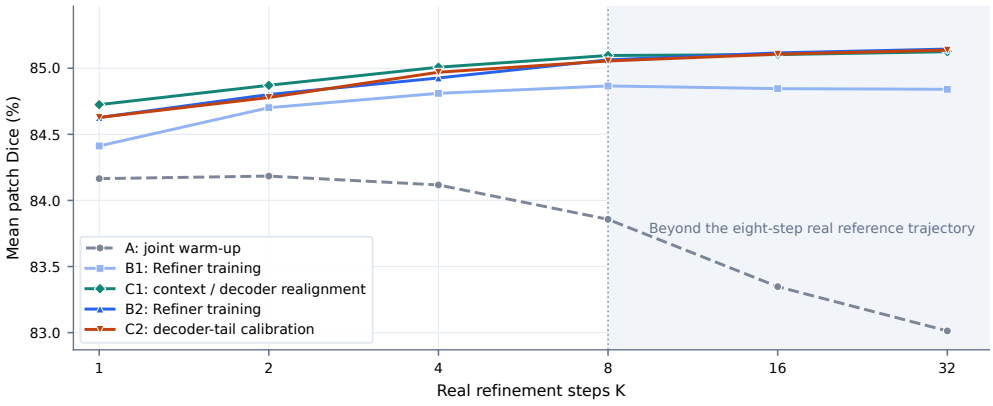


Figure 11: **Accuracy across the five training phases.** The checkpoints after A, B_1, C_1, B_2, C_2 are evaluated on the same 500 held-out BraTS patches over $K \in \{1, 2, 4, 8, 16, 32\}$. Shading marks depths beyond the eight-step training rollout. The largest deep-refinement improvement occurs after B_1; subsequent phases retain or modestly improve this gain. B_2 has the highest patch Dice at $K = 32$. This patch diagnostic is separate from the 74-case full-volume ablations in Table 2 and Figure 8.

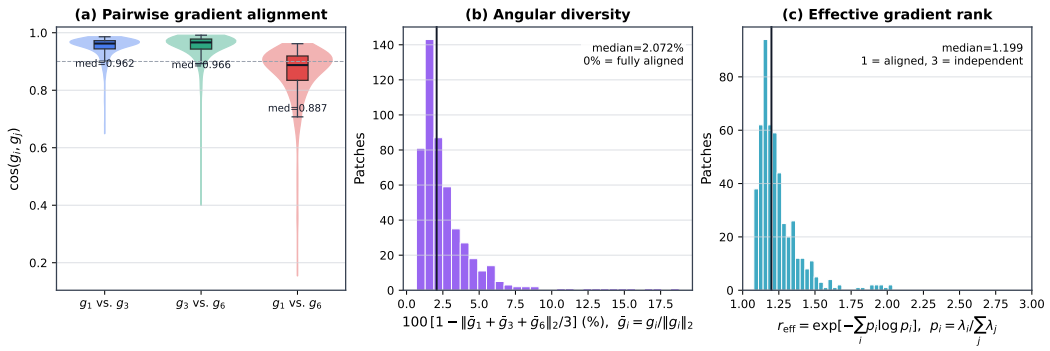


Figure 12: Gradient diversity across three multi-start windows on 500 held-out BraTS patches sampled with training-time foreground oversampling at the checkpoint after the initial joint phase. The gradients g_1, g_3, g_6 come from two-step windows starting at d_1, d_3, d_6, respectively. **(a)** Pairwise cosine similarity; values near one indicate alignment. **(b)** Angular diversity of normalized gradients; zero indicates complete alignment. **(c)** Effective rank of their normalized Gram matrix; one indicates a single shared direction and three indicates equally represented orthogonal directions. The gradient directions are strongly aligned while retaining a measurable depth-dependent component.

is compared with the real $d_{\rho+j}$ after $j \in \{0, 1, 2\}$ real corrections. We also evaluate geometric extrapolation at the B_1 and B_2 checkpoints from the same training run.

The 500-patch analysis in Figure 13 samples additional target depths and complements the case-paired comparison in Figure 5.

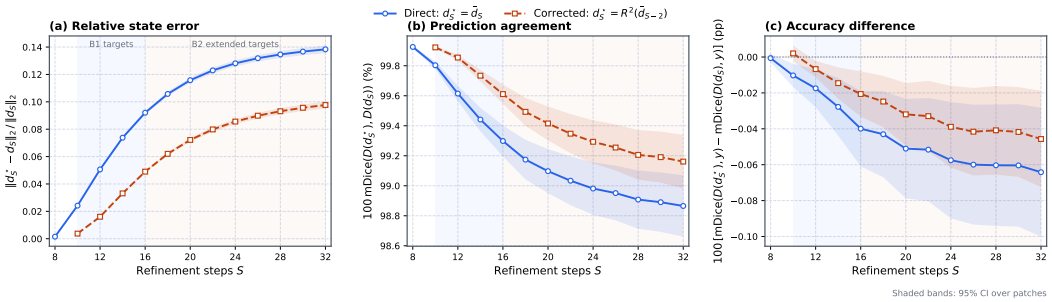


Figure 13: Fidelity of virtual extrapolation to a matched real rollout on 500 held-out BraTS patches. Both curves compare an approximate endpoint d'_S with the real state d_S: Direct uses $d'_S = \widehat{d}_S$, while Corrected uses $d'_S = R^2_\theta(\widehat{d}_{S-2})$. **(a)** Relative feature error $\|d'_S - d_S\|_2/\|d_S\|_2$ (lower is better). **(b)** Prediction agreement $100\,\mathrm{mDice}(\mathcal{C}(d'_S), \mathcal{C}(d_S))$ (higher is better). **(c)** Approximate-minus-real Dice difference (pp); zero indicates equal task accuracy. Two real corrections improve feature and prediction agreement. Curve-centered bands are 95% patch-level confidence intervals, which do not account for dependence among patches from the same case; the pale background marks the two virtual-depth ranges used during training (10 to 16 and 17 to 32; the background shading is schematic).

C.7 Anchor Sensitivity of Geometric Extrapolation

Analytic inference in Figure 3 constructs $\hat{d}_\infty$ from the real states d_4, d_6, d_8; its fidelity depends on how well these anchors identify the local decay rate. We test a cheaper alternative that requires only the first four real steps: the consecutive one-step anchors d_2, d_3, d_4, with $u_3 = d_3 - d_2$ and $u_4 = d_4 - d_3$, using the same coefficient clipping and P_{d_0} normalization before direct decoding. Because these anchors are one step apart rather than two, their coefficient estimates a *per-step* decay and must be compared with the per-step decay implied by the two-step fit, $\sqrt{q_{468}} \approx 0.746$, rather

than with q_{468} itself. Table 7 reports both spacings under the standard full-volume protocol on the 74 held-out BraTS cases, where the real $K = 32$ reference reproduces the per-case scores used in the main evaluation. The early anchors retain a high cosine similarity to the real d_{32}, but have a larger relative feature error and a smaller fitted per-step decay coefficient ($q_{234} \approx 0.713$). Their mDice is 0.12 pp below the real rollout, compared with 0.02 pp for the eight-step anchors. Truncating the early-anchor series at $K = 32$ rather than at the limit of its fitted series changes mDice by less than 10^{-5} on the 0–1 scale, consistent with the small fitted tail ($q^{28} \approx 7 \times 10^{-5}$). The anchor comparison holds the checkpoint and inference protocol fixed.

Table 7: Sensitivity to anchor selection and real-prefix length on 74 held-out BraTS volumes. Both estimates use a clipped per-sample decay coefficient, conditional normalization, and direct decoding. Feature errors and Dice differences are measured relative to the real $K = 32$ rollout of the same checkpoint; higher cosine similarity and lower relative ℓ_2 error indicate closer feature agreement. Δ reports the mean Dice difference in pp with a 95% case-bootstrap interval. The per-step decay is q_{234} for consecutive anchors and $\sqrt{q_{468}}$ for two-step anchors. Evaluation uses FP32, 128^3 patches, tile step 0.5, Gaussian weighting, and eight-way mirror TTA.

Anchors	Real prefix steps	cos vs real d_{32}	rel. ℓ_2	Per-step decay	mDice	Δ vs real K=32 (pp) [95% CI]
Real $K = 32$ rollout (reference)	32	1.000	0.000	—	85.26	—
d_4, d_6, d_8 (used in training)	8	0.989	0.143	0.746	85.24	$-0.02\ [-0.09, +0.05]$
d_2, d_3, d_4	4	0.960	0.282	0.713	85.14	$-0.12\ [-0.23, -0.02]$

C.8 Further Refinement after Analytic Inference

For starts d_8, d_{32}, and $\hat{d}_\infty$, we execute $j \in \{0, 1, 2, 8, 32, 64\}$ additional real updates. Figure 14 shows the relative one-step residual $\|R(z) - z\|_2/\|z\|_2$, drift from the starting state, and decoded Dice change. For VD-Refine, the initial residual at $\hat{d}_\infty$ is 0.02945, versus 0.04139 at d_8 and 0.00189 at d_{32}. After 64 updates its relative drift is 0.1513, while its mean patch Dice change is $+0.073$ pp (95% CI $[-0.021, 0.182]$). Further refinement changes the feature substantially while producing a small mean change in decoded Dice.

At the finite reference d_{1024}, the maximum one-step residual over the final eight measured transitions is at most 10^{-4} for 74/74 full-model patches and 69/74 no-VD patches. These residuals characterize the finite real trajectories. The extrapolated state $\hat{d}_\infty$ is obtained independently from the limit of the fitted geometric series.

D Runtime Evaluation

D.1 Independent-Forward Timing

The patch-latency experiment uses batch size one and FP16 on H100, with 20 warm-ups and 60 randomized independent full-forward measurements per mode. Prefixes are not cached between modes; the Prelude and one final Coda are included. This protocol measures latency across inference depths. The VD-Refine and no-Refiner rows in Table 1 instead use FP16 with 50 warm-ups and 200 timed forwards, reporting mean $\pm$ standard deviation. Patch latency increases approximately linearly with recurrent depth: fitted R^2 is above 0.998 for all three datasets.

D.2 Complete-Case Timing

The complete-case experiment uses 10 cases per dataset, chosen in advance by filename-hash order. DRIVE accuracy uses seed 2029; both timing experiments retain the original checkpoint with the same architecture and inference settings. Each mode independently reads, preprocesses, predicts, resamples, and exports each case, excluding model loading and Dice evaluation. A case/mode is timed once. For LiTS, all seven modes use the same 10 cases in one serial process on a single H100, with a deterministic randomized mode order within each case. OS file caches are not forced cold. Accuracy is evaluated on the full test cohort independently of the timing subset. Table 8 and Figure 15 show why reduced network time has a smaller effect when data processing dominates.

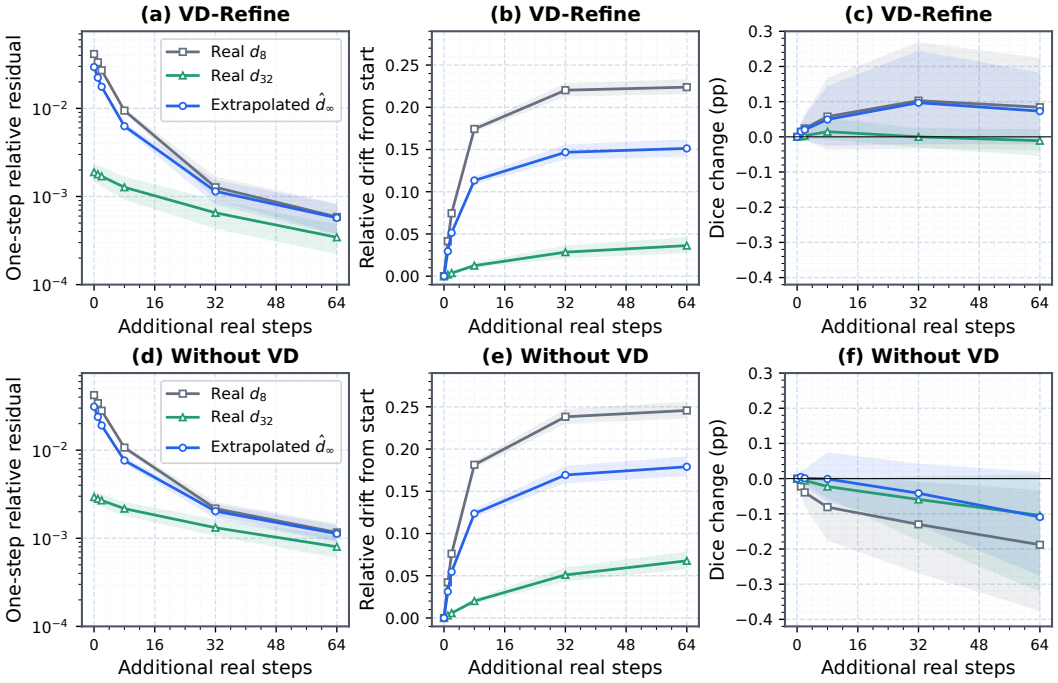


Figure 14: Further refinement after real-rollout or analytic inference. Rows compare VD-Refine and the no-VD checkpoint on one fixed patch from each of 74 BraTS cases. Curves start from d_8, d_{32}, or the extrapolated state $\hat{d}_\infty$; the horizontal axis counts additional real updates. Columns show relative one-step residual (lower indicates a smaller next update), feature drift relative to each curve's starting state, and Dice change from its own starting prediction (pp). The full model retains similar mean Dice while its features continue to evolve. Bands are 95% case-bootstrap CIs; the residual axis is logarithmic.

Table 8: Mean complete-case time in seconds on an H100 over 10 prespecified cases per dataset. Total time includes reading, preprocessing, inference, resampling, and export, excluding model loading and Dice evaluation. The mode 8 + analytic denotes analytic inference: eight real steps followed by geometric extrapolation and direct decoding. Intervals reflect case variation, not repeated-run timing noise.

Dataset	Mode	Total time	95% CI	Inference only
BraTS	Real 8	4.00	[3.94, 4.08]	3.31
BraTS	Real 32	4.88	[4.82, 4.95]	4.19
BraTS	8 + analytic	4.01	[3.95, 4.10]	3.33
LiTS	Real 8	117.82	[84.74, 153.70]	19.88
LiTS	Real 32	122.44	[88.79, 158.77]	24.98
LiTS	8 + analytic	116.94	[84.27, 152.42]	19.92
DRIVE	Real 8	0.52	[0.50, 0.57]	0.45
DRIVE	Real 32	0.87	[0.86, 0.87]	0.79
DRIVE	8 + analytic	0.50	[0.50, 0.51]	0.43

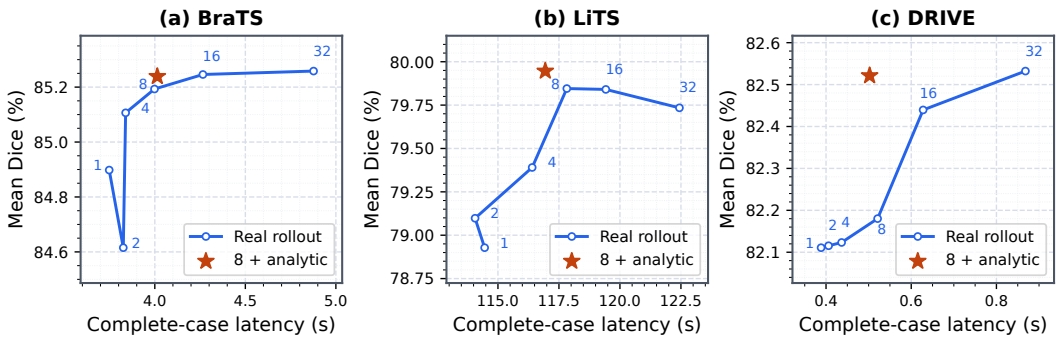


Figure 15: Accuracy–latency trade-offs. Dice uses the full test cohorts (74 BraTS, 26 LiTS, and 20 DRIVE cases); latency is averaged over a fixed subset of 10 cases per dataset, shared across modes. Blue labels give the number of real Refiner steps; stars denote analytic inference (8 + analytic). DRIVE combines seed-2029 accuracy with the original same-architecture timing benchmark. Timing uses an H100 and includes CPU preprocessing and output export in addition to network inference. Analytic inference offers mean Dice close to 32 real steps at lower measured latency.

E Qualitative Segmentation Examples

Figure 16 shows tumor-core predictions at six recurrent depths for two BraTS examples. The overlays reveal spatial changes in segmentation errors, complementing the cohort-level depth-scaling results in Section 4.3. The displayed Dice scores also show that improvements need not be monotonic at every depth.

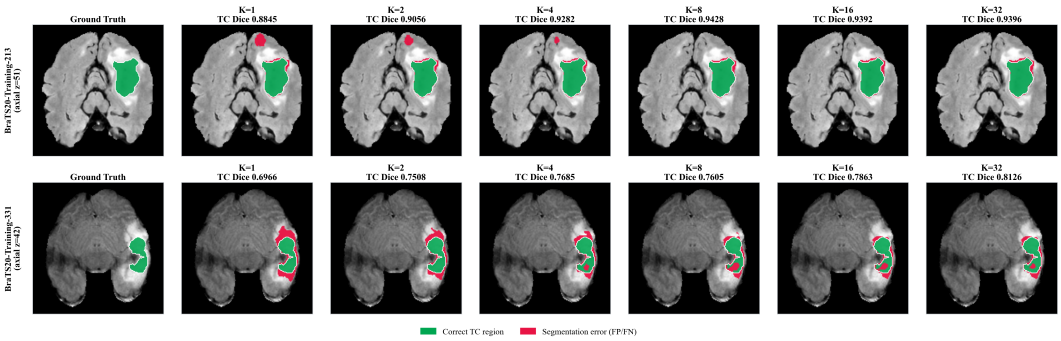


Figure 16: **Qualitative tumor-core segmentation across recurrent depths.** Rows show axial slices from BraTS20-Training-213 ($z = 51$) and BraTS20-Training-331 ($z = 42$). Columns show the ground truth and predictions at $K = 1, 2, 4, 8, 16, 32$; the displayed TC Dice values are computed on the shown axial slices, on a 0–1 scale. Green marks correctly segmented tumor-core regions, and red marks false-positive or false-negative regions. Refinement reduces visible false-positive regions and boundary errors, with small fluctuations across depths. These slice-level examples complement the full-cohort accuracy and reliability results.