COMPRESSING HISTORY INTO MEMORY: DISTILLING TRANSFORMERS INTO RECURRENT TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers are AI's workhorse but their computational cost becomes prohibitive when processing long sequences. We target long-horizon streaming vision and robotics applications, where it is particularly impractical to store and maintain a history of observations. Recurrent Transformers address this limitation by maintaining fixed-size memory but their performance lags behind that of transformers operating over the full observation history. We argue that this gap does not stem from architectural limitations, but from differences in how these models learn to compress past information. Without access to an observation history, recurrent models must explicitly decide what to retain in memory at each step, a significantly harder learning problem. In this work, we propose a distillation approach that transfers the compression strategy of a classical full-history transformer to a recurrent variant. We enable this by designing a teacher model that explicitly compresses its observation history into a fixed-size bottleneck representation and directly supervise the student's memory with this bottleneck representation, effectively aligning the two compression mechanisms. We show that this approach allows to train a recurrent latent robotic memory with linear-time complexity on the Mem-RPE task while substantially narrowing the performance gap to full-history transformers. We additionally validate the same principle on streaming visual question answering (VQA) and observe improved recurrent predictions thanks to memory distillation.

1 INTRODUCTION

Transformers (Vaswani et al., 2017) successfully emerged as arguably the most general model architecture in Machine Learning (ML) history and have been successfully applied in NLP (Brown et al., 2020), computer vision (Dosovitskiy et al., 2021), speech (Radford et al., 2023), robotics (Black et al., 2025), physics (Janny et al., 2023; Xu et al., 2025) and more. Their quadratic complexity $O(T^2)$ w.r.t. the number of input embeddings T has posed some stringent problems for domains like robotics, where attending to a rapidly growing number of observations can become impossible, *cf*. Figure 1 (left). While it is possible to ignore this in applications like manipulation, where often past observations beyond a recent cut-off point are ignored (treating them as fully observable problems), this is not an option for applications where the full past needs to be remembered. Examples are 3D reconstruction (Wang et al., 2024; 2025a), SLAM (Maggio & Carlone, 2026), navigation towards previously observed goals (Sariyildiz et al., 2025a) and generally any application where agents need to be able to recall any previously observed information. Our main target application is memory-based pose estimation, where an agent is asked to situate a visual query in the explored environment and with respect to its own moving coordinate frame: "*Where is this thing with respect to your current position?*" Answering this question requires information beyond the current observation.

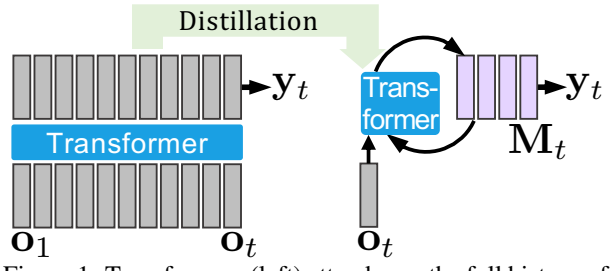


Figure 1: Transformers (left) attend over the full history of observations with complexity $O(T^2)$ at each step t. Recurrent transformers (right) maintain memory and take decisions with $O(1)$. We distill "*History Transformers*" into recurrent transformers, combining prediction performance and efficiency.

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.

Recurrent models like LSTMs (Hochreiter & Schmidhuber, 1997) and GRUs (Cho et al., 2014) have potentially unlimited context length and are restricted only by the size of their memory, the hidden state. However, while they initially were the models of choice in NLP, they famously lacked scaling capabilities and were largely replaced by transformers. Recurrent models have recently made a comeback by replacing the flat vectorial hidden state with a memory bank and performing the update with a transformer (van Steenkiste et al., 2024; Sariyildiz et al., 2025a; Ryoo et al., 2023), *cf*. Figure 1 (right). Their complexity is $O(1)$ at each step t with respect to the number of input observations (here we ignore complexity w.r.t. memory size, which is constant over time). This solves several issues by decoupling memory size from network capacity, allowing them to scale. In this work, we study the differences between *Recurrent Transformers* and their classical counterparts, which for the sake of clarity we name *History Transformers*, under the lens of their underlying *compression mechanisms*.

Recurrent Transformers (RTs) maintain some form of memory $\mathbf{M}_t$ and at each time step t have access to the current observation $\mathbf{o}_t$ only. Memory is limited and they need to decide which information from $\mathbf{o}_t$ needs to be retained. Information not stored in $\mathbf{M}_t$ is lost forever.

History Transformers (HTs) do not have these shortcomings. At each time step t they can access the full observation history and make use of it freely. Their drawback is the requirement to store the observation history and to attend to it.

Both models must compress past observations into representations useful for future predictions. However, they do so in fundamentally different ways. HTs implicitly compress history when predictions are needed: attention dynamically selects relevant past tokens, and gradients can supervise all past representations directly. In contrast, RTs must explicitly decide at each time step what information to store in their fixed-size memory without having access to future requirements for prediction. Information not written to memory is irreversibly discarded, and supervision of these memory updates is indirect and delayed. As a result, learning effective memory representations becomes a significantly harder optimization problem.

We propose to bridge this gap by turning memory learning into a supervised compression task. We consider a History Transformer augmented with a fixed-size bottleneck representation that explicitly compresses its observation history, similar to the Perceiver Resampler architecture (Jaegle et al., 2021). We then distill this bottleneck representation into the memory of a Recurrent Transformer, directly aligning the student's memory with the teacher's learned compression. In this way, we transfer the compression strategy itself.

We validate our approach, called *Chimera* for **C**ompressing **H**istory **I**nto **Me**mo**r**y with Tr**a**nsformers, primarily on Mem-RPE. This task provides a detailed study of long-horizon memory, including comparisons with existing recurrent models, generalization to longer sequences, ablations of the distillation procedure, and analysis of the learned representations. Distillation substantially narrows the gap to the full-history teacher while preserving the student's inference architecture and memory budget. The student can even outperform the teacher on long sequences. Our approach is further validated on streaming VQA with a frozen LongVA-7B backbone (Zhang et al., 2025a).

2 Related work

Dealing with long context: Causal models and KV-caching have decreased computational complexity, as the attention distributions for past tokens do not need to be recomputed at each time step. Linear formulations of attention lead to significant speed-ups but lack expressivity (Katharopoulos et al., 2020). The *Markovian Thinker* proceeds in chunks, each starting with a carry-over from the past history (Aghajohari et al., 2026). This strategy has been applied to 3D reconstruction (*cf*. below).

Recurrent models initially had vectorial hidden states *e.g*. *LSTM* (Hochreiter & Schmidhuber, 1997) and *GRU* (Cho et al., 2014). Their memory scaling problem was addressed by external neural memory (Graves et al., 2014; Sukhbaatar et al., 2015; Kaiser & Sutskever, 2016). Recurrence is currently re-emerging, for instance through state space models like *S4* (Gu et al., 2022), *Mamba* (Gu & Dao, 2024) and *LRU* (Orvieto et al., 2023) inspired by control theory. *xLSTM* adds QKV-attention for updates (Beck et al., 2024), and full-fledged Transformers handling a memory bank are proposed by *MooG* (van Steenkiste et al., 2024), *TTM* (Ryoo et al., 2023) and *Kinaema* (Sariyildiz et al., 2025a). *TRecViT* (Pătrăucean et al., 2026) runs RNNs over the embeddings of ViTs for video.

Long-context in 3D reconstruction and robotics: data driven reconstruction, pioneered by *DUSt3R* (Wang et al., 2024) and *MASt3R* (Leroy et al., 2024), first integrated long context with global optimization but then evolved into applying transformers over the full sequence as in *MUSt3R* (Cabon et al., 2025) and *VGGT* (Wang et al., 2025a). For robotics applications this was not considered sufficient as the base models could not deal with longer episodes. SLAM variants shifted to "stitching" based methods, where reconstruction is run on chunks and results are integrated. Examples are *S-MUSt3R* (Antsfeld et al., 2026), *VGGT-long* (Deng et al., 2025) and *VGGT-SLAM 2.0* (Maggio & Carlone, 2026). Chunks have also been integrated end-to-end with TTT layers (Chen et al., 2026; Jin et al., 2026; Xie et al., 2026; Zhang et al., 2026). *CUT3R* (Wang et al., 2025c) is a recurrent model for this task, which maintains a flat vectorial memory integrated into reconstruction with cross-attention. *Kinaema* (Sariyildiz et al., 2025a) is a recurrent model which maintains an implicit memory bank, avoids 3D reconstruction altogether and directly trains for memory-based map-free pose estimation.

Distillation was initially proposed by Hinton et al. (2014) to transfer the power of larger teacher models into smaller students. Distillation can also create universal models by distilling multiple teachers with different strengths or even tasks into a single model, as in *DUNE* (Sariyildiz et al., 2025b) or *AM-RADIO* (Heinrich et al., 2025). While smaller models are often inherently more efficient, distillation can also target students with specific architectures tailored for efficiency. In the context of robotic world models, *DreamDojo* distills a bidirectional transformer into a causal one and with fewer diffusion steps (Gao et al., 2026). Recent work has distilled transformers into state space models (Bick et al., 2025; Wang et al., 2025b). Our formulation differs in that (1) the student is a recurrent transformer which features distributed memory (a memory bank) and, (2) we distill the compression mechanism and therefore directly address the shortcoming of recurrent transformers.

Connections between recurrent and history transformers: recurrence has been added to classical transformers either through recurrent supervision in *NextLat* (Teoh et al., 2025) or through additional latent pathways in *Full-Bandwidth Transformers* (Wang et al., 2026). Both keep context and target improved reasoning, but unlike ours they provide neither constant efficiency nor constant memory. Kumar & Isola (2026) supervise training of recurrent memory in single step predictions from memory generated with an encoder-decoder model which has seen context. In contrast, we distill over sequences and do not rely on passing dynamics information through subsequent gradient optimization steps, and we implement full-fledged systems targeting robotics and streaming computer vision.

3 Distilling Recurrent Transformers

Problem formulation. We consider tasks where a growing history of observations $\{\mathbf{o}_{t'}\}_{t' \in 1:t}$ is taken as input, and a prediction is required at a query time t. We particularly consider tasks where an individual observation $\mathbf{o}_{t'}$, $t' < t$, can have a large impact on the prediction, including observations from far in the past. Predictions are conditioned on a query $\mathbf{q}_t$ that is distinct from the incoming observations. The query describes what information should be retrieved from the history: it is a goal image for Mem-RPE ("*What is the pose of this image w.r.t. your current pose?*") and a textual question for streaming VQA. We keep the method formulation general, then present our main application to Mem-RPE in Section 4 and an additional validation on streaming VQA in Section 5.

For *History Transformers* this prediction is done using the full sequence $\{\mathbf{o}_{t'}\}_{t' \in 1:t}$ plus the query $\mathbf{q}_t$,

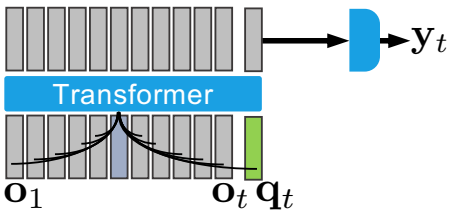
$$\left(\{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_t\}\ ,\ \mathbf{q}_t\right) \xrightarrow{\theta} \mathbf{y}_t. \tag{1}$$

Concretely, as shown in Figure 2(a), a set of multi-head self-attention blocks would process $\{\mathbf{o}_{t'}\}_{t' \in 1:t}$ to obtain a set of token features which then, potentially combined with query features, would be fed to a prediction head. In streaming and robotics settings, as the sequence length grows, the cost of both storing the sequence and attending to it becomes large and at some point prohibitive.

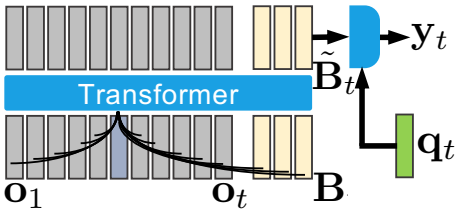
In contrast, *Recurrent Transformers* maintain Memory $\mathbf{M}_t$ and predict from it:

$$(\mathbf{M}_{t-1}, \mathbf{o}_t) \xrightarrow{\theta} \mathbf{M}_t, \quad (\mathbf{M}_t, \mathbf{q}_t) \xrightarrow{\theta} \mathbf{y}_t. \tag{2}$$

They do not require storing a history of observations and only access the current observation $\mathbf{o}_t$ and, more importantly, they do not need to *attend* over an observation history. In what follows, we will describe a method for distilling a History Transformer teacher into a Recurrent Transformer student.



(a) History Transformer



(b) Latent Bottleneck History Transformer

Figure 2: ***Latent Bottleneck History Transformer*** teacher (b). Compared to LLM-style models (a), sequence information is compressed into a fixed-size set of embeddings $\tilde{\mathbf{B}}_t$before going through the prediction head conditioned on the query $\mathbf{q}_t$.

Teacher: Latent Bottleneck History Transformer. We will first motivate a variant of the History Transformer architecture which does not change its computational complexity, but rather introduces an explicit compression mechanism which, once trained, can be distilled into Recurrent Transformers. The "*Latent Bottleneck History Transformer*" (LBHT), illustrated in Figure 2(b) will be used as teacher in the subsequent distillation process.

The LBHT adds a set of B trainable read-out tokens $\mathbf{B} = \{\mathbf{B}^{(1)}, \mathbf{B}^{(2)}, ..., \mathbf{B}^{(B)}\}$ to the inputs and contextualizes them into output tokens $\tilde{\mathbf{B}}_t$. This bears resemblance to the *Perceiver Resampler* (Jaegle et al., 2021) or a similar mechanism in *Flamingo* (Alayrac et al., 2022), and also to action read-out tokens in action experts of VLAs (Black et al., 2025; Shukor et al., 2025). The model learns how to take a sequence of observations and compresses them into a memory bank in the form of the contextualized tokens $\tilde{\mathbf{B}}_t$. Most importantly, this compression does not have access to the query, *i.e.* the goal of the task. The bottleneck $\tilde{\mathbf{B}}_t$ can therefore be interpreted as a form of latent representation, or latent "map", of the observations, which should be potentially useful for any task querying this content.

Teacher: causal vs. non-causal. Transformers come in two variants depending on how attention is computed. Transformers in modern LLMs are causal, *i.e.* past tokens are not re-contextualized with information from newer tokens. Together with KV-caching and teacher forcing during training, this allows to decrease complexity significantly, although the model still needs to attend to all past tokens for each prediction made at each time step t. In this work we opted for a non-causal transformer for three reasons: our experiments show that a non-causal model provides significant gains, which we explain with the advantages of bidirectional attention, making alignment more powerful. Furthermore, we do not perform next token prediction, so teacher forcing during training does not apply. Finally, the teacher is not deployed, its computational complexity is thus of somewhat lesser importance. Note that even non-causal models do not "*cheat*" in streaming applications, as the model does *not* attend to "future" frames. Rather, non-causal models at time t allow tokens from time $t-\tau$ to be re-contextualized by newer tokens from $t-\tau+1$ up to t, but not beyond t.

Student: Recurrent Transformers. In recurrent models, observations are processed sequentially, and at each time step t only the current observation $\mathbf{o}_t$ is available. The model maintains and updates a fixed-size memory $\mathbf{M}_t$, *cf*. Figure 1 (right), integrating new information at each step:

$$\mathbf{M}_t = f_\theta(\mathbf{M}_{t-1}, \mathbf{o}_t), \tag{3}$$

where f_θ is a trainable function like an RNN, LSTM, GRU, or in our case a transformer as in *Kinaema* (Sariyildiz et al., 2025a). The interpretation of this recurrent memory $\mathbf{M}_t$ is similar to the teacher's bottleneck embeddings $\tilde{\mathbf{B}}_t$: both are a latent representation of the observed history which should potentially be useful for any later task querying it. However, its handling and maintenance by the model is fundamentally different and is based on a different compression mechanism. The recurrent model needs to decide which information to retain from the current observation $\mathbf{o}_t$ at each time step, and all information not retained is lost for the future. Blindly storing the full observation is of course not possible, so a smart lossy compression mechanism needs to be learned.[1] We argue, and will show in the experiments, that learning such a mechanism is harder than training a transformer with access to full observation history, but that the latter can act as a teacher for the former.

Distilling into Recurrent Transformers. We propose to distill the memory content of a Latent Bottleneck History Transformer into a Recurrent Transformer (RT) model. As a key design choice, we align the dimensions of the teacher's latent representation $\tilde{\mathbf{B}}_t$ and the student's memory representation

[1]Compression mechanisms in recurrent networks have been analyzed in detail by Dong et al. (2019) in terms of mutual information and the information bottleneck criterion (Bialek et al., 2001).

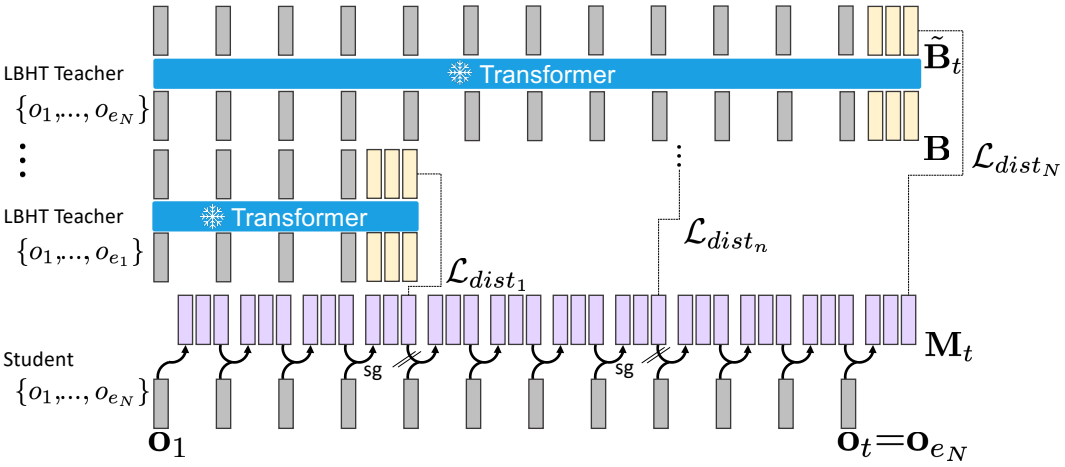


Figure 3: A student implemented as a recurrent transformer with *Kinaema* architecture is trained with distillation from a teacher implemented as *Latent Bottleneck History Transformer (LBHT)*. A single rollout of full length of the student is aligned with multiple rollouts of the teacher over N sub-sequences $\{\mathbf{o}_1,..., \mathbf{o}_{e_n}\}_{n=1..N}$, whose bottleneck embeddings $\tilde{\mathbf{B}}_{e_n}$ are distilled into recurrent memories $\mathbf{M}_{e_n}$ on corresponding time steps. Backpropagation of the student is limited to a fixed number of steps (shown as sg="stop grad") to ensure stability and limit GPU memory usage.

$\mathbf{M}_t$ by using an identical number of embeddings and embedding dimensions. This alignment enables direct distillation via an L_1 loss and in practice, we apply the L_1 loss between the flattened representations $\tilde{\mathbf{B}}_t$ and $\mathbf{M}_t$. Note that in case of dimension mismatch, a linear layer on $\mathbf{M}_t$ could be added. The distillation procedure is based on the following two key hypotheses:

H1: Learnability

It is easier to learn to compress a full sub-sequence, *i.e.* $\{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_t\} \xrightarrow{\theta} \tilde{\mathbf{B}}_t$,
than it is to learn to compress it through recurrence, *i.e.* $(\mathbf{M}_{t-1}, \mathbf{o}_t) \xrightarrow{\theta} \mathbf{M}_t$.

H2: Expressivity of $\tilde{\mathbf{B}}_t$ through recurrence

A sequence of bottleneck representations $\{\tilde{\mathbf{B}}_1, \tilde{\mathbf{B}}_2, ..., \tilde{\mathbf{B}}_t\}$ obtained by applying a trained LBHT teacher to subsequences of increasing lengths can be sufficiently approximated through a learned recurrence function, *i.e.* $(\tilde{\mathbf{B}}_{t-1}, \mathbf{o}_t) \xrightarrow{\theta} \tilde{\mathbf{B}}_t$.

Both hypotheses will be verified experimentally, showing that History Transformers can be trained on substantially longer sequences than Recurrent Transformers, which suffer from stability issues when trained with downstream losses only. *MooG* was trained on sequences of 8 steps, *TRecViT* for 64 steps and *Kinaema* for 100 steps, and we confirm this limitation in our own Mem-RPE experiments. In contrast, we could train HTs over visual sequences of up to 400 steps before running out of GPU memory and without encountering stability issues.

Segment-wise distillation. To best benefit from how the teacher model has learned a strategy to compress sequences of different lengths, we propose segment-wise distillation, illustrated in Figure 3. A rollout of the student over a sequence of length t is decomposed into N equal segments. Segments are indexed by n, their end time step by e_n and the corresponding recurrent memory bank by $\mathbf{M}_{e_n}$. We distill each of the final end memory bank with the following L_1 loss:

$$\mathcal{L}_{dist} = \sum_{n=1}^{N} \mathcal{L}_{dist_n} = \sum_{n=1}^{N} \left|\left| \mathbf{M}_{e_n} - \tilde{\mathbf{B}}_{e_n} \right|\right|_1 . \tag{4}$$

As shown in Figure 3, a single forward pass over the student produces all the memory representations $\mathbf{M}_{e_n}$, whereas each individual bottleneck representation $\tilde{\mathbf{B}}_{e_n}$ is obtained with its own separate forward pass over the teacher, as we distill from a non-causal teacher model. Let us note that different loss terms $\mathcal{L}_{dist_n}$ are attached to different time steps of the full sequence. To limit GPU memory usage, we limit backpropagation of each loss term to a maximum allowable length. In the experimental section we experiment with different segment lengths and backpropagation horizons. When backpropagation horizon is larger than the segment length, gradient calculations "overlap", *i.e.* for a given time step e_n, gradients from multiple loss terms are accumulated.

The teacher is first trained with the downstream task loss and then frozen for distillation. Student training combines downstream supervision with memory distillation: $\mathcal{L}_{student} = \mathcal{L}_{task} + \lambda_{dist}\mathcal{L}_{dist}$. The task loss and its supervision points are defined by the application.

4 Application to Memory-Based Relative Pose Estimation

Task. We evaluate our method on a challenging task in Embodied AI, which requires building a representation from visual observations collected along a trajectory and answering visual queries from it. Given a query image, the task is to predict the pose of the shown scene with respect to the robot's current pose. This task is related to map-free localization (Barroso-Laguna et al., 2026) but, as introduced in (Sariyildiz et al., 2025a) as "*Mem-RPE*" (memory-based relative pose estimation), features additional challenges specific to Embodied AI: (1) observations are processed sequentially as sensed by an agent, and (2) the coordinate frame for pose estimation is centered on the agent, and is thus a *moving coordinate frame*. This makes the task realistic and goal oriented, as an agent can take decisions based on its state. Agents move in 3D environments and at each time step t receive a pair of sensor readings $\mathbf{o}_t = (\mathbf{x}_t, \mathbf{u}_t)$, where $\mathbf{x}_t \in \mathbb{R}^{3\times H\times W}$ is an RGB image of size 112×112, and $\mathbf{u}_t \in \mathbb{R}^7$ is an odometry estimate in the form of a difference of agent poses between t and $t-1$. The task is conditioned on a query image $\mathbf{q}_t \in \mathbb{R}^{3\times H\times W}$ and consists in predicting pose $\mathbf{y}_t = (\mathbf{t}, \mathbf{R})$, where $\mathbf{t} \in \mathbb{R}^3$ is the translation and $\mathbf{R} \in SO(3)$ is the rotation matrix.

Observation encoding. Both student and teacher encode observations $\mathbf{o}_t = (\mathbf{x}_t, \mathbf{u}_t)$ in the same way: visual inputs are encoded via a fine-tuned DINO-v2 ViT-s (Oquab et al., 2024), where each of the 64 patch embeddings is down-projected to 64 values. Odometry $\mathbf{u}_t$ is embedded via an MLP into 64 values. The two representations are concatenated channel-wise into $\tilde{\mathbf{o}}_t = (\tilde{\mathbf{x}}_t, \tilde{\mathbf{u}}_t)$ of dimension 4160.

Student architecture. Our student is based on the *Kinaema* model (Sariyildiz et al., 2025a) , a high-capacity recurrent architecture maintaining and updating a distributed memory state $\mathbf{M}_t$ consisting of B=20 embeddings of dimension E=3072. More details are given in appendix B.2.

Teacher architecture. We design a teacher tailored for distilling the *Kinaema* student given above. The teacher is an LBHT (see Section 3 and Figure 2) with the same embedding size E=3072 and with B=20 bottleneck embeddings. By design, the bottleneck tensor of the teacher, $\tilde{\mathbf{B}}_t$, and the memory bank of the student, $\mathbf{M}_t$, have thus the same tensor shape. The teacher receives the same type of encoded observations $\tilde{\mathbf{o}}_t$, but observation encoders are not shared between teacher and student. The teacher has 7 transformer layers compared to the 3 layers of the *Kinaema* student, while both architectures were optimized. The fact that the optimal number of layers is higher for the teacher than for the student is not necessarily surprising. It is well known that recurrent models tend to unroll some of their representation computations over time, additionally to performing them over layers[2].

Teacher training. the teacher is trained by directly supervising pose estimation as in (Sariyildiz et al., 2025a), $\mathcal{L}_{RPE} = \sum_i \big(|\mathbf{t}_i - \mathbf{t}_i^*| + |\mathbf{R}_i - \mathbf{R}_i^*| \big)$, where $(\mathbf{t}_i, \mathbf{R}_i)$ and $(\mathbf{t}_i^*, \mathbf{R}_i^*)$ are predicted and GT pose for training image i, respectively. The teacher also uses an auxiliary masked image modeling loss, which reconstructs the same query images after they have been masked through a second decoder head. Training the student combines the distillation loss $\mathcal{L}_{dist}$ (Equation 4), with the downstream loss $\mathcal{L}_{RPE}$. The teacher is trained on sequences of randomized length of $50\leq T\leq400$, far beyond the limit of the original *Kinaema* model ($T\leq100$).

Student distillation. Masked image modeling is not used for the student. Distillation is done on sequences of randomized length $50\leq T\leq200$ with multiple segments (see Section 3), their numbers are ablated in the experimental section. The downstream loss $\mathcal{L}_{RPE}$ is additionally applied on each segment n by taking the memory $\mathbf{M}_{e_n}$, predicting relative pose for $2e_n$ query images of two different types: (i) the e_n observed images $\{\mathbf{x}_t\}_{t\in1:e_n}$, and (ii) e_n *alternative* images $\{\mathbf{x}_t^{alt}\}_{t\in1:e_n}$, which have *not* been observed but lie in the previously observed region of the scene, generated in the simulator.

Experimental setup. We use the data and protocol from Sariyildiz et al. (2025a)[3]: data was generated in the Habitat simulator (Szot et al., 2021) from the HM3D (Ramakrishnan et al., 2021) and Gibson

[2] As an extreme case one can consider a single-layer RNN, which has been shown to be Turing complete in spite of its simplicity (Siegelmann & Sontag, 1995). However, this might induce delays in outputs, and it has been shown that 1-layer RNNs are fundamentally equivalent to multi-layer RNNs with delays (Turek et al., 2020).

[3] https://github.com/naver/kinaema

Table 1: **Comparisons with the state of the art** on **RPE-test**.

Model	Obs hist	Mem size	Seq len 200 $1m$ 10°	Seq len 200 $1m$ 90°	Seq len 200 $2m$ 90°	Seq len 800 $1m$ 10°	Seq len 800 $1m$ 90°	Seq len 800 $2m$ 90°
LBHT (Teacher) *(not comparable)*	✓	61.4k	**44**	**56**	**72**	**16**	**22**	**44**
MooG (van Steenkiste et al., 2024)	✗	524.3k	0	5	14	0	3	9
LRU (Orvieto et al., 2023)	✗	3.1k	4	18	34	2	9	20
EMA (Eberhard et al., 2025)	✗	153.6k	6	18	34	3	11	24
xLSTM (Beck et al., 2024)	✗	2,359.3k	8	23	47	5	13	29
GRU (Cho et al., 2014)	✗	3.1k	12	32	56	4	14	31
Kinaema (Sariyildiz et al., 2025a)	✗	61.4k	21	41	63	10	21	37
Chimera (ours)	✗	61.4k	**38**	**45**	**65**	**22**	**26**	**48**

(Xia et al., 2018) datasets with independent scene splits related to Mem-RPE, namely **RPE-train**: 800 scenes from the HM3D/train split; **RPE-val**: 72 scenes from the Gibson/train split; **RPE-test**: 100 scenes from the HM3D/val split. All result tables have color-coded backgrounds indicating the splits. Actions are chosen from a discrete action space, *e.g.* $\mathcal{A}$ ={`move forward, turn left, turn right`}. The test set shows some out-of-distribution settings as (1) test sequences are significantly longer than the training sequence length (up to T=800); (2) Motion between subsequent poses is different, as forward is 25cm instead of 10cm, and turning angles are 10° instead of 5°.

Metrics. We report accuracy of correctly estimated poses with three thresholds as in (Sariyildiz et al., 2025a): $< 1m$ of translation and $< 10°$ of rotation errors, $<$ 1m and 90°, and $<$ 2m and 90°.

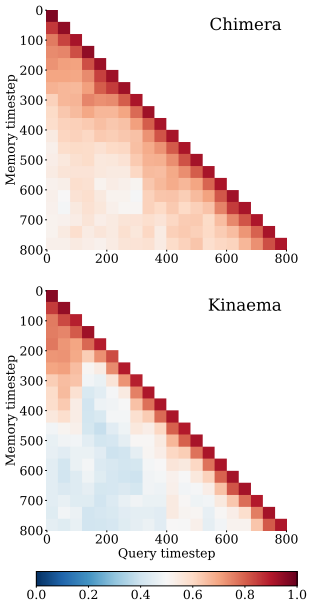


Figure 4: **Comparison between *Chimera* and *Kinaema*** in terms of Mem-RPE performance (2m 90°) for varying query and memory timesteps. Each cell (t, t') corresponds to accuracy for queries $[q_{t'-40}, q_{t'}]$ from memory $\mathbf{M}_t$.

Comparison to the state of the art. Table 1 provides a comparison with the state of the art in terms of Mem-RPE accuracies on **RPE-test**. The LBHT teacher is the only model with access to the observation history and therefore is not comparable to the other models. All other models are recurrent and we report them with their corresponding memory size. *Chimera* significantly outperforms the state of the art, including *Kinaema* that shares the exact same architecture, including all hyper-parameters (numbers of layers, heads, embedding size etc.). This shows the effectiveness of our proposed compression strategy to distill from a LBHT teacher. On sequence length 200, our method performs close to the teacher, with *e.g.* 65% accuracy at 2m90°*vs.* 72% for LBHT and 63% for *Kinaema*. Interestingly, *Chimera* even performs slightly better than LBHT on long sequences and significantly better than Kinaema. All remaining ablations and analysis below have been performed on **RPE-val**.

Mem-RPE performance with varying query age. Figure 5 (left) shows how performance evolves when memory $\mathbf{M_{800}}$ is fixed after 800 observed frames, and considering queries further in the past, *i.e.* earlier in the sequence, or close to t=800 and thus to the most recent observation $\mathbf{o}_t$ for the memory. Note that we do not query observed frames but *alternative* images, *i.e.* query images close to actual observations. Performance increases on the right part of the plots for all methods as the queries become closer to the last observations encoded within the memory. We observe that *Chimera* performs better than *Kinaema* and close to the LBHT teacher at all timesteps.

Mem-RPE performance with varying eval sequence length T. In Figure 5 (right), we show how Mem-RPE accuracies evolve when considering subsequences of various lengths, up to 800. *Chimera* outperforms *Kinaema* for all sequence lengths. Interestingly, the LBHT teacher performance drops steadily for sequences longer than 400 frames, used for its training, while our recurrent model is more stable for longer sequences despite being trained with sequences of at most 200 frames. Figure 4 further compares our approach to *Kinaema* by considering all ranges of queries and memories timesteps. The diagonal corresponds to queries that are close to the current position, explaining its high performance. We observe overall that Chimera significantly outperforms Kinaema, in particular for large values of memory timesteps.

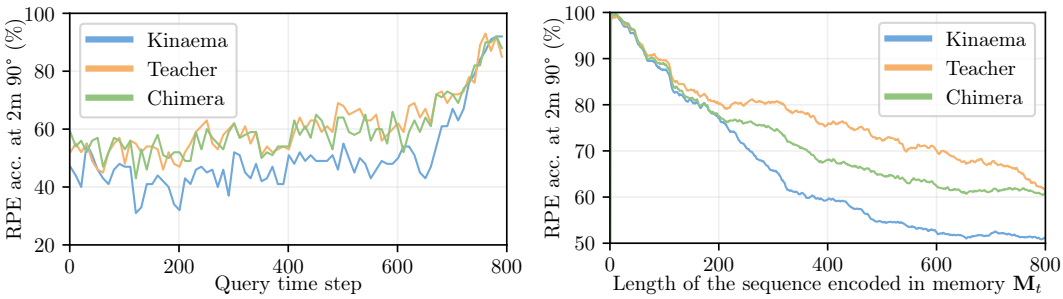


Figure 5: **Mem-RPE performance as function of encoded sequence length and query age** for the teacher, our model, and *Kinaema* on **RPE-val** (accuracy at 2m 90°). *Left:* We vary t on the x-axis and consider memory $\mathbf{M}_{800}$ having encoded a full sequence of 800 frames. *Right:* we vary the length t of the sequence encoded into memory $\mathbf{M}_t$ and query it with frames $\leq t$. All queries are "alternative" frames, *i.e.* close to observations in the scene, but *not* actually observed.

Table 2: **Impact of model choice on full-scale training** (seq. length T=400 and full 600 epochs). *Kinaema* diverges when trained on these scales (**RPE-val**).

Model	Seq len 200			Seq len 800		
	$1m$ 10°	$1m$ 90°	$2m$ 90°	$1m$ 10°	$1m$ 90°	$2m$ 90°
History Transformer	**59**	**74**	**85**	6	19	35
Latent Bottleneck History Transformer	28	51	74	**16**	**31**	**55**
Kinaema (Sariyildiz et al., 2025a)	diverges			diverges		

Impact of model choice when full-scale trained. Table 2 compares several base models in terms of Mem-RPE accuracies on **RPE-val** when training on long sequences of up to T=400 and for 600 epochs. We compare HT without bottleneck, *cf*. Figure 2(a), with the LBHT teacher, *cf*. Figure 2(b). In this large-scale training setting, *Kinaema* diverges; the baseline transformer (a) obtains excellent accuracy for sequence length 200, that is among the sequence lengths seen during training. However, it performs poorly on longer sequences, which might come from the difficulty of applying transformers on sequences significantly longer than the ones seen during training (Zhao et al., 2024; Huang et al., 2025). In contrast, LBHT is significantly less affected by this phenomenon.

Design choices for LBHT teacher training. Table 3 reports a sensitivity analysis of varying the maximum length T of training sequences and the number of epochs when training the LBHT teacher. Large maximum sequence lengths (*e.g.* 400) are necessary, and longer sequences require longer training too. Causal transformers, where past tokens are not re-contextualized by newer observations within the available prefix, perform substantially worse on this task.

Memory stability. Figure 6 shows the stability of the memory as the *normalized memory update norm* $\mathcal{N}$, defined as the L1 norm of the memory update between two consecutive timesteps, normalized by the average norm of the memory over the sequences: $\mathcal{N}_t = \frac{1}{\eta}||\mathbf{M}_t - \mathbf{M}_{t-1}||_1$, where $\eta = \frac{1}{T}\sum_{t=1}^{T}||\mathbf{M}_t||_1$. *Kinaema* shows a higher rate of change up to T=100, for which it was trained, followed by a drop, which we conjecture to be linked to a lack of generalization of its memory management system: when new content is added to memory filled beyond amounts seen in training, it seems to change its update behavior. In contrast, the LBHT teacher exhibits a flat change rate over time and also beyond the length of T=400 on which it was trained, with an overall lower norm. The distillation process makes *Chimera*'s behavior similar to the teacher.

Specialization of memory tokens. We investigate whether individual memory tokens develop specialized roles. We compute the absolute change of each memory token at each time step, sum it across the trajectory, and then normalize by the total change across all tokens. A uniform distribution would indicate that all tokens are updated equally, while a skewed distribution would suggest that certain tokens specialize for storing persistent information while others are more dynamic and frequently rewritten. Figure 7 shows the results obtained on **RPE-val** with sequences of length $T = 200$. Both models exhibit a non-uniform distribution, confirming that memory tokens do specialize. Interestingly, the distilled model displays a more skewed distribution, with the majority of the tokens changing only a little while a few tokens account for a large portion of the change. This suggests that *Chimera* learns to maintain a stable core of memory tokens that are updated less frequently, while capturing discriminative information in a few highly dynamic tokens.

Table 3: **Design choices for LBHT teachers.** Left: different training sequence lengths T and different training lengths for a non-causal model. Right: Impact of transformer causality $\mathcal{C}$ (**RPE-val**).

Train T	Num epochs	Seq len 200 $1m$ 10°	Seq len 200 $1m$ 90°	Seq len 200 $2m$ 90°	Seq len 800 $1m$ 10°	Seq len 800 $1m$ 90°	Seq len 800 $2m$ 90°
400	**600**	28	51	74	16	31	55
400	**300**	21	49	73	10	26	50
400	**200**	9	28	51	6	19	40
300	**200**	12	34	59	6	19	42
200	**200**	16	40	65	6	18	40

Train T	Num epochs	$\mathcal{C}$	Seq len 200 $1m$ 10°	Seq len 200 $1m$ 90°	Seq len 200 $2m$ 90°	Seq len 800 $1m$ 10°	Seq len 800 $1m$ 90°	Seq len 800 $2m$ 90°
100	**200**	✗	17	35	55	4	10	30
100	**200**	✓	8	18	35	2	6	16
200	**300**	✗	28	54	76	6	14	38
200	**300**	✓	10	22	40	4	10	27

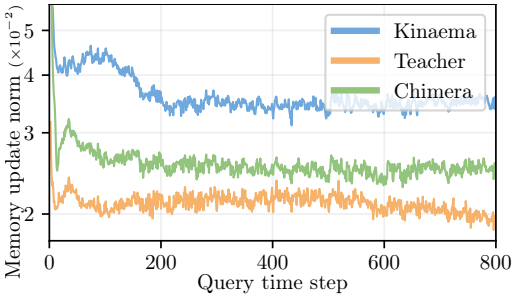


Figure 6: **Stability of memory** for the teacher, our model, and *Kinaema* on **RPE-val** . Memory stability is measured by the update norm $\mathcal{N}$.

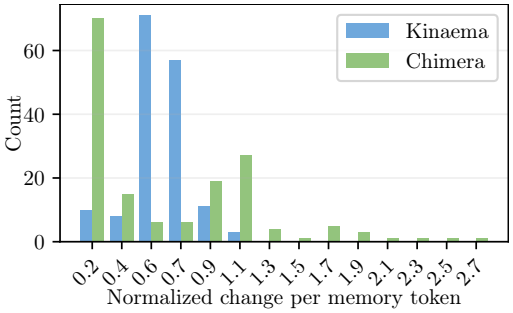


Figure 7: **Histogram memory tokens changes** accumulated over a trajectory. Aggregated results for **RPE-val** sequences of length $T = 200$.

5 Additional Application: Streaming Visual Question Answering

To complement our main evaluation on MemRPE, we apply bottleneck distillation to streaming visual question answering, where a model answers textual questions using the video observations available at query time. This experiment tests whether supervising the compressed history also benefits recurrent predictions in a video-language setting.

Table 4: **Streaming VQA results** (accuracies in %).

Model	OVO-Bench	StreamingBench (Real)
No vision	33.5	39.3
Bottleneck teacher	**37.0**	49.4
Student w/o distillation	34.3	48.1
Student w/ distillation	36.8	**49.8**

We use LongVA-7B (Zhang et al., 2025a) with a frozen visual encoder and language model, and train temporal compression modules on LLaVA-Video (Zhang et al., 2025b). The LBHT teacher jointly compresses the observed frame representations into a bottleneck, while the recurrent student updates a fixed-size memory sequentially. Both provide visual context to the same language model. We compare students trained with the answer-prediction loss alone and with the additional segment-wise bottleneck supervision of Section 3. Appendix G provides more details about the architecture, datasets, training setup, and teacher/student ablations.

Table 4 reports accuracy on OVO-Bench (Niu et al., 2025) and the 'Real' task of StreamingBench (Wu et al., 2024). Distillation improves the recurrent student from 34.3% to 36.8% on OVO-Bench and from 48.1% to 49.8% on StreamingBench Real, gains of 2.5 and 1.7 percentage points, respectively. The distilled student approaches the teacher on OVO-Bench and slightly exceeds its reported Real score. These gains provide additional evidence that our proposed memory distillation can also improve recurrent memory learning for video-language prediction.

6 Conclusion

We addressed the fundamental performance gap between History Transformers and Recurrent Transformers in long-horizon visual tasks and argued that it is linked to the optimization difficulty associated with learning "on-the-fly" compression. We demonstrated that a full-history teacher's compression strategy can be distilled into the fixed-size memory of a recurrent student. Our experiments on sequential map-free pose estimation and streaming visual question answering confirm our two primary hypotheses: first, that supervising a student's memory with a teacher's bottleneck representation significantly eases the learning of complex temporal dependencies; and second, that recurrent architectures are sufficiently expressive to approximate these non-causal compression strategies.

AI USE STATEMENT

In this work, we used generative AI tools for "research ideation or execution": while we have initially fully developed the code for Mem-RPE, we provided an AI tool with the Mem-RPE codebase and prompted it to adapt it for the Streaming-VQA task. The codebase has been then fully reviewed before experiments were launched. AI tools were also used for "aid or polish writing": just to find typos and mathematical formulas errors once the paper were ready to be submitted, and to beautify a draft of Figure 13 from the appendix. We did not use AI tools to, for "retrieval and discovery", "draft sections of the paper", "generating synthetic datasets". "Proving mathematical claims" is not applicable to our work. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide all architectural details and training hyperparameters through the main paper and appendix.

REFERENCES

Milad Aghajohari, Kamran Chitsaz, Amirhossein Kazemnejad, Sarath Chandar, Alessandro Sordoni, Aaron Courville, and Siva Reddy. The markovian thinker: Architecture-agnostic linear scaling of reasoning. In *International Conference on Learning Representations (ICLR)*, 2026.

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikołaj Bińkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karén Simonyan. Flamingo: a visual language model for few-shot learning. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Leonid Antsfeld, Boris Chidlovskii, Yohann Cabon, Vincent Leroy, and Jerome Revaud. S-MUSt3R: Sliding multi-view 3d reconstruction. *arXiv preprint arXiv:2602.04517*, 2026.

Axel Barroso-Laguna, Tommaso Cavallari, Victor Adrian Prisacariu, and Eric Brachmann. A scene is worth a thousand features: Feed-forward camera localization from a collection of image features. In *International Conference on Learning Representations (ICLR)*, 2026.

Maximilian Beck, Korbinian Pöppel, Markus Spanring, Andreas Auer, Oleksandra Prudnikova, Michael Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. xLSTM: Extended long short-term memory. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

William Bialek, Ilya Nemenman, and Naftali Tishby. Predictability, Complexity, and Learning. *Neural Computation*, 2001.

Aviv Bick, Kevin Y. Li, Eric P. Xing, J. Zico Kolter, and Albert Gu. Transformers to ssms: Distilling quadratic knowledge to subquadratic models. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott

Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2020.

Yohann Cabon, Lucas Stoffl, Leonid Antsfeld, Gabriela Csurka, Boris Chidlovskii, Jerome Revaud, and Vincent Leroy. MUSt3R: Multi-view network for stereo 3d reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. TTT3R: 3d reconstruction as test-time training. In *International Conference on Learning Representations (ICLR)*, 2026.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. FlashAttention: Fast and memory-efficient exact attention with io-awareness. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2022.

Kai Deng, Zexin Ti, Jiawei Xu, Jian Yang, and Jin Xie. VGGT-Long: Chunk it, loop it, align it – pushing vggt's limits on kilometer-scale long RGB sequences. *arXiv preprint arXiv:2507.16443*, 2025.

Zhe Dong, Deniz Oktay, Ben Poole, and Alexander A Alemi. On predictive information in rnns. *arXiv preprint arXiv:1910.09578*, 2019.

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 (ICLR)*, 2021.

Onno Eberhard, Michael Muehlebach, and Claire Vernade. Partially observable reinforcement learning with memory traces. *arXiv preprint arXiv:2503.15200*, 2025.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, Qianli Ma, Seungjun Nah, Loic Magne, Jiannan Xiang, Yuqi Xie, Ruijie Zheng, Dantong Niu, You Liang Tan, K. R. Zentner, George Kurian, Suneel Indupuru, Pooya Jannaty, Jinwei Gu, Jun Zhang, Jitendra Malik, Pieter Abbeel, Ming-Yu Liu, Yuke Zhu, Joel Jang, and Linxi "Jim" Fan. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *First Conference on Language Modeling*, 2024.

Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

Greg Heinrich, Mike Ranzinger, Hongxu, Yin, Yao Lu, Jan Kautz, Andrew Tao, Bryan Catanzaro, and Pavlo Molchanov. RADIOv2.5: Improved baselines for agglomerative vision foundation models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *NeurIPS Deep Learning Workshop*, 2014.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computing*, 1997.

Xinting Huang, Andy Yang, Satwik Bhattamishra, Yash Sarrof, Andreas Krebs, Hattie Zhou, Preetum Nakkiran, and Michael Hahn. A formal framework for understanding length generalization in transformers. In *International Conference on Learning Representations (ICLR)*, 2025.

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

Steeven Janny, Aurélien Beneteau, Nicolas Thome, Madiha Nadri, Julie Digne, and Christian Wolf. Eagle: Large-scale learning of turbulent fluid dynamics with mesh transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Haian Jin, Rundi Wu, Tianyuan Zhang, Ruiqi Gao, Jonathan T. Barron, Noah Snavely, and Aleksander Holynski. Zipmap: Linear-time stateful 3d reconstruction via test-time training. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Łukasz Kaiser and Ilya Sutskever. Neural gpus learn algorithms. In *International Conference on Learning Representations (ICLR)*, 2016.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning (ICML)*, 2020.

Akarsh Kumar and Phillip Isola. Pretraining recurrent networks without recurrence. *arXiv preprint arXiv:2606.06479*, 2026.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with MASt3R. In *European Conference on Computer Vision (ECCV)*, 2024.

Dominic Maggio and Luca Carlone. VGGT-SLAM 2.0: Real-time dense feed-forward scene reconstruction. *arXiv preprint arXiv:2601.19887*, 2026.

Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. OVO-Bench: How far is your video-llms from real-world online video understanding? In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research (TMLR)*, 2024.

Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting Recurrent Neural Networks for Long Sequences. In *International Conference on Machine Learning (ICML)*, 2023.

Viorica Pătrăucean, Xu Owen He, Joseph Heyward, Chuhan Zhang, Mehdi S. M. Sajjadi, George-Cristian Muraru, Artem Zholus, Mahdi Karami, Ross Goroshin, Yutian Chen, Simon Osindero, João Carreira, and Razvan Pascanu. TRecViT: A recurrent video transformer. *Transactions on Machine Learning Research (TMLR)*, 2026.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine Mcleavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning (ICML)*, 2023.

Santhosh Kumar Ramakrishnan, Aaron Gokaslan, Erik Wijmans, Oleksandr Maksymets, Alexander Clegg, John M Turner, Eric Undersander, Wojciech Galuba, Andrew Westbury, Angel X Chang, Manolis Savva, Yili Zhao, and Dhruv Batra. Habitat-Matterport 3D dataset (HM3D): 1000 large-scale 3D environments for embodied AI. In *NeurIPS Datasets and Benchmarks Track*, 2021.

Michael S. Ryoo, Keerthana Gopalakrishnan, Kumara Kahatapitiya, Ted Xiao, Kanishka Rao, Austin Stone, Yao Lu, Julian Ibarz, and Anurag Arnab. Token turing machines. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Mert Bulent Sariyildiz, Philippe Weinzaepfel, Guillaume Bono, Gianluca Monaci, and Christian Wolf. Kinaema: a recurrent sequence model for memory and pose in motion. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2025a.

Mert Bulent Sariyildiz, Philippe Weinzaepfel, Thomas Lucas, Pau de Jorge, Diane Larlus, and Yannis Kalantidis. DUNE: Distilling a universal encoder from heterogeneous 2D and 3D teachers. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025b.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, Simon Alibert, Matthieu Cord, Thomas Wolf, and Remi Cadene. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Hava T. Siegelmann and Eduardo D. Sontag. On the Computational Power of Neural Nets. *Journal of Computer and System Sciences*, 1995.

Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2015.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Jayden Teoh, Manan Tomar, Kwangjun Ahn, Edward S Hu, Tim Pearce, Pratyusha Sharma, Akshay Krishnamurthy, Riashat Islam, Alex Lamb, and John Langford. Next-latent prediction transformers learn compact world models. *arXiv preprint arXiv:2511.05963*, 2025.

Javier S. Turek, Shailee Jain, Vy Vo, Mihai Capota, Alexander G. Huth, and Theodore L. Willke. Approximating stacked and bidirectional recurrent architectures with the delayed recurrent neural network. In *International Conference on Machine Learning (ICML)*, 2020.

Sjoerd van Steenkiste, Daniel Zoran, Yi Yang, Yulia Rubanova, Rishabh Kabra, Carl Doersch, Dilara Gokay, Joseph Heyward, Etienne Pot, Klaus Greff, Drew A. Hudson, Thomas Albert Keck, Joao Carreira, Alexey Dosovitskiy, Mehdi S. M. Sajjadi, and Thomas Kipf. Moving off-the-grid: Scene-grounded video representations. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

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

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. VGGT: Visual geometry grounded transformer. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025a.

Penghao Wang, Yuhao Zhou, Mengxuan Wu, Panpan Zhang, Zhangyang Wang, and Kai Wang. Data efficient any transformer-to-mamba distillation via attention bridge. *arXiv preprint arXiv:2510.19266*, 2025b.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025c.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

Xi Wang, Ziyang Cai, Zheng Zhan, Harry Dong, Ying Fan, Gustavo de Rosa, Tim Pearce, and John Langford. Full-bandwidth transformer. *arXiv preprint arXiv:2608.08888*, 2026.

Cheng-Kuang Wu, Zhi R Tam, Chieh-Yen Lin, Yun-Nung Chen, and Hung-yi Lee. Streambench: Towards benchmarking continuous improvement of language agents. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024.

Fei Xia, Amir R Zamir, Zhiyang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Tao Xie, Peishan Yang, Yudong Jin, Yingfeng Cai, Wei Yin, Weiqiang Ren, Qian Zhang, Wei Hua, Sida Peng, Xiaoyang Guo, and Xiaowei Zhou. Scal3R: Scalable test-time training for large-scale 3d reconstruction. *arXiv preprint arXiv:2604.08542*, 2026.

Zeyi Xu, Jinfan Liu, Kuangxu Chen, Ye Chen, Zhangli Hu, and Bingbing Ni. AMR-Transformer: Enabling efficient long-range interaction for complex neural fluid simulation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Junyi Zhang, Charles Herrmann, Junhwa Hur, Chen Sun, Ming-Hsuan Yang, Forrester Cole, Trevor Darrell, and Deqing Sun. LoGeR: Long-context geometric reconstruction with hybrid memory. *arXiv preprint arXiv:603.03269*, 2026.

Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. Long context transfer from language to vision. *Transactions on Machine Learning Research (TMLR)*, 2025a.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. LLaVA-Video: Video instruction tuning with synthetic data. *Transactions on Machine Learning Research (TMLR)*, 2025b.

Liang Zhao, Xiachong Feng, Xiaocheng Feng, Weihong Zhong, Dongliang Xu, Qing Yang, Hongtao Liu, Bing Qin, and Ting Liu. Length extrapolation of transformers: A survey from the perspective of positional encoding. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2024.

Appendix

A Limitations

The Mem-RPE model has currently been trained on fixed camera intrinsics and image sizes. Training on multiple intrinsics is ongoing work. It is also trained in simulation only, which allows to provide ground-truth poses for observed frames and, most importantly, to generate "alternative frames", something which is hard to do with video input.

The streaming VQA experiments use real videos but are limited to a single vision-language backbone which is fully frozen.

B Mem-RPE architecture details

B.1 Input encoders

Input images are of resolution 112×112 and are encoded with a DINO-v2 (Oquab et al., 2024) ViT-s backbone that is initialized with DINO-v2 and finetuned. The encoder has a patch size 14×14, leading to 8×8 patch tensors per image. Patch embeddings are linearly down-projected to 64 values, resulting in visual embeddings of dimension 4096, which are combined with linear delta pose embeddings of size 64.

B.2 Student

The student has the same architecture as *Kinaema* (Sariyildiz et al., 2025a). The model performs two operations, a recurrent memory update step, and a decoding step which queries memory given a query $\mathbf{q}_t$,

$$\mathbf{M}_t = \mathit{Update}\left(\mathbf{M}_{t-1}, \tilde{\mathbf{o}}_t\right), \quad \mathbf{y}_t = \text{Decoder}\big(\mathbf{M}_t, \mathit{Enc}_{\mathit{goal}}\,\mathbf{q}_t\big). \tag{5}$$

The decoder is implemented as a transformer using cross-attention between the goal query and the memory embeddings $\mathbf{M}_t$.

The core innovation of *Kinaema* lies in its scalable and gated update mechanism. To handle high-capacity memory without parameter growth, Update is decomposed into three stages:

1. **Correction:** each of the B memory embeddings $\{\mathbf{M}_{t-1,b}\}_{b\in1:B}$ is channel-wise concatenated with the current encoded observations $\tilde{\mathbf{o}}_t$ and summed with a positional embedding $\mathbf{e}_b$, then projected: $\mathbf{M}^{corr}_{t,b} = \mathit{Linear}([\mathbf{M}_{t-1,b} + \mathbf{e}_b, \tilde{\mathbf{x}}_t, \tilde{\mathbf{u}}_t])$.
2. **Contextualization:** a transformer with self-attention $\tilde{\mathbf{M}}_t = \mathit{SelfAttn}(\mathbf{M}^{corr}_t)$ allows for interaction between the B distributed memory slots.
3. **Shared Gating:** to ensure stability and model varying temporal dynamics, a GRU-based (Cho et al., 2014) gating block is applied: $\mathbf{M}_{t,b} = \mathit{GRU}(\mathbf{M}_{t-1,b}, \tilde{\mathbf{M}}_{t,b})$. Crucially, the gating weights are shared across all B memory embeddings, decoupling the memory capacity from the gating complexity.

Our student is identical, with the exception of *Kinaema*'s masked image modeling head, not used here. The architecture details are:

- The transformer block has 3 layers, 24 heads, and an MLP factor of 4. The gating block is a GRU with 3 layers.
- **Memory reshape**. As in *Kinaema* (Sariyildiz et al., 2025a), the 20 tokens of dim 3072 are reshaped into 160 tokens of dim 384 before passing them to the decoder. This allows the memory model to handle more values (higher dim) in the feed-forward layer of each memory token while at the same time allowing the decoder to compute fine-grained attention between query image patches and a larger number of memory tokens.
- The decoder (which queries into the updated memory to provide pose predictions) employs a cross-attention (CA) layer with no residual connection. Its keys and values are provided by the (reshaped) memory tokens and the queries are the patch tokens from the goal image encoder $\mathit{Enc}_{\mathit{goal}}$. The motivation behind this design with missing skip connections is to

make the decoder rely solely on the memory outputs and prevent information leakage from Enc_{goal} into the decoder. A learnable token (*i.e.* a `CLS` token) is attached to the output of the CA layer and given as input to a sequence of 4 standard self-attention (SA) blocks (Vaswani et al., 2017). Finally, the `CLS` token is detached and relative pose of the goal image is estimated by a MLP with 1 hidden layer.

B.3 Latent Bottleneck Transformer Teacher

The transformer block of the Latent Bottleneck History Transformer (LBHT) teacher has 7 layers, 24 heads, and an MLP ratio of 4.0. The input embeddings of size 4160 are linearly down-projected to size E=3072, and then token-wise concatenated with the B=20 bottleneck embeddings of the same embedding dim. The contextualized bottleneck embeddings $\tilde{\mathbf{B}}_t$ are used for distillation. For the RPE loss, the teacher uses the same decoder architecture as the student.

C Mem-RPE training details

C.1 Training the LBHT teacher

Batch size	32
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 1e{-}15$)
Base Learning rate	1.5e-4
Minimum LR	1e-8
Scheduler	Cosine annealing
Warm-up epochs	40
Gradient clipping	1.0
Weight decay	0.05
Epochs	600
Weight for RPE loss	1.0
Weight for MIM loss	1.0

C.2 Distilling the student

To compute the distillation loss (Equation 4), we first flatten the memory representation, resulting in vectors of dimension 61,440. We then compute the sum of absolute differences. The hyperparameter λ_{dist} that combines the loss $\mathcal{L}_{dist}$ with $\mathcal{L}_{RPE}$ is set to 3×10^{-5} in all experiments of the main paper, *i.e.* the training loss for a given segment n is $\mathcal{L}_n = \lambda_{dist}\mathcal{L}_{dist_n} + \mathcal{L}_{RPE_n}$.

Table 5 provides an ablation on this hyperparameter. First, we observe that removing distillation leads to a huge drop of performance. This highlights the importance of our approach that consists in adding this memory distillation loss from a LBHT teacher model. Then the performance increases until a plateau centered around 3×10^{-5} while the performance starts decreasing again for too high values.

Batch size	32
Optimizer	AdamW ($\beta_1 = 0.9, \beta_2 = 0.99, \epsilon = 1e{-}15$)
Base Learning rate	1.5e-4
Minimum LR	1e-8
Scheduler	Cosine annealing
Warm-up epochs	40
Gradient clipping	1.0
Weight decay	0.05
Epochs	400
Weight for RPE loss	1.0
Weight for Distillation loss	3×10^{-5}

D Mem-RPE distillation design choices

In Table 6 we report a sensitivity analysis of the distillation procedure into the student. We find that performing multi-step distillation improves performance significantly (*cf*. Equation 4), with

Table 5: **Impact of the weight λ_{dist} of the distillation loss** on **RPE-val**. This ablation is done with training for 200 epochs.

λ_{dist}	Seq len 200			Seq len 800		
	$1m$ 10°	$1m$ 90°	$2m$ 90°	$1m$ 10°	$1m$ 90°	$2m$ 90°
0.0	5	23	47	3	15	37
1×10^{-5}	32	45	68	20	28	50
$\mathbf{3 \times 10^{-5}}$	**38**	**57**	76	**25**	**40**	**63**
1×10^{-4}	34	**57**	**77**	21	37	60
3×10^{-4}	36	56	**77**	22	35	59
1×10^{-3}	27	49	74	16	30	54

Table 6: **Design choices for distilling**: we explore different numbers N of distillation segments (left) and limits for backpropagation (right, top rows, N=4), as well as adding a Masked Image Modeling loss (right, bottom row, N=4). This ablation is performed under 200-epoch training schemes.

Num. segments	Seq len 200			Seq len 800		
	$1m$ 10°	$1m$ 90°	$2m$ 90°	$1m$ 10°	$1m$ 90°	$2m$ 90°
1	14	33	61	8	19	45
2	16	41	66	10	25	50
3	23	49	72	15	34	58
4	**38**	**57**	76	**25**	**40**	**63**
5	37	**57**	**77**	**25**	**40**	**63**

Backprop length	+ MIM loss	Seq len 200			Seq len 800		
		$1m$ 10°	$1m$ 90°	$2m$ 90°	$1m$ 10°	$1m$ 90°	$2m$ 90°
32	no	25	48	72	15	31	57
t/4	**no**	**38**	**57**	**76**	**25**	**40**	**63**
100	no	20	46	71	12	28	53
t/4	yes	7	27	53	4	17	39

performance saturating starting from N=4 segments. We also explore different backpropagation lengths and find that using a length equal to the segment length performs well. This also allows for an efficient implementation, where at the end of each segment, memory is saved and detached and losses are applied for this segment. Finally, we study what happens if we also add a Masked Image Modeling loss like (Sariyildiz et al., 2025a) and find it detrimental to performance: we hypothesize that such auxiliary loss help their training stability but does not help the task itself: in our case, training stability is achieved thanks to memory distillation.

E MEM-RPE WALL-CLOCK RUNTIME MEASUREMENTS

In order to showcase the benefit of recurrent transformers with $\mathcal{O}(1)$ update cost, compared to history transformers like the LBHT Teacher that exhibits quadratic complexity with T, we plot in Figure 8 the runtime and GPU memory required to process sequences of increasing lengths. Timings are measured on a NVIDIA A100 GPU. We observe the quadratic complexity with sequence length T of History Transformers in terms of runtime (left), while *Chimera* has a constant time of around 7ms between two timesteps. Considering memory, *Chimera* has a constant usage, while History Transformers need to store the full history of frames, thus the linear complexity. Please note that FlashAttention (Dao et al., 2022) lowers memory usage, reducing transformers complexity scaling with T from quadratic to linear.

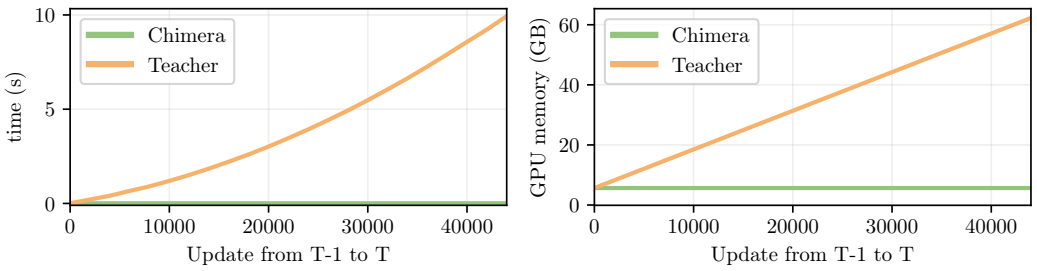


Figure 8: **Inference timings and memory requirements** for our *Chimera* model and a LBHT Teacher. Recall that *Chimera* and *Kinaema* have the exact same architecture and therefore identical computational complexity and wall clock runtimes.

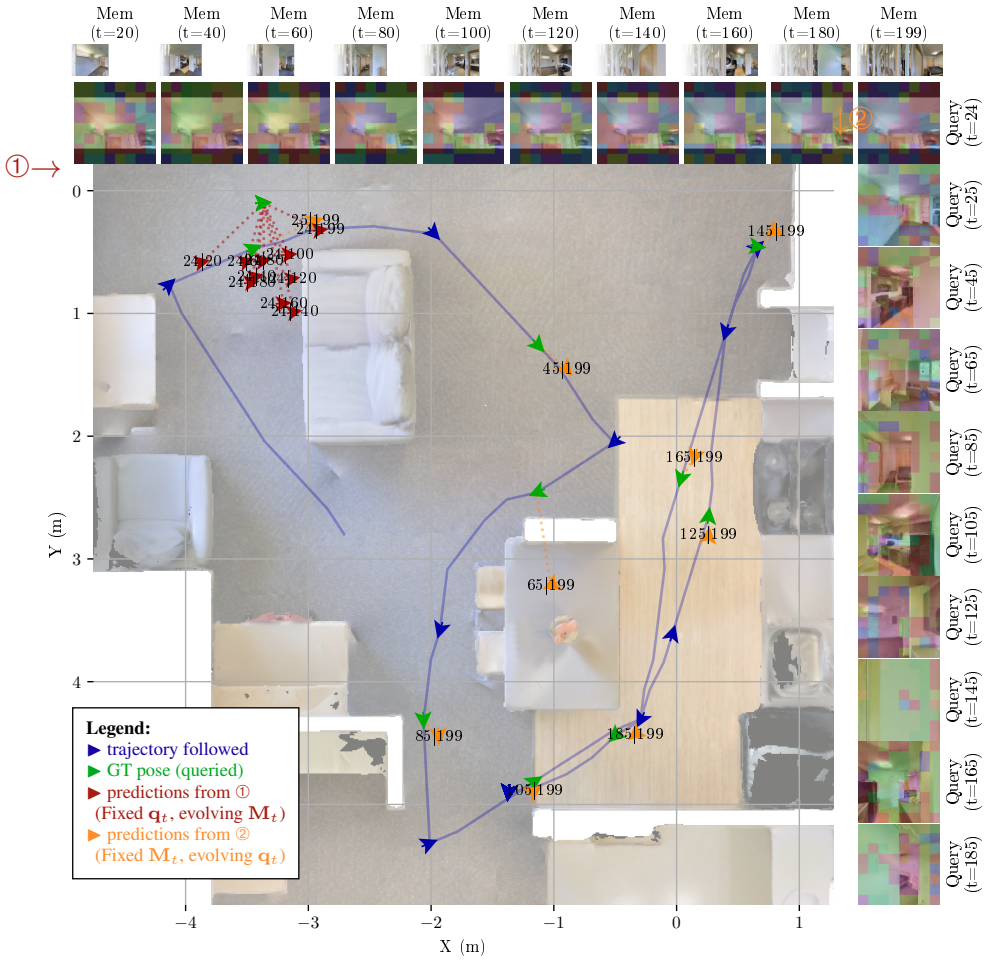


Figure 9: Attention distributions over memory tokens for ①⟶ a fixed query against an evolving memory, and ↓② evolving queries against fixed memory extracted at the end.

F Mem-RPE qualitative examples

Figures 9 to 12 provide Mem-RPE qualitative examples. They show the evolution of attention patterns of *Chimera* for one trajectory per Figure, for ①⟶ a fixed query against an evolving memory, and ↓② evolving queries against fixed memory extracted at the end. There does not seem to be a strong correlation between memory tokens and scene semantics, *e.g.* walls, floor, windows can trigger different tokens in different trajectories. We observe some stability and coherence along a specific trajectory, as dominant memory tokens will follow the motion of the camera, and a fixed query will also mostly conserve its attention patterns against an evolving memory. It is also worth noting that some tokens seem to receive more attention overall, as if the model had "default registers" regardless of the input observations. These effects are more noticeable in animations where query and memory steps are not sub-sampled. We refer the interested reader to the videos provided as supplementary material.

G Details on Streaming Visual Question Answering

This section provides the architecture, data, and training details for the video-language application (Section 5). These experiments complement our main Mem-RPE study by testing the same principle in a different prediction setting: first learn to compress an accessible observation history into a bottleneck, then use this representation to supervise the hidden state of a recurrent model.

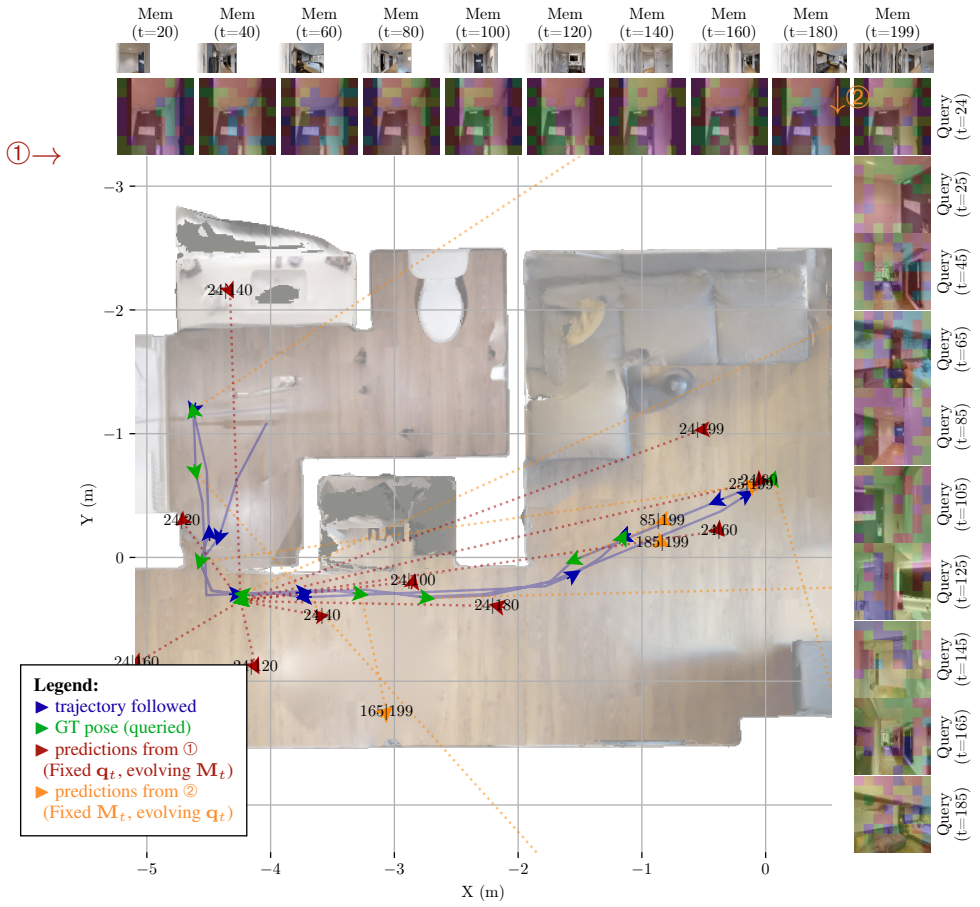


Figure 10: Attention distributions over memory tokens for ①⟶ a fixed query against an evolving memory, and ↓② evolving queries against fixed memory extracted at the end.

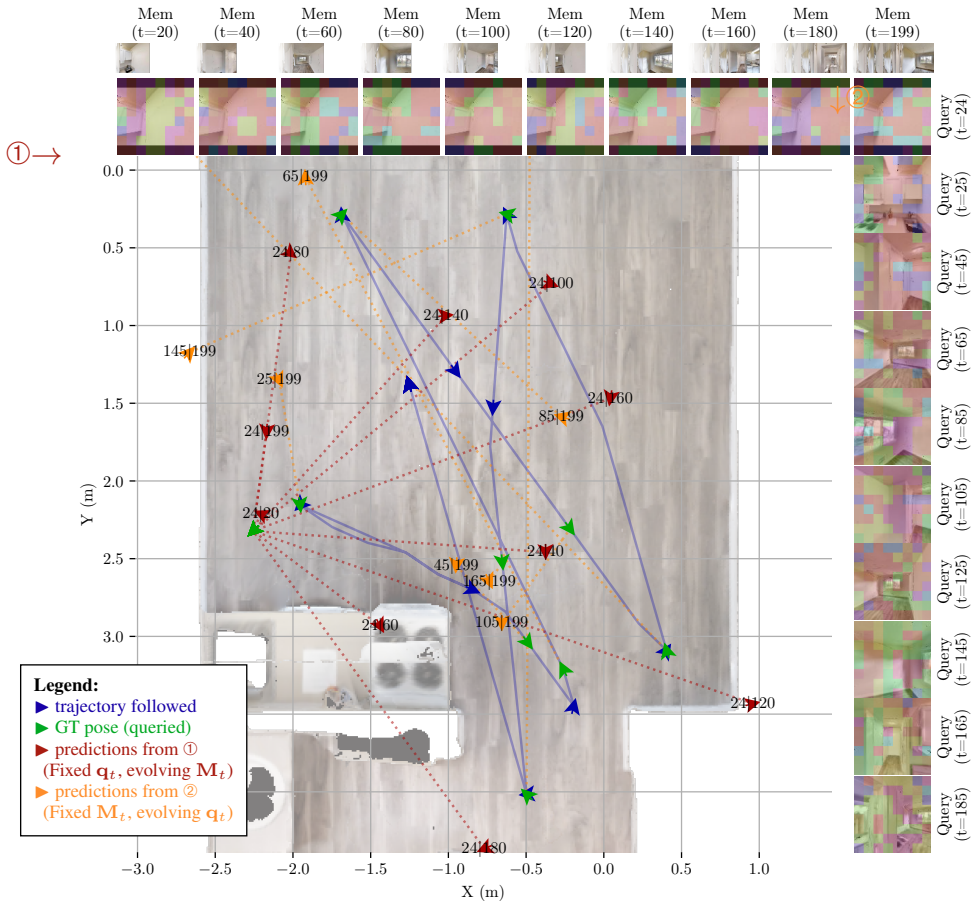


Figure 11: Attention distributions over memory tokens for ①⟶ a fixed query against an evolving memory, and ↓② evolving queries against fixed memory extracted at the end.

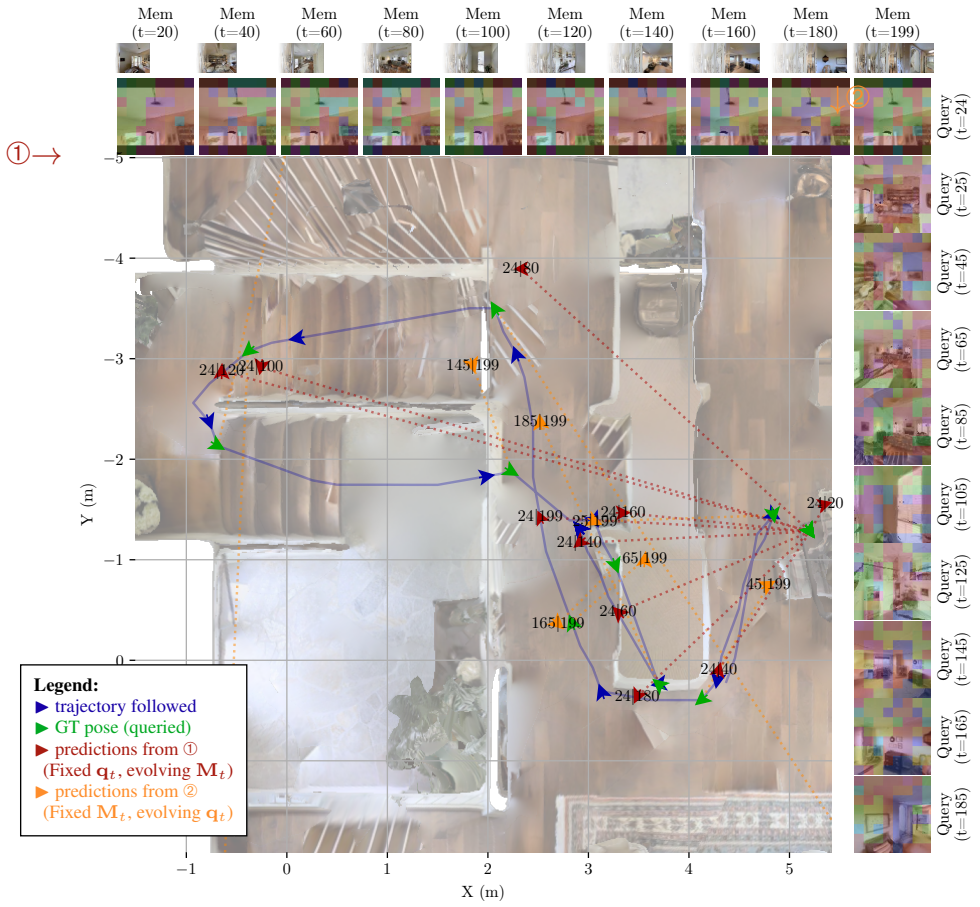


Figure 12: Attention distributions over memory tokens for ①⟶ a fixed query against an evolving memory, and ↓② evolving queries against fixed memory extracted at the end.

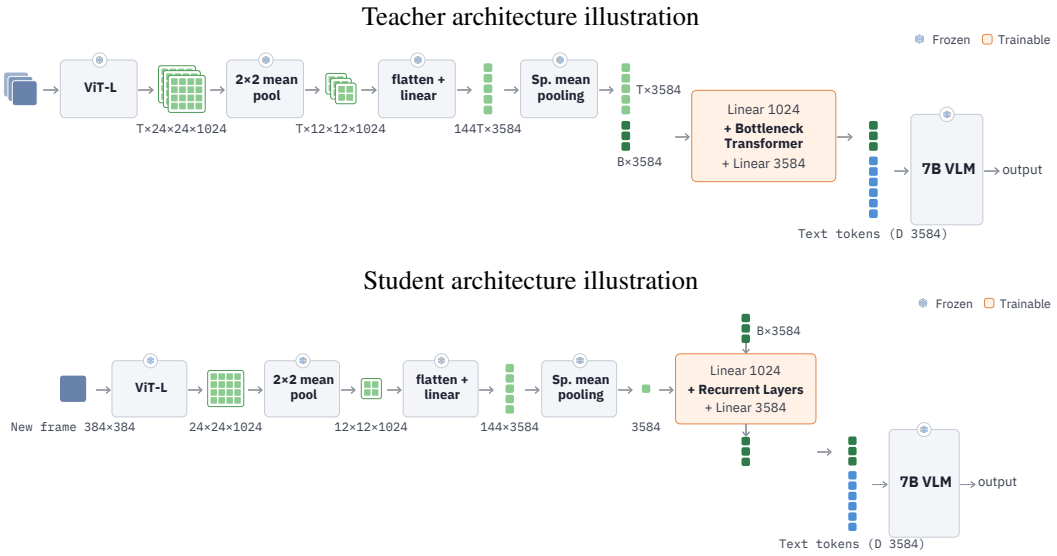


Figure 13: Illustration of the teacher (top) and student (bottom) architectures for Streaming Visual Question Answering.

G.1 Visual representations and model adaptation

We start from LongVA-7B (Zhang et al., 2025a), whose language model has been trained for long contexts and has shown to transfer well for processing long video inputs. Its ViT-Large visual encoder, 2×2 spatial pooling, and projection to 3584 dimensions (the one from the language model) produce 144 visual tokens per frame from video inputs resized to 384×384px. In our experiments, we average these tokens spatially to obtain one global representation per frame. This additional pooling reduces the input to the temporal model to one token per observed frame. The visual encoder and language model remain frozen during teacher and student training.

The teacher uses six non-causal transformer layers of width 1024. It processes the frame tokens together with $B{=}32$ learnable bottleneck tokens, whose contextualized outputs provide the visual representation for the language model. The recurrent student processes the frame representations one by one and replaces joint history processing with a fixed-size memory update. In both cases, temporal compression is independent of the query *i.e.* of the question. The resulting teacher or student representation is mapped to the language model's input space and combined with the textual question for answer prediction. Figure 13 summarizes these two paths.

G.2 Data and temporal sampling

Training data. We train on LLaVA-Video (Zhang et al., 2025b), a collection of 178,510 videos with video-language annotations. The videos' duration ranges from 5 to 180 seconds. Frame features are extracted at 6 fps, and training uses a random stride $s \in \{1, \ldots, 12\}$ over the cached features. The effective frame rate thus ranges from 0.5 to 6 fps. At the highest sampling rate, a three-minute video contains up to 1080 frame tokens. A subset of LLaVA-Video is kept for validation using multiple-choice questions.

Evaluation data. Streaming evaluation uses OVO-Bench (Niu et al., 2025) and the 'Real' task of StreamingBench (Wu et al., 2024) (the 'real' subset mainly focuses on visual content; the other tasks involve sequential question answering where later questions refer to entities or events encountered earlier, with the 'omni' subset mostly based on audio). Predictions use the video observations available at query time. We also report offline results on the held-out LLaVA-Video multiple-choice subset and on Video-MME (Fu et al., 2025). In the offline setting, a complete video is available before answering, whereas the streaming setting requires answers from successive prefixes.

Table 7: **Bottleneck teacher ablations on video question answering** on the number of bottleneck tokens (B) and the depth (d) of the transformer. Results are reported on both non-streaming (LLaVA-Video val & Video-MME) as well as streaming benchmarks (OVO-Bench, StreamingBench Real task).

Model	Llava-Video (val)	Video-MME	OVO-Bench	StreamingBench (real)
no vision	57.1	37.1	33.5	39.3
non-bottleneck transformer (d=6)	73.4	38.8	35.5	50.3
LBHT (B=8,d=6)	71.9	41.1	34.9	50.3
LBHT (B=16,d=6)	72.2	41.6	35.6	48.6
LBHT (B=32,d=6)	72.4	40.6	37.0	49.4
LBHT (B=64,d=6)	71.7	41.8	34.5	50.0
LBHT (B=32,d=9)	71.2	40.7	36.1	49.5

G.3 Training the teacher and student

The teacher is first trained with the vision-language answer-prediction loss. The frozen language model provides supervision for learning which visual information to retain in the bottleneck. The teacher's visual history encoder is non-causal over the available prefix, but cannot access observations beyond the endpoint being supervised.

Student training follows the multi-segment distillation strategy. For a video represented by T frame tokens, we use $N{=}4$ endpoints, approximately $e_n = nT/4$ for $n \in \{1, 2, 3, 4\}$. The frozen teacher is run independently on each prefix $\{\mathbf{o}_1, \ldots, \mathbf{o}_{e_n}\}$ to obtain its bottleneck representation. One sequential student rollout supplies the corresponding recurrent memory states. We combine the VQA loss with L_1 matching of the aligned teacher and student representations, as in Equation 4. Compared to the Mem-RPE setting, the task loss is only applied at the final step. The baseline without distillation uses the same recurrent architecture and VQA supervision, with $\lambda_{dist} = 0$.

G.4 Teacher ablations

Table 7 reports performance for various LBHT configurations, varying the number of bottleneck tokens B, and the transformer depth d. We first report a no-vision version, where the frozen vision-language model input is limited to text tokens. We also consider a transformer without bottleneck, for which the language model receives the processed tokens corresponding to all input frames.

On the held-out LLaVA-Video subset, the non-bottleneck transformer achieves 73.4%, compared with 72.4% for the default $B{=}32$, $d{=}6$ teacher. Bottlenecks with 8 and 16 tokens achieve 71.9% and 72.2%, respectively. Increasing further the number of bottleneck tokens or the depth of the transformer do not bring noticeable gains. We use the 6-layer, 32-token non-causal model as the distillation teacher.

Although the bottleneck limits the number of visual tokens passed to the language model, its history transformer still processes a sequence whose length grows with the video. In streaming use-cases, non-causal history representations must be recomputed for successive prefixes; the frozen per-frame features can be cached, but the temporal transformer cannot reuse its previous contextualized representations in the same way as a causal KV cache. In contrast, the recurrent student carries its fixed-size visual memory forward. This provides bounded visual-history storage and constant-cost memory updates at a fixed model size; it does not establish an offline wall-clock speed advantage, since a history transformer can process a complete video in parallel.

We provide a few qualitative examples in the supplementary video. We refer to Figure 14 for an example snapshot explaining the layout. Overall, we observe that the student trained with our approach maintains the teacher performance and it is better able to answer to questions than the student trained without distillation.

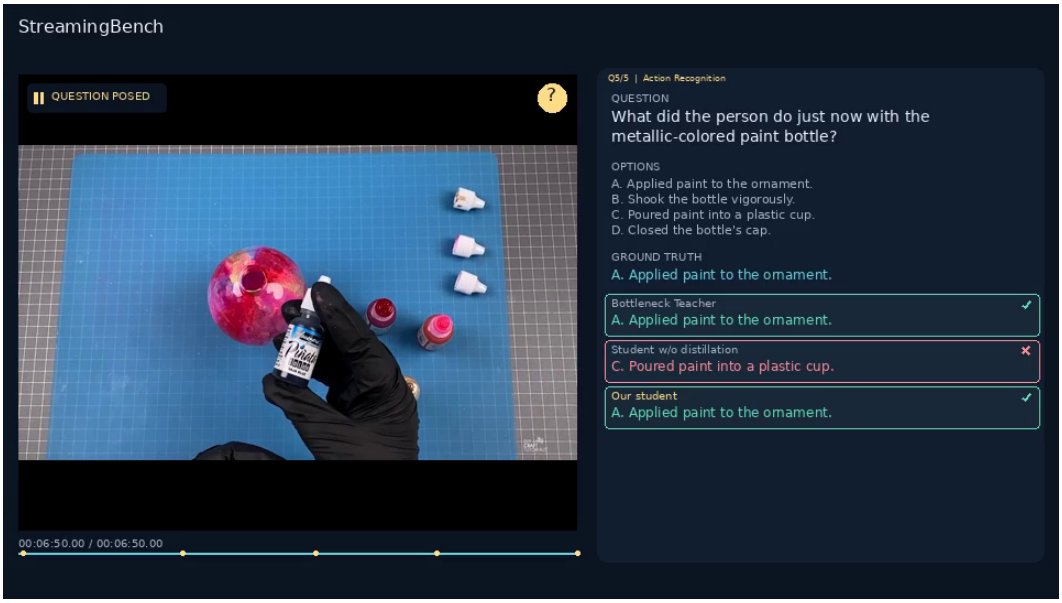


Figure 14: **Qualitative examples of Streaming Visual Question Answering.** The supplementary video shows a few result comparisons on the StreamingBench-real benchmark. The video is shown on the left (played at $\times 10$ speed) and stopped when there is a question. At this moment, the video is paused during a few questions for an interested reader to be able to see the multiple-choice question, the list of answer choices, the ground-truth as well as the answer provided by (a) the bottleneck teacher, (b) the student trained without distillation and (c) 'Our student' which is trained with memory distillation from the bottleneck teacher.