THE TIME MACHINE: ON THE POWER OF MOTION FOR EFFICIENT PERCEPTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Video representation learning has seen tremendous progress in recent years. This has been driven by many factors, including the scale of training and the success of self-supervised models trained on next-frame prediction. While these factors have pushed the boundaries of what video models can do, they also introduce their own set of limitations. First, scaling video models can reach prohibitive costs, with recent models needing hundreds of years of video to be trained. Second, learning to predict the next frame (or embedding of the frame) naturally focuses on spatial information and as a result, video models still struggle with temporal understanding. In this paper we propose a novel approach that uses motion as a modality to alleviate both of these core issues. Given the motion in a video in the form of point-tracks, we mask some of the tracks and use a masked autoencoder to reconstruct the missing tracks. This allows us to learn a representation in a self-supervised manner which we call *TIME* (Temporally-Informed Motion Embedding), and is focused on capturing temporal information. Also, as motion is inherently appearance-invariant, TIME needs far fewer examples to generalize well. As a result, without bells and whistles, on temporal tasks TIME performs on par with state-of-the-art models, using up to 4 orders of magnitude less training data. For general tasks, TIME can be used in combination with existing representations, and we observe that it leads to a significant improvement for V-JEPA 2, RVM and VideoMAE on standard benchmarks such as SSV2, EgoExo4D and Diving48. These results point to a new promising video paradigm for both more temporally-aware as well as more scalable models.

1 INTRODUCTION

Video Understanding has seen great progress in the last few years. Some tasks, such as action recognition (e.g., Kinetics (Carreira & Zisserman, 2017)) or general question-answering (e.g., NExT-QA (Xiao et al., 2021)) can be performed extraordinarily well by current video models.

This impressive progress has been possible thanks to several factors. One of the key factors is clearly the scale of training data. Similar to other areas in vision, the transition from pre-training on a single large dataset (e.g., Sports1M (Karpathy et al., 2014)) to pre-training on vast amounts of data (Assran et al., 2025) has brought the capabilities of video models to a new level. However, scaling video training presents its own limitations in the trade-off between cost and accuracy. For example, recent examples of models (Assran et al., 2025) show that increasing the size of the training data by millions of videos may only improve accuracy by 1-2%, pointing to a possible performance plateau (Bordes et al., 2025) despite the use of larger and more diverse datasets and scaling laws.

Another key factor for progress in recent years has been the success of models trained on next-frame or next-frame-embedding prediction (Assran et al., 2025; Zoran et al., 2025). These self-supervised models avoid the limitation of using language as a supervision signal (Radford et al., 2021), which may limit the range of learned concepts to those appearing in the captions used for training. In these self-supervised models spatial accuracy is explicitly rewarded, but temporal accuracy is only learned as a byproduct. As a consequence, representing temporal dynamics and especially detailed fine-grained motion information may pose a challenge for these models.

The struggle to capture temporal dynamics has also been observed in Multi-modal Large Language Models (MLLMs), with a variety of new benchmarks illustrating this problem (MotionBench (Hong

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.

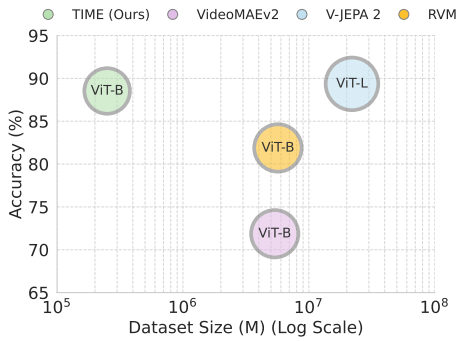

Figure 1: Model performance on SSv2 "Arrow of Time" task performance. Our TIME model achieves "appearance-free" action classification performance on-par with state-of-the-art V-JEPA 2 model despite using several magnitudes less pre-training data.

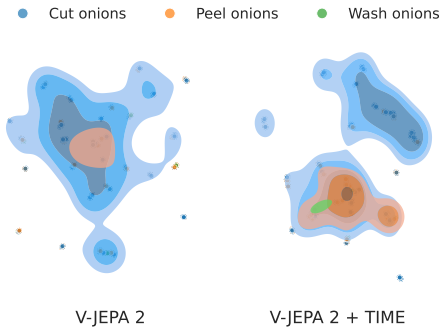

Figure 2: UMAP plot ($n_components = 2$) showing class separation for V-JEPA 2 vs. concatenating V-JEPA 2 and TIME, on frozen features of videos from Ego-Exo4D. Using TIME greatly improves the separation of similar fine-grained actions.

et al., 2025b), TempCompass (Liu et al., 2024), VideoMME (Fu et al., 2025), VLM4D (Zhou et al., 2025)).

In this paper we make the observation that these two issues in video can be addressed with a key tool: learning video representations from motion. In particular, we use the motion of a video, in the form of point-tracks (sparsely sampled) as the only input modality to a masked-autoencoder. The input tracks are masked, and the training objective is to reconstruct the missing tracks. Crucially, the model is trained exclusively on synthetic point trajectories from rigid-body physics simulations from Kubric (Greff et al., 2022), and has not seen any real-world video. The learned representation, which we call TIME *(Temporally Informed Motion Embedding)*, can then be used for a variety of tasks in a zero-shot manner. In addition, it can be used as a complementary modality for appearance-based video models (e.g., VideoMAE (Tong et al., 2022), V-JEPA 2 (Assran et al., 2025)) showing that the information that TIME captures is complementary, and non-overlapping with these appearance-based models. The benefits of TIME are twofold.

First, we experimentally observe that TIME improves performance on standard benchmarks (e.g., SSv2, Ego-Exo4D) when added to state-of-the-art models such as V-JEPA 2 (Assran et al., 2025) and RVM (Zoran et al., 2025). This is particularly striking in fine-grained tasks within Ego-Exo4D, where the use of TIME improves performance up to 18%. This confirms that fine-grained temporal information is hard to capture by appearance-based models. We also test the performance of TIMEon its own. Since there are no established benchmarks that focus solely on temporal information, we test on a split of two existing benchmarks: CLEVRER (Yi et al., 2020) and Something-something V2 (SSv2) (Goyal et al., 2017), as it is often the case for temporal classes (Bagad & Zisserman, 2025). For SSv2 we test on classes that can be discriminated without image information (e.g., opening vs. closing). For CLEVRER we also test on tasks that do not require temporal information, such as collision timing. The splits and tasks will be available upon publication. Across tasks, TIME matches or outperforms state-of-the-art models such as V-JEPA 2 and RVM.

Second and crucially, learning from motion reduces the amount of data needed to train the model by up to 4 orders of magnitude (measured in hours of video training data), compared to the same state-of-the-art appearance-based video models. This reduction stems from several facts: 1) Motion can be learned from synthetic data as the domain gap between natural and synthetic motion is surprisingly small (Dosovitskiy et al., 2015). In other words, noise-less synthetic data allow models to converge faster than noisy real-world data. 2) Low-dimensional motion is more discriminative than low-dimensional images (Davison et al., 2026; Boduljak et al., 2025). This implies that motion can be represented as a sparser set of points (e.g., a 32×32 point trajectory grid) instead of a high-resolution equivalent (e.g., 256×256 pixel image) enabling more efficient training. 3) Learning strong temporal information within an appearance-based model is hard, as most training tasks can be solved relying heavily on appearance information, even when trained on extremely large amounts

of data. Hence, separating the learning process of temporal information guarantees that the model cannot "cheat" by looking at pixel values or other appearance representations.

In summary, the proposed TIME video representation provides a novel, highly scalable and compute-efficient paradigm for future research in video understanding.

2 Related Work

Self-Supervised Video Models. The first video models trained with large-scale datasets (Karpathy et al., 2014; Carreira & Zisserman, 2017) used classification supervision for training. While these models (Carreira & Zisserman, 2017; Bertasius et al., 2021) have worked well, they require large amounts of labeled training data, which is expensive to produce. In addition, the rise of "foundation models" in other domains such as language or image (Caron et al., 2021), that are designed to be used in a "zero-shot" fashion, without the need to fine-tune, has created increasing interest in self-supervised video models. Research in this area has led to a wide variety of pre-text tasks. Some works (Recasens et al., 2021; Ranasinghe et al., 2022) have explored the use of spatial and temporal transformations, in combination with a contrastive learning paradigm where a video and a transformed copy should be represented as similar as possible. While this works well, the contrastive paradigm can be sensitive to the choice of negative pairs. In the search of more efficient video model pre-training, the notion of masked autoencoders (MAE) (He et al., 2021) in image domain have been adopted for self-supervised video model pre-training tasks as well. More precisely, works such as VideoMAE (Tong et al., 2022; Wang et al., 2023) propose models that learn effective spatiotemporal representations by reconstructing masked portions of raw frame spatio-temporal patch tubes as their primary training objective. While such models achieve state-of-the-art results in some video understanding tasks, the very nature of such data modelling inherently biases the network to memorize high-frequency spatial details (textures, colors) rather than abstract motion. To address this issue, research works on learning efficient video representations include joint-embedding predictive architectures (JEPA) that forecast missing latent spatiotemporal regions (Assran et al., 2025). Authors reason that such a model is no longer forced to to reconstruct exact pixels, therefore it is forced to learn high-level semantic and kinematic abstractions instead. However, despite pre-training on 114 years-equivalent of curated video data, such models severely underperform in physical understanding tasks (Bordes et al., 2025) with evaluation performance being close to a random chance. While latest self-supervised learning works introduce an explicit sparse motion signal during full model pre-training as an effective way to encode temporal dynamics in the learned representations (Vandeghen et al., 2026), the sparsity of the signal and the requirement to retrain the entire model from scratch hinders its applications further.

Challenges in Temporal Understanding. Learning temporal information has been a long-standing challenge for video research. Historically, video models have tended to have stronger image understanding than temporal understanding capabilities. There are potentially several reasons for this, one of which is the challenge of designing tasks that force models to learn temporal information, without resorting to image information alone (Sevilla-Lara et al., 2021; Bagad & Zisserman, 2025). Even designing benchmarks that allow us to measure temporal capabilities alone is not trivial. Several datasets (Goyal et al., 2017; Li et al., 2018; Chen et al., 2024) have tried to focus on actions where appearance carries limited information. Others have explored tasks that are action independent such as counting (Dwibedi et al., 2024), skill determination (Doughty et al., 2018), adverb prediction (Doughty et al., 2020; Moltisanti et al., 2023), for tasks that are fairly object-independent such as the reasoning benchmark CLEVRER (Yi et al., 2020). Overall, solving these remains an open problem, as for example the accuracy of top models on the rather "temporal" Something-something dataset published in 2017, still only achieves values around 75%, while accuracy on the rather "appearance-based" Kinetics published the same year achieves around 90%.

In the new era of Multi-Modal Language Models (MLLMs), new challenges in temporal understanding have arisen. The key obstacles might be different now, and relate to the reliance of language in the learning objective, or to token dilution. Still, it has been noted that MLLMs notoriously struggle with understanding temporal concepts (Upadhyay et al., 2025; Xue et al., 2025), and there have been a large variety of multi-modal benchmarks (TempCompass (Liu et al., 2024), MotionBench (Hong et al., 2025a), TemporalBench (Cai et al., 2024), VideoMME (Fu et al., 2025), VLM4D (Fu et al.,

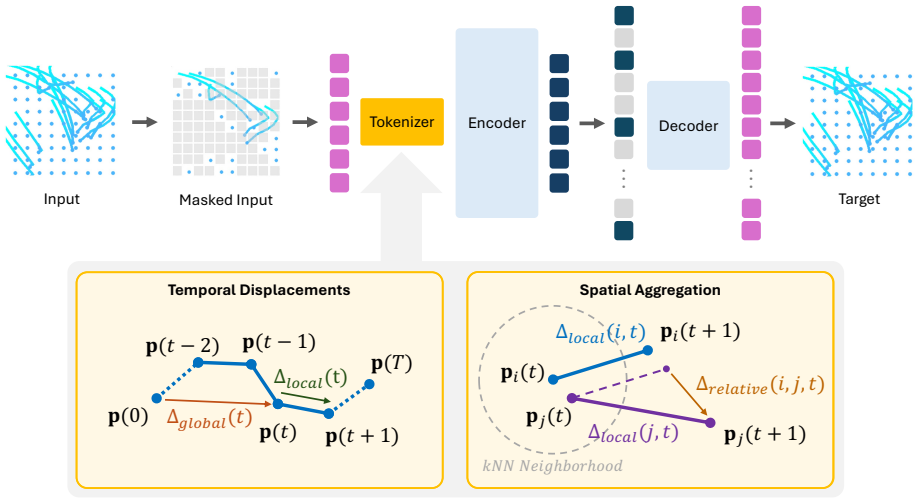

Figure 3: TIME Architecture. Given a set of point trajectories, the model groups them into tubelets and encodes them into trajectory tokens using a tokenizer. During training, we mask 75% of the tokens and only the visible ones are passed through the encoder. A decoder then attempts to reconstruct the masked trajectory tokens from the encoder output. The training loss is computed only on the masked trajectories.

2025)). All these efforts point out that temporal understanding is very much a one of the main challenges in video research today, across different technology paradigms.

Learning Motion Models from Synthetic Data. While learning from synthetic data might seem like a limiting choice due to the sim-to-real domain gap, such limitation is significantly less restrictive for temporal kinematics than it is for spatial appearance. In fact, there is a long history of learning motion from synthetic data that shows surprisingly good results. One of the first success stories was FlowNet (Dosovitskiy et al., 2015), where an optical flow model was learned from the FlyingChairs dataset, consisting of 3D models of chairs moving around with an image background. Despite its simplicity or lack of realism, training on this dataset was a breakthrough on what could be learned from purely synthetic data, which led to other work leveraging this insight (Huang et al., 2023). In a different domain, point-tracking systems have also leveraged this. For example, the state-of-the-art dense point tracker model CoTracker (Karaev et al., 2024) is also primarily trained using synthetic data from a simulated environment (Greff et al., 2022) without loss of real-world generality. Work in generative video modeling argue that current video model inability to model physics correctly arises from the overhead of generating pixels instead of motion, therefore, they rely on the very same synthetic Kubric environments and point tracking modules to obtain quasi-dense trajectories for the task of forecasting the future trajectories (Boduljak et al., 2025) from a single input image. Remarkably, coupled with a single-frame spatial context signal from DINO (Caron et al., 2021), such models surpass expensive generative video models.

However, these prior methods exclusively use synthetic data to solve low-level vision tasks (e.g., pixel displacement or point tracking). In this work, we go one step beyond to demonstrate that purely synthetic simulations can be used to learn high-level semantic video representations that are generalizable and directly transferable to real-world scenarios. While we initially presumed a real-world fine-tuning stage would be necessary to bridge the gap to complex human actions, surprisingly, our synthetic kinematic model transfers entirely zero-shot to real-world videos (e.g., SSv2 (Goyal et al., 2017)). Operating out-of-the-box, it not only matches, but frequently outperforms and complements massively scaled, foundation video models.

3 METHODS

Architecture Overview. We now describe the details of the proposed TIME architecture, which is illustrated in Fig. 3. Recall that the goal of this architecture is to learn a video representation that



Figure 4: **Sample Training Scene.** Given the input tracks of a scene created with Kubric (Greff et al., 2022), the proposed architecture is able to fill in the gaps and estimates the masked tracks using a masked autoencoder, with high fidelity.

will capture the temporal content of the video. For this, we use a self-supervised training paradigm, which allows us to bypass the restrictions of learning from video captions. In particular, given the point tracks of a scene, sampled sparsely but in a grid, we mask a large set of them spatially (i.e., we mask the full point track, not just a portion or segment in time). We convert these point tracks to tokens, using a tokenization process to capture both spatial context of the tracks as well as local and global temporal information (details described below). These tokens are input to a masked autoencoder, similar to previous work such as VideoMAE, composed by an encoder and decoder. This autoencoder is trained using the full set point tracks as target for the reconstruction. At inference time, given a video, we compute the point tracks over the entire scene using point tracking methods (Karaev et al., 2024), and use them as input to the tokenizer, and in turn to the encoder. The resulting output from the encoder is a feature vector with strong capabilities to capture the temporal information of the scene. We now describe each of these components in detail.

From Point Tracks to Tokens. Our model operates over a grid of point tracks sampled uniformly. In our experiments, we use a grid of 32×32 point tracks, over a length of 24 frames. Therefore the input to our model is $N = 1024$ points. Let $p_i(t) = (x, y)$ be the location in image coordinates of each of these points p_i at time t. For every point p_i we compute its corresponding token s_i (for segment) to include essential information that may be useful for reconstruction. First, we compute the local displacement of the point to the next time step, $\Delta_{\text{local}}(i,t) = p_i(t+1) - p_i(t)$. Note this is essential, as using as input absolute point locations instead of relative displacements leads to collapse during training. Another failure mode could be for the network to accumulate small errors over a point track, in the end leading to a poor estimate. To prevent this, we incorporate information in the tokens about the global displacement $\Delta_{\text{global}}(i,t) = p_i(t) - p_i(0)$. This process is described in Fig. 3 in the "Temporal Displacements" box. Capturing spatial structure and geometry is also helpful for point track reconstruction, for example for establishing motion boundaries. To incorporate this information, we take inspiration from the literature in point cloud self-supervised learning (Pang et al., 2022). In particular, for a given point in time $p_i(t)$ and its given local displacement $\Delta_{\text{local}}(i,t)$, we compute how much the displacements of the spatial neighborhood around p_i deviate from its local displacement. This process is described in Fig. 3, in the box "Spatial Aggregation". Specifically, given the K nearest neighbors $p_j(t)$ of anchor point $p_i(t)$, we compute the relative displacement between each pair as $\Delta_{\text{relative}}(i,j,t) = p_j(t+1) - p_i(t+1)$. We do max-pooling over the relative displacements, to obtain the maximum deviation $\Delta_{\text{max-deviation}}(i,t)$, which will also be part of the token. In our experiments we use $K = 16$. Finally, we include an occlusion bit $o_i(t)$, with information of whether point p_i is occluded or not. In summary, for point $p_i(t)$, its corresponding token $s_i(t)$ is computed as:

$$s_i(t) = [\Delta_{\text{local}}(i,t),\ \Delta_{\text{global}}(i,t),\ \Delta_{\text{max-deviation}}(i,t),\ o_i(t)]. \tag{1}$$

Factorized Spatiotemporal Attention Backbone. Following the tokenization steps, we feed the resulting tokens into a model largely based on VideoMAE (Tong et al., 2022). Our encoder and decoder contains 12 and 4 layers respectively. In addition, we factorize the attention to improve computational efficiency. More specifically, as the original ViT relies on *full spatio-temporal attention*,

its time complexity is expensive at $O((T \cdot S)^2)$. We leverage successful applications of *factorized attention* for point tracking (Karaev et al., 2024) and image-to-trajectory forecasting (Boduljak et al., 2025), and factorize the attention mechanism as well. First, we compute *spatial attention* across all unmasked trajectory tokens within the same pair of frames. Next, we compute temporal attention across the time dimension for each individual trajectory. The resulting separation of spatial and temporal components reduces attention time complexity to $O(T \cdot S^2 + S \cdot T^2)$ leading to a five-fold reduction in training time. Further tokenization details can be found in Appendix C.4.

Training Objective. The proposed model is trained to reconstruct masked trajectory tokens from visible context (Fig. 3). During training, we mask 75% of the spatial trajectory tokens and apply this mask across the entire temporal dimension. The encoder processes only the visible tokens and produces a representation of the video. The decoder then inserts learnable mask tokens for the missing trajectory positions and reconstructs the full sequence.

Loss Function. We employ the Huber loss $\mathcal{L}_{\text{Huber}}$ (Smooth L_1) over masked tokens. We compute the loss of local and global displacements separately as they naturally have very different numerical values, and need to be weighted differently. We use a hyperameter λ_{global} to weight down the values of the global displacements ($\lambda = 0.5$). The resulting loss target ensures a balance between local high-frequency kinematics and full temporal length-wide global motion:

$$\mathcal{L}_{\text{target}}(i) = \mathcal{L}_{\text{Huber}}\Big(\hat{\Delta}_{\text{local}}(i), \Delta_{\text{local}}(i)\Big) + \lambda_{\text{global}}\, \mathcal{L}_{\text{Huber}}\Big(\hat{\Delta}_{\text{global}}(i), \Delta_{\text{global}}(i)\Big). \tag{2}$$

The natural distribution of displacements is also highly imbalanced. Since over 80% of points in the training data are static, there is a risk that the model might predict everything as static. Hence, we also weigh non-static points with a higher weight ω, computed as:

$$\omega(i) = \begin{cases} \gamma & \text{if } \|\Delta_{\text{global}}(i)\|_2 > \tau \\ 1.0 & \text{otherwise} \end{cases} \tag{3}$$

where τ is a pre-defined static motion threshold ($\tau = 0.002$), and $\gamma = 7.0$ represents the motion boost hyperparameter. The final reconstruction loss is calculated as the weighted average across all masked trajectory tokens:

$$\mathcal{L} = \frac{1}{\sum_i \omega^{(i)}} \sum_i \omega^{(i)}\, \mathcal{L}^{(i)}_{\text{target}}. \tag{4}$$

Additional loss function design details are available in Appendix C.3.

Training Data. To train our model, we modify the Kubric simulation environment (Greff et al., 2022) to obtain a uniform grid of point trajectories. This choice has several advantages. First, it generates perfectly accurate and clean data i.e., without the noise from the real-world tracks obtained using point tracking. This helps the training procedure making it more stable and faster to converge. Second, using generated data allows us to speed up the generation of point tracks, as instead of estimated they are directly computed. We produce 250K videos using MOVi-B dataset generation pipeline consisting of scenes with static camera and a random number of geometric primitives and simple shapes colliding in the middle of the scene. Further settings can be found in Appendix D.1.

Training Details. Our largest TIME model is trained on 250K synthetic samples for 100 epochs, using 3 warm-up epochs with an effective batch size of 40. We set the learning rate to 1.5e-4 and use the AdamW optimizer with cosine annealing learning rate scheduling technique with a weight decay of 0.05. Training using 4 NVIDIA A100 80GB GPUs takes around 120 hours ($\approx$ 5 days). Full training details can be found in Appendix D.3.

4 Experimental Results

We evaluate the proposed representation to answer several questions. First, we test whether in fact TIME is able to capture temporal information that is general enough to succeed in a zero-shot task. To this end we re-purpose standard video datasets (SSv2 (Goyal et al., 2017), CLEVRER (Yi et al., 2020)) to create tasks where appearance is not necessary, and motion carries all the information needed for the task. Section 4.3 describes the details of these experiments.

Table 1: **Video Understanding Tasks.** We report top-1 accuracy across TIME features fused with Ego-Exo4D, SSv2, and Diving48. Δ denotes the improvement over the baseline. Standalone model performance can be found in Appendix A.2.

	Ego-Exo4D		SSv2		Diving48	
Model	**Top-1 ($\pm$ Std %)**	Δ	**Top-1 ($\pm$ Std %)**	Δ	**Top-1 ($\pm$ Std %)**	Δ
TIME + CLIP (4f)	39.33 ± 0.24	**+14.22**	31.56 ± 0.05	**+13.00**	16.11 ± 0.28	**+2.14**
TIME + DINOv3 (4f)	39.63 ± 0.22	**+14.75**	34.60 ± 0.10	**+12.65**	18.08 ± 0.21	**+1.73**
TIME + RVM†	37.00 ± 0.17	**+11.32**	41.92 ± 0.06	**+1.84**	11.74 ± 0.11	**+2.53**
TIME + VideoMAEv2†	51.14 ± 0.23	**+10.12**	38.74 ± 0.09	**+5.69**	24.97 ± 0.24	**+0.73**
TIME + V-JEPA 2†	45.60 ± 0.12	**+10.75**	50.00 ± 0.10	**+0.22**	30.04 ± 0.15	**−0.04**

† Denotes video models whose pre-training data explicitly includes the SSv2 dataset.

Second, for general visual tasks, we explore how well the TIME embedding complements other video models. For this we experiment on full standard benchmarks (such as SSv2 (Goyal et al., 2017), Ego-Exo4D (Grauman et al., 2024), Diving48 (Li et al., 2018)) using state-of-the-art video models and combining TIME features with them. Details are shown in Sec. 4.2.

Finally, in Sec. 4.4 we perform ablation studies to understand the behavior of TIME and what factors (such as dataset size, masking ratio, etc) affect results most.

4.1 Experimental Details

Datasets. We use a suite of standard, diverse datasets to evaluate TIME. For video understanding tasks, we evaluate on standard benchmarks that are generally temporally rich, including: the Something-Something-V2 (SSv2) (Goyal et al., 2017) dataset, the Diving48 (Li et al., 2018) dataset and Ego-Exo4D (Grauman et al., 2024). Note that for Ego-Exo4D, we use all available video takes containing *keystep annotations* (i.e., "Bike Repair", "Cooking", and "Health" scenes), resulting in a set containing 24,060 training and 6,014 validation samples across 620 classes (full details in Appendix F.2). We report average accuracy as well as the standard deviation of 5 separate runs.

For temporal understanding tasks there are no standard datasets. We use a subset of SSV2 and a subset of CLEVRER (Yi et al., 2020), covering the tasks that require strictly temporal information. Further details on the split are given in Sec. 4.3.

Video Model Baselines. As baseline comparisons we choose several video representations. We use VideoMAEv2 (Tong et al., 2022) as the architecture that the proposed embedding is most closely related. This allows us to make comparisons across modalities for comparable architectures. We also use V-JEPA 2 (Assran et al., 2025) and RVM (Zoran et al., 2025) as representatives of state-of-the-art in self-supervised video representations. All models are of ViT-B size-equivalent except for V-JEPA 2 which is ViT-L (refer to Appendix E.2). A complementary comparison against motion-only baselines is available in Appendix A.1.

4.2 TIME for Video Understanding

Description of Video Understanding Tasks. We now explore the use of the proposed embedding for general visual tasks, that require both spatial and temporal information. Our goal here is to test the ability of TIME to provide additional information to standard, appearance-based, video models. To ensure a fair comparison between model features, we maintain the standard linear probing protocol in self-supervised learning - mean pooling over the token sequence. The final resulting feature vector dimensions are available in Appendix E.4. Further experiments with alternative probing can be found in Appendix A.3. Further evaluation design choices can also be found in Appendix C.

Results. Table 1 shows the results. We observe that TIME provides performance gains across datasets and models, showing that it is very much providing complementary information. These gains are particularly noticeable in the in Ego-Exo4D, where TIME improves the performance of V-JEPA 2 close to 11%. This suggest that TIME might be particularly useful for smaller, non-rigid motions. This is a particularly remarkable result because TIME has been trained on orders of

Table 2: **Temporal Tasks**. Comparison of temporal directionality reasoning. Our purely kinematic model (TIME), trained exclusively on 140 hours of synthetic simulations, dramatically outperforms a size-matched RGB model (VideoMAEv2) and maintains high competitiveness against V-JEPA 2, an architecture pre-trained on over $12,000$ times the real-world pre-training data.

	VideoMAEv2† (Wang et al., 2023)	RVM† (Zoran et al., 2025)	V-JEPA 2† (Assran et al., 2025)	TIME (Ours)
SSv2 (AoT)	71.87	81.88	**89.36**	88.53
CLEVRER (T1*)	77.73 ± 0.00	86.56 ± 1.17	92.64 ± 0.25	$\mathbf{93.84 \pm 0.37}$
CLEVRER (T2*)	33.33 ± 0.00	58.17 ± 2.57	58.92 ± 3.40	$\mathbf{74.95 \pm 7.06}$
CLEVRER (T3*)	36.69 ± 0.06	59.10 ± 0.68	$\mathbf{77.84 \pm 1.84}$	61.35 ± 3.33
Modality	224×224 Video	224×224 Video	256×256 Video	32×32 Tracks
Size	ViT-B	ViT-B	ViT-L	ViT-B
Dataset	Real-World	Real-World	Real-World	Synthetic
Training Samples	5.36M (27×)	5.68M (23×)	22M (88×)	0.25M (1×)
Video Hours Eq.	≈7,500 (54×)	≈0.25M (1,785×)	≈1.73M (12,357×)	140 (1×)

† Denotes models whose pre-training data explicitly includes the SSv2 dataset.
* **T1**: Does a collision happen? **T2**: When does the first collision happen? **T3**: Number of collisions

magnitude less data than the other models and because it was solely trained on synthetic data. This result suggests that training image representations and temporal representations independently may be an interesting solution to prevent models from overly relying on appearance information to solve the training tasks. Standalone model performance and additional dataset analyses are available in Appendix A. Additional model efficiency clarification can be found in Appendix G.

4.3 TIME FOR TEMPORAL REASONING

Description of Temporal Tasks. Standard video understanding benchmarks often combine spatial and temporal information. Hence there are no existing benchmarks focusing strictly on temporal information (i.e., the solution to the task requires only temporal information). Here we split the section of SSv2 and CLEVRER that is strictly temporal, to allow us to directly compare the temporal capabilities of video models.

In SSv2 there are many classes that require appearance information to be identified. Thus, we select 14 pairs of classes that have directional labels, such as "Moving something up" vs. "Moving something down", for a total of 28 classes (full details in Appendix F.1). This subset consists of 43,583 videos from the training set and 5,859 videos from the validation set. The task is to classify each pair using frozen features and a linear classifier. We average the accuracy across the 14 binary classifiers. We refer to this task as "Arrow of Time" borrowing the name from previous work (Pickup et al., 2014). Results are shown in Table 2, in the row "SSv2 (AoT)".

We also use the CLEVRER (Yi et al., 2020) dataset. This dataset is also designed to focus on temporal reasoning, but its question-answering would potentially involve the use of a multi-modal model, which could complicate the process of strictly measuring temporal abilities. Instead, we use the meta-data present in the videos to design three tasks: Task 1, Classification ("Does a collision happen?"); Task 2, Detection ("When does the first collision happen?"); Task 3, Counting ("What is the number of collisions?"). Each task is evaluated on the full dataset containing 10,000 training and 5,000 validation videos. The task is similar to before, to use frozen features to train a linear classifier or regressor. Results are shown in Table 2.

Results. The experiments show a surprising finding. TIME has been trained only on synthetic data and on orders of magnitude less data than state-of-the-art models, while those models have been trained on real data, which includes for example the SSv2 dataset. Concretely, compared to the V-JEPA 2 model which is the most expensive to train, the comparison is remarkable, 140 hours of simulations vs. 200 years of real-world video respectively. Yet, TIME performs on par or even surpasses the other self-supervised video models in temporal understanding. For the detection task in particular, the proposed TIME has a gap of over 15% over V-JEPA 2. This is an interesting finding

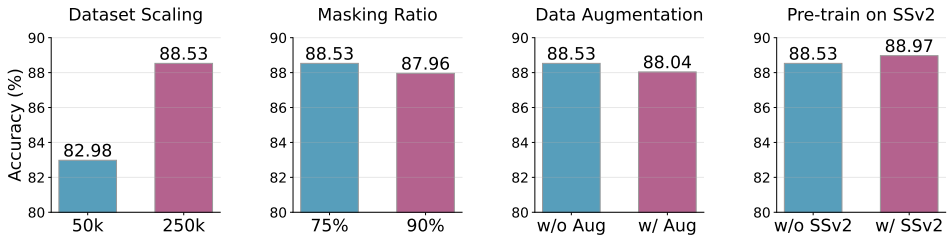

Figure 5: **Ablation study of TIME on the "Arrow of Time" task.** We find that scaling the model from 50k to 250k Kubric samples leads to significant performance gains. Very high masking ratios (e.g., 90% masking used in pixel-based video models) lead to worse performance likely due to point trajectories containing less redundant information than pixels. Simple data augmentation techniques (e.g., simulated camera zoom or panning effect via matrix operations on trajectory grids) do not lead to improved performance. We also experiment with a second stage pre-training on SSv2 (i.e., trajectories extracted using CoTracker3) for 30 epochs leading to a minor performance improvement.

that suggests that despite training scale, current video model training paradigms might overly rely on visual appearance cues, hindering them from learning temporal structure. Instead, TIME is forced to learn a strong kinematic representation, which shows excellent generalization abilities, even to unseen tasks, and real-world data.

4.4 Ablation Study of TIME

We now measure the effect of different aspects of the model on the performance and show the results in Fig. 5. For this we use the "Arrow of Time" task on SSv2 from Sec. 4.3. We observe that the results are rather stable with respect to most parameters. These include: varying the masking ratio during training time; augmenting data using synthetic camera motion (this is, not using the full Kubric pipeline, but simply adding panning to the point tracks); and fine-tuning on 50K videos of SSv2 on the self-supervised task. We observe that the masking ratio and data augmentation have a slight negative effect, while the pre-training on SSv2 have a slight positive effect. We also explore the effect of scaling data and observe that by far it has the strongest effect on performance. Multiplying the amount of training data by 5 yields a significant 5.4% improvement. Additional ablation studies on token components, masking strategies, static camera simulations, and sim-to-real generalization are available in Appendix B.

5 Conclusion

We have presented a self-supervised model called TIME to learn video representations, based purely on temporal information in the form of point tracks. We have shown that learning video representation from motion alone has two core benefits. First, the model can learn to solve temporal tasks extremely well, on par or even surpassing state-of-the-art models trained on up to 4 orders of magnitude more data. Indeed, learning motion patterns proves to be extremely data-efficient, even compared to a very similar pixel-based architecture. Second, we have shown that this temporal model can be combined with appearance features, providing considerable improvements over using appearance alone on a wide range of tasks. We hope this model can be useful to the video research community either on its own for purely temporal tasks or as an add-on, easy to train model, that provides fundamental temporal information for perception.

REFERENCES

Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark, 2016. URL https://arxiv.org/abs/1609.08675.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba, Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-jepa 2: Self-supervised video models enable understanding, prediction and planning, 2025. URL https://arxiv.org/abs/2506.09985.

Piyush Bagad and Andrew Zisserman. Chirality in action: Time-aware video representation learning by latent straightening, 2025. URL https://arxiv.org/abs/2509.08502.

Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1728–1738, October 2021.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video, 2024. URL https://arxiv.org/abs/2404.08471.

Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, 2021.

Gabrijel Boduljak, Laurynas Karazija, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. What happens next? anticipating future motion by generating point trajectories. *arXiv preprint arXiv:2509.21592*, 2025.

Florian Bordes, Quentin Garrido, Justine T Kao, Adina Williams, Michael Rabbat, and Emmanuel Dupoux. Intphys 2: Benchmarking intuitive physics understanding in complex synthetic environments, 2025. URL https://arxiv.org/abs/2506.09849.

Mu Cai, Reuben Tan, Jianrui Zhang, Bocheng Zou, Kai Zhang, Feng Yao, Fangrui Zhu, Jing Gu, Yiwu Zhong, Yuzhang Shang, Yao Dou, Jaden Park, Jianfeng Gao, Yong Jae Lee, and Jianwei Yang. Temporalbench: Benchmarking fine-grained temporal understanding for multimodal video models, 2024. URL https://arxiv.org/abs/2410.10818.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6299–6308, 2017.

Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset, 2022. URL https://arxiv.org/abs/1907.06987.

João Carreira, Dilara Gokay, Michael King, Chuhan Zhang, Ignacio Rocco, Aravindh Mahendran, Thomas Albert Keck, Joseph Heyward, Skanda Koppula, Etienne Pot, Goker Erdogan, Yana Hasson, Yi Yang, Klaus Greff, Guillaume Le Moing, Sjoerd van Steenkiste, Daniel Zoran, Drew A. Hudson, Pedro Vélez, Luisa Polanía, Luke Friedman, Chris Duvarney, Ross Goroshin, Kelsey Allen, Jacob Walker, Rishabh Kabra, Eric Aboussouan, Jennifer Sun, Thomas Kipf, Carl Doersch, Viorica Pătrăucean, Dima Damen, Pauline Luc, Mehdi S. M. Sajjadi, and Andrew Zisserman. Scaling 4d representations, 2025. URL https://arxiv.org/abs/2412.15212.

Wei-Yi Chen, Yi-Ling Lin, Yu-An Su, Wei-Hsin Yeh, and Lun-Wei Ku. Yourskatingcoach: A figure skating video benchmark for fine-grained element analysis. *ArXiv*, abs/2410.20427, 2024. URL https://api.semanticscholar.org/CorpusID:273654721.

Willem Davison, Xinyue Hao, and Laura Sevilla-Lara. It's a matter of time: Three lessons on long-term motion for perception, 2026. URL https://arxiv.org/abs/2602.14705.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement, 2023. URL https://arxiv.org/abs/2306.08637.

Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

Hazel Doughty, Dima Damen, and Walterio Mayol-Cuevas. Who's better? who's best? pairwise deep ranking for skill determination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6057–6066, 2018.

Hazel Doughty, Ivan Laptev, Walterio Mayol-Cuevas, and Dima Damen. Action modifiers: Learning from adverbs in instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 868–878, 2020.

Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, and Andrew Zisserman. Ovr: A dataset for open vocabulary temporal repetition counting in videos, 2024. URL https://arxiv.org/abs/2407.17085.

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 *CVPR*, 2025.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Frund, Peter Yianilos, Moritz Mueller-Freitag, et al. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pp. 5842–5850, 2017.

Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, Eugene Byrne, Zach Chavis, Joya Chen, Feng Cheng, Fu-Jen Chu, Sean Crane, Avijit Dasgupta, Jing Dong, Maria Escobar, Cristhian Forigua, Abrham Gebreselasie, Sanjay Haresh, Jing Huang, Md Mohaiminul Islam, Suyog Jain, Rawal Khirodkar, Devansh Kukreja, Kevin J Liang, Jia-Wei Liu, Sagnik Majumder, Yongsen Mao, Miguel Martin, Effrosyni Mavroudi, Tushar Nagarajan, Francesco Ragusa, Santhosh Kumar Ramakrishnan, Luigi Seminara, Arjun Somayazulu, Yale Song, Shan Su, Zihui Xue, Edward Zhang, Jinxu Zhang, Angela Castillo, Changan Chen, Xinzhu Fu, Ryosuke Furuta, Cristina Gonzalez, Prince Gupta, Jiabo Hu, Yifei Huang, Yiming Huang, Weslie Khoo, Anush Kumar, Robert Kuo, Sach Lakhavani, Miao Liu, Mi Luo, Zhengyi Luo, Brighid Meredith, Austin Miller, Oluwatumininu Oguntola, Xiaqing Pan, Penny Peng, Shraman Pramanick, Merey Ramazanova, Fiona Ryan, Wei Shan, Kiran Somasundaram, Chenan Song, Audrey Southerland, Masatoshi Tateno, Huiyu Wang, Yuchen Wang, Takuma Yagi, Mingfei Yan, Xitong Yang, Zecheng Yu, Shengxin Cindy Zha, Chen Zhao, Ziwei Zhao, Zhifan Zhu, Jeff Zhuo, Pablo Arbelaez, Gedas Bertasius, David Crandall, Dima Damen, Jakob Engel, Giovanni Maria Farinella, Antonino Furnari, Bernard Ghanem, Judy Hoffman, C. V. Jawahar, Richard Newcombe, Hyun Soo Park, James M. Rehg, Yoichi Sato, Manolis Savva, Jianbo Shi, Mike Zheng Shou, and Michael Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives, 2024. URL https://arxiv.org/abs/2311.18259.

Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti, Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang,

Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator, 2022. URL https://arxiv.org/abs/2203.03570.

Chunhui Gu, Chen Sun, David A. Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, Cordelia Schmid, and Jitendra Malik. Ava: A video dataset of spatio-temporally localized atomic visual actions, 2018. URL https://arxiv.org/abs/1705.08421.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners, 2021. URL https://arxiv.org/abs/2111.06377.

Wenyi Hong, Yean Cheng, Zhuoyi Yang, Weihan Wang, Lefan Wang, Xiaotao Gu, Shiyu Huang, Yuxiao Dong, and Jie Tang. Motionbench: Benchmarking and improving fine-grained video motion understanding for vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8450–8460, June 2025a.

Wenyi Hong, Yean Cheng, Zhuoyi Yang, Weihan Wang, Lefan Wang, Xiaotao Gu, Shiyu Huang, Yuxiao Dong, and Jie Tang. Motionbench: Benchmarking and improving fine-grained video motion understanding for vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8450–8460, June 2025b.

Hsin-Ping Huang, Charles Herrmann, Junhwa Hur, Erika Lu, Kyle Sargent, Austin Stone, Ming-Hsuan Yang, and Deqing Sun. Self-supervised autoflow. In *CVPR*, 2023.

Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. In *Proc. arXiv:2410.11831*, 2024.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1725–1732, 2014.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Limin Wang, and Yu Qiao. Uniformerv2: Spatiotemporal learning by arming image vits with video uniformer, 2022.

Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 513–528, September 2018.

Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. Tempcompass: Do video llms really understand videos? *arXiv preprint arXiv: 2403.00476*, 2024.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips, 2019. URL https://arxiv.org/abs/1906.03327.

Davide Moltisanti, Frank Keller, Hakan Bilen, and Laura Sevilla-Lara. Learning action changes by measuring verb-adverb textual relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23110–23118, June 2023.

Yatian Pang, Wenxiao Wang, Francis E. H. Tay, Wei Liu, Yonghong Tian, and Li Yuan. Masked autoencoders for point cloud self-supervised learning, 2022. URL https://arxiv.org/abs/2203.06604.

Lyndsey Pickup, Zheng Pan, Donglai Wei, Yichang Shih, Andrew Zisserman, William T Freeman, and Bernhard Schölkopf. Seeing the arrow of time. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2035–2042, 2014.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Kanchana Ranasinghe, Muzammal Naseer, Salman Khan, Fahad Shahbaz Khan, and Michael S. Ryoo. Self-supervised video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2874–2884, June 2022.

Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video, 2017. URL https://arxiv.org/abs/1702.00824.

Adrià Recasens, Pauline Luc, Jean-Baptiste Alayrac, Luyu Wang, Florian Strub, Corentin Tallec, Mateusz Malinowski, Viorica Pătrăucean, Florent Altché, Michal Valko, Jean-Bastien Grill, Aäron van den Oord, and Andrew Zisserman. Broaden your views for self-supervised video learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1255–1265, October 2021.

Laura Sevilla-Lara, Shengxin Zha, Zhicheng Yan, Vedanuj Goswami, Matt Feiszli, and Lorenzo Torresani. Only time can tell: Discovering temporal data for temporal modeling. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 535–544, January 2021.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Ujjwal Upadhyay, Mukul Ranjan, Zhiqiang Shen, and Mohamed Elhoseiny. Time blindness: Why video-language models can't see what humans can?, 2025. URL https://arxiv.org/abs/2505.24867.

Renaud Vandeghen, Fida Mohammad Thoker, Marc Van Droogenbroeck, and Bernard Ghanem. Trackmae: Video representation learning via track mask and predict, 2026. URL https://arxiv.org/abs/2603.27268.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking, 2023. URL https://arxiv.org/abs/2303.16727.

Junbin Xiao, Xindi Shang, Angela Yao, and Tat-Seng Chua. Next-qa: Next phase of question-answering to explaining temporal actions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9777–9786, 2021.

Zihui Xue, Mi Luo, and Kristen Grauman. Seeing the arrow of time in large multimodal models, 2025. URL https://arxiv.org/abs/2506.03340.

Kexin Yi, Chuang Gan, Yunzhu Li, Pushmeet Kohli, Jiajun Wu, Joshua Tenenbaum, and Antonio Torralba. Clevrer: Collision events for video representation and reasoning. In *International Conference on Learning Representations (ICLR)*, 2020.

Rowan Zellers, Jiasen Lu, Ximing Lu, Youngjae Yu, Yanpeng Zhao, Mohammadreza Salehi, Aditya Kusupati, Jack Hessel, Ali Farhadi, and Yejin Choi. Merlot reserve: Multimodal neural script knowledge through vision and language and sound. In *CVPR*, 2022.

Shijie Zhou, Alexander Vilesov, Xuehai He, Ziyu Wan, Shuwang Zhang, Aditya Nagachandra, Di Chang, Dongdong Chen, Xin Eric Wang, and Achuta Kadambi. Vlm4d: Towards spatiotemporal awareness in vision language models. *arXiv preprint arXiv:2508.02095*, 2025.

Daniel Zoran, Nikhil Parthasarathy, Yi Yang, Drew A Hudson, Joao Carreira, and Andrew Zisserman. Recurrent video masked autoencoders, 2025. URL https://arxiv.org/abs/2512.13684.

6 APPENDIX

In this supplementary material, we include the following:

A Additional Experimental Results

A.1 Motion-only baseline comparison

A.2 Video Understanding Task Standalone Model Performance

A.3 Attentive Probing

B Additional Ablation Studies

B.1 Masking Strategy

B.2 Sim-to-Real Generalization

B.3 Static Camera Simulations

B.4 Token Components

C Method Design Choices

C.1 Zero-Shot Evaluation Setting

C.2 Late Fusion Choice

C.3 Loss Function Design

C.4 Tokenization Details

D TIME Model Training Details

D.1 Kubric pipeline settings

D.2 Kubric Occlusion Handling

D.3 Model training settings

E Baseline Model Details

E.1 VideoMAEv2 Pre-training Details

E.2 V-JEPA 2 Pre-training Details

E.3 RVM Pre-training Details

E.4 Linear Probe Baseline Feature Dimensions

F Evaluation Design Choices

F.1 SSv2 “Arrow of Time” class pairs

F.2 Ego-Exo4D Preprocessing

G TIME Model Efficiency Claims

G.1 Training Efficiency

G.2 Addressing Point Tracking Penalty

H Limitations

I AI use statement

J Ethics statement

K Reproducibility statement

L Synthetic Data Visualizations

A Additional Experimental Results

A.1 Motion-only baseline comparison

As a sanity check, we include a comparison via a linear probe between our TIME model representations and the corresponding backbone features of CoTracker3 (Karaev et al., 2024), TAPIR (Doersch et al., 2023), and features obtained from raw trajectories. Our model maintains the lead in all benchmarks. Results can be seen in Table 3.

Table 3: **Linear probe accuracy of trajectory-based methods on SSv2, Ego-Exo4D, and Diving48.** Each cell reports top-1 accuracy with standard deviation.

Model	SSv2 Top-1 (± Std %)	Ego-Exo4D Top-1 (± Std %)	Diving48 Top-1 (± Std %)
TIME (Ours)	**19.42** ± 0.04	**17.27** ± 0.12	**8.37** ± 0.15
CoTracker3 backbone (Karaev et al., 2024)	3.88 ± 0.02	10.42 ± 0.12	<u>6.48</u> ± 0.10
TAPIR backbone (Doersch et al., 2023)	4.77 ± 0.02	<u>12.33</u> ± 0.22	5.32 ± 0.23
Features from raw trajectories	<u>8.63</u> ± 0.02	2.87 ± 0.58	4.76 ± 0.28

A.2 Video Understanding Task Standalone Model Performance

Full SSv2, Diving48 and Ego-Exo4D linear probing results including standalone model performance can be found in Table 4.

Table 4: **Video Understanding Task Performance.** We report top-1 accuracy. Here, Δ denotes the improvement over the baseline.

	Ego-Exo4D		SSv2		Diving48	
Model	**Top-1 (± Std %)**	**Δ**	**Top-1 (± Std %)**	**Δ**	**Top-1 (± Std %)**	**Δ**
Image Foundation Models						
CLIP (4f) (Radford et al., 2021)	25.11 ± 0.19	–	18.56 ± 0.04	–	13.97 ± 0.25	–
DINOv3 (4f) (Caron et al., 2021)	24.88 ± 0.13	–	21.95 ± 0.06	–	16.35 ± 0.19	–
Point Trajectory Encoder						
TIME (Point Tracks)	17.27 ± 0.12	–	19.42 ± 0.04	–	8.37 ± 0.15	–
Video Foundation Models						
RVM† (Zoran et al., 2025)	25.67 ± 0.18	–	40.08 ± 0.07	–	9.21 ± 0.16	–
VideoMAEv2† (Wang et al., 2023)	41.01 ± 0.15	–	33.05 ± 0.07	–	24.24 ± 0.42	–
V-JEPA 2† (Assran et al., 2025)	34.85 ± 0.24	–	49.78 ± 0.05	–	30.08 ± 0.30	–
Ours + Spatial / Spatiotemporal						
TIME + CLIP (4f)	39.33 ± 0.24	**+14.22**	31.56 ± 0.05	**+13.00**	16.11 ± 0.28	**+2.14**
TIME + DINOv3 (4f)	39.63 ± 0.22	**+14.75**	34.60 ± 0.10	**+12.65**	18.08 ± 0.21	**+1.73**
TIME + RVM†	37.00 ± 0.17	**+11.32**	41.92 ± 0.06	**+1.84**	11.74 ± 0.11	**+2.53**
TIME + VideoMAEv2†	51.14 ± 0.23	**+10.12**	38.74 ± 0.09	**+5.69**	24.97 ± 0.24	**+0.73**
TIME + V-JEPA 2†	45.60 ± 0.12	**+10.75**	50.00 ± 0.10	**+0.22**	30.04 ± 0.15	**−0.04**

† Denotes models whose pre-training data explicitly includes the SSv2 dataset.

A.3 Attentive Probing

We experiment with the attentive probing protocol (Bardes et al., 2024). Due to hardware resource constraints (e.g., storing raw unpooled video model encoder features required for training the attentive probe needs terabytes of storage), we follow the low-shot frozen feature evaluation protocol from the V-JEPA v1 paper (Bardes et al., 2024). We extract and store raw features from each model (TIME, V-JEPA 2, RVM, VideoMAEv2) from 5% of the training data ($\approx$ 48 samples per class). We then follow the attentive probe protocol from the publicly available code (Carreira et al., 2025) and train the attentive probes on the extracted frozen training data features. We evaluate on the frozen validation set features. We repeat the experiment 3 times and report the mean accuracy and the standard deviation. We confirm the correctness of the attentive probing protocol based on

VideoMAEv2 performance in the original V-JEPA v1 paper (Bardes et al., 2024). In this work, the reported accuracy of VideoMAEv2 5% low-shot attentive probe (28.0 ± 1.0) aligns with that of our protocol implementation (28.22 ± 0.96) in Table 5. We conclude that, under the attentive probe protocol, TIME benefits all models.

Table 5: **Low-Shot (5%) Attentive Probe Evaluation on SSv2.** For concatenated models, Δ indicates the improvement over the corresponding standalone model.

Model	SSv2 Acc. (%)	Δ
TIME (Ours)	12.92 ± 1.02	–
VideoMAEv2 (Wang et al., 2023)	28.22 ± 0.96	–
TIME + VideoMAEv2	29.46 ± 0.83	**+1.24**
RVM (Zoran et al., 2025)	39.83 ± 0.71	–
TIME + RVM	42.16 ± 0.78	**+2.33**
V-JEPA 2 (Assran et al., 2025)	48.96 ± 0.43	–
TIME + V-JEPA 2	50.21 ± 0.41	**+1.25**

B Additional Ablation Studies

B.1 Masking Strategy

We explore how TIME deals with point trajectories that appear later, by performing a middle frame tracking experiment. We initialize the point trajectory grid from the middle frame and obtain trajectory paths in both directions. This enables us to place point trajectories on all objects that appear in the middle of the video rather than at the beginning. We switch to middle-frame trajectories directly at inference (i.e., using the original TIME model trained on first-frame trajectories). We train a linear probe on the full SSv2 dataset and report mean and standard deviation after 5 runs in Table 6. As seen in the results, tracking from the middle frame leads to an additional 1% gain in TIME linear probe performance. This suggests that the model performance may increase during the point grid initialization at a timestep when objects of interest are visible in the scene. However, while in SSv2 we do observe that objects of interest often appear after the first frame, this behavior is dataset-dependent. Therefore, determining the optimal time to initialize general point tracking is beyond the scope of this project, hence, we leave it for future work. Additionally, results suggest that the proposed model is invariant to the grid initialization time.

Table 6: **Comparison of TIME using first-frame versus middle-frame tracks on Something-Something v2.**

Baseline	Acc. (%)
TIME (First frame tracks)	19.42 ± 0.04
TIME (Middle frame tracks)	20.46 ± 0.12

B.2 Sim-to-Real Generalization

We further evaluate synthetic versus real-world trajectory impact on model pre-training. We train two TIME models from scratch. The first model is trained on a subset (50k) of Kubric simulations. The second model is trained on a subset (50k) of SSv2 trajectories obtained using CoTracker3. We evaluate these models on SSv2 validation set via a standard linear probe. Each experiment is repeated 5 times and the results are shown in the Table 7. We observe that training on real-world trajectories leads to a minor increase in performance (1.68% relative gain) on the SSv2 validation set. Similarly to our ablation study in Figure 5, the results suggest that synthetic trajectories already generalize well to real-world scenarios.

Table 7: **Comparison of Kubric baseline and real-world trajectories on SSv2.**

Setting	Acc. (%)	Δ (%)
Baseline (Kubric)	14.88 ± 0.08	–
Real-world traj.	15.13 ± 0.03	+1.68%

Table 8: **Accuracy (%) on Something-Something v2 for selected camera motion labels.** Comparison between TIME and VideoMAEv2.

	Acc. (%)	
Something-Something v2 Action Label	**TIME**	**VideoMAEv2**
Turning the camera right while filming something	96.91	88.27
Turning the camera left while filming something	93.57	91.43
Moving away from something with your camera	84.09	69.32
Approaching something with your camera	83.62	78.45
Turning the camera upwards while filming something	66.67	76.67
Turning the camera downwards while filming something	62.90	67.74
Moving something towards the camera	47.30	45.95
Moving something away from the camera	44.64	32.14

B.3 Static Camera Simulations

The synthetic training data pipeline produces samples with a static camera. During evaluation, however, TIME does not assume a static camera as it is tested on datasets that include diverse camera motion, such as the full Something-Something v2 (SSV2) (Goyal et al., 2017) dataset and the full Diving48 (Li et al., 2018) dataset. We conduct experiments to show that the proposed model is able to generalize well: for example some of the classes in the SSV2 datasets are types of camera motion, and the proposed method does particularly well in those. The performance of standalone TIME on action labels from SSV2 that are specific to camera motion can be seen in Table 8.

B.4 Token Components

We complete a leave-one-out ablation study by excluding KNN (max-deviation), global displacement and occlusion signals.The TIME models are trained from scratch on a subset (50k) of Kubric simulations and evaluated on SSv2 via a linear probe. Each experiment is repeated 5 times and the results are shown in Table 9. We report that each ablated component significantly impacts the encoder downstream task performance. Occlusion (-23.66% drop) is the most impactful followed by neighborhood (-20.90% drop) and global displacement (-11.29% drop) signals.

Table 9: **Leave-one-out ablation study on TIME token components on Something-Something v2.**

Setting	Acc. (%)	Δ (%)
Baseline (50k Kub.)	14.88 ± 0.08	–
w/o KNN	11.77 ± 0.10	-20.90%
w/o Global Disp.	13.20 ± 0.16	-11.29%
w/o Occlusion	11.36 ± 0.09	-23.66%

C METHOD DESIGN CHOICES

C.1 ZERO-SHOT EVALUATION SETTING

Our reported evaluation is on zero-shot model performance from frozen features. Therefore, the models themselves are not pretrained on the evaluation datasets unless specified. As mentioned in the main paper, we follow the usual linear probe evaluation protocol from frozen features. First, we obtain training data and validation data features using a forward pass for each frozen model separately. Next, frozen training data features are then used to train linear probes. Finally, the corresponding frozen validation data features are passed to the trained linear probes to obtain the performance metrics. As a result, the evaluation on all models is zero-shot as only the linear probe is trained and evaluated on frozen model features.

C.2 LATE FUSION CHOICE

We concatenate frozen features of TIME and standard video and image representations, and use a linear classifier on the full benchmarks. The goal of this work is to explore a novel temporal paradigm "without bells and whistles", not to achieve the best possible results on the benchmarks. Hence the choice of fusion is indeed deliberately simple. Therefore, we leave the integration with images, languages, etc. for future works. A parallel work (Vandeghen et al., 2026), attempts to fuse trajectories with RGB video during training. In contrast, our work specifically chooses to model motion independently of appearance. This enables us to test whether removing high-dimensionality associated with visual cues would help to learn temporal motion manifold more efficiently. As seen in Appendix A.2, Table 4, late fusion of TIME features with sparse spatial features from DINOv3 (4 video frames) allows us to exceed baseline VideoMAEv2 performance (34.60 ± 0.10 versus 33.05 ± 0.07 respectively). This suggests that late fusion of spatial and temporal features may be a possible alternative to resource-intensive modelling of spatio-temporal representations directly.

C.3 LOSS FUNCTION DESIGN

The original MAE reconstruction methodology considers all masked patches equally (Tong et al., 2022). As a result, we follow this methodology and treat each trajectory point (occluded or not) equally in the reconstruction objective as well. In addition, as we train purely on synthetic data, all point trajectories are ground-truth. This reliability component further forces the model to learn semantically rich events without differentiating between occlusion flags. Furthermore, deliberately excluding occlusion in the reconstruction objective lessens the effect of a distribution shift between occluded ground-truth points during training and noisy points during real-world inference tasks. Similarly, we keep the reconstruction loss only on local and global displacement tokens to maintain simplicity. We empirically find that this is sufficient to prevent mode-collapse during training, thus, we omit any additional token information during this step.

C.4 TOKENIZATION DETAILS

In the proposed TIME model, corresponding tokens are organized along temporal and spatial axes:

Temporal axis. Following the original VideoMAEv2 paper (Wang et al., 2023), we set the tubelet size to 2 consecutive frames. Thus, we group each of the 24 input frames and partition them into 12 non-overlapping tubelets maintaining the original temporal order.

Spatial axis. Moving point trajectories are likely to diverge from their initialized 2D grid location over time. Therefore, we treat them as a flat, orderless set of 1024 points. Next, to model point location along the spatial axis, we assign each point a single learnable embedding vector. We repeat this across all 12 tubelets per each point. This mechanism enables the model to learn the spatial location of each point during training.

D TIME MODEL TRAINING DETAILS

D.1 KUBRIC PIPELINE SETTINGS

We provide additional Kubric MOVi-B pipeline (Greff et al., 2022) settings for experiment reproducability purposes in the Table 10 below.

Table 10: **Kubric dataset pipeline settings.**

Parameter	**Distribution**
Background	Solid floor
# Objects	Uniform selection between 3 and 10 objects
Shapes	Uniform selection from 11 KuBasic assets: cube, cylinder, sphere, cone, torus, gear, torus knot, sponge, spot, teapot, suzanne
Scale	Uniform number between 0.7 and 1.4 (inclusive)
Material	Uniform choice between metal or rubber properties
Spawn region	Between regions $x, y \in [-5, 5]$ and $z \in [1, 5]$ without object overlap
Initial velocity	Uniform choice of values between $[-4, 4]$ for x and y axes. Biased toward origin
Physics	PyBullet simulation for 24 frames at 12 fps (2 seconds)

D.2 KUBRIC OCCLUSION HANDLING

Synthetic trajectories generated by Kubric are marked as occluded if they are covered by another object or if they fall outside the image frame margins. The "noise from occlusions" line refers to the noisy real-world tracks obtained using point tracking. Using synthetic data, we can guarantee that occluded points are as accurate as non-occluded, thus preventing the proposed model from learning from incorrect point trajectories during training.

D.3 MODEL TRAINING SETTINGS

In this section we provide more extensive details of our model training. We report the numbers that were used to train the TIME model on $250,000$ synthetic Kubric MOVi-B samples. More information regarding hyperparameters can be found in Table 11.

To repeat all the evaluation experiments (including feature extraction from pre-trained video model baselines and use of tracking software) in this paper, we estimate it would take $<$100 GPU hours.

E BASELINE MODEL DETAILS

E.1 VIDEOMAEV2 PRE-TRAINING DETAILS

We use the official VideoMAEv2-Base model checkpoint pre-trained for 800 epochs in a self-supervised way on UnlabeldHybrid-1M dataset. The pre-training dataset details can be seen in Table 12.

E.2 V-JEPA 2 PRE-TRAINING DETAILS

We use the official pre-trained V-JEPA 2 ViT-L model checkpoint. This is the smallest publicly available V-JEPA 2 checkpoint. As a result, the discrepancy between model sizes (i.e., 80M parameters for other baselines vs 300M for V-JEPA 2) may provide a competitive performance advantage to the V-JEPA 2 model during evaluation. The pre-training dataset details can be seen in Table 13.

Table 11: **TIME Pre-training Hyperparameters.** Our 250K model is trained for 100 epochs. Training on 4×A100 80GB GPUs lasts ≈ 5 days (≈700 GPU hours).

Parameter	**Value**
Number of frames	24
Frames per Second	12.0
Crop Size	256
Sampling Rate	1
Grid Size (track points)	32×32
Number of Track Points	1024
Tubelet Size	2
Token Count per Sample	12,288
Mask Ratio	0.75
Visible Tokens	∼1,230 (25%)
Encoder Depth	12
Decoder Depth	4
k-Nearest Neighbors	16
Loss Type	Huber ($\beta = 0.5$)
Motion Loss Boost	7.0
Epochs	100
Warmup Epochs	3
Batch Size (per GPU)	10
Batch Size (global, 4×GPUs)	40
Starting Learning Rate	1.5×10^{-4}
Final Learning Rate	1.0×10^{-6}
Weight Decay	0.05
Drop Path	0.1
Clip Gradient	0.02
Optimizer	AdamW
Precision	AMP (Automatic Mixed Precision)
Data Augmentation	None
Coordinate Range	$[-256, 512]$
Visibility Mask	Enabled

Table 12: **VideoMAEv2 Pre-training Dataset Overview.**

Dataset	Size	Source
K710 (Li et al., 2022)	658k	YouTube
SSv2 (Goyal et al., 2017)	169k	Shot from Scripts
AVA (Gu et al., 2018)	21k	Movie
WebVid2M (Bain et al., 2021)	250k	Internet
self-collected	250k	Instagram
UnlabeledHybrid	1.348M	Multi-Source

Table 13: **V-JEPA 2 VideoMix22M (VM22M) Pretraining Dataset.**

Source	Samples	Type	Total Hours	Apply Curation	Weight
SSv2 (Goyal et al., 2017)	168K	EgoVideo	168	No	0.056
Kinetics (Carreira et al., 2022)	733K	ExoVideo	614	No	0.188
Howto100M (Miech et al., 2019)	1.1M	ExoVideo	134K	No	0.318
YT-Temporal-1B (Zellers et al., 2022)	19M	ExoVideo	1.6M	Yes	0.188
ImageNet (Deng et al., 2009)	1M	Images	n/a	No	0.250

E.3 RVM Pre-training Details

We use the official RVM ViT-B pre-trained checkpoint from Google DeepMind public repository. The pre-training dataset details can be seen in Table 14.

Table 14: **RVM Pretraining Dataset Composition.**

Source	Samples	Type	FPS	Apply Curation	Weight
SSv2 (Goyal et al., 2017)	168K	EgoVideo	25	No	0.056
Kinetics-700 (Carreira et al., 2022)	733K	ExoVideo	25	No	0.188
Howto100M (Miech et al., 2019)	1.1M	ExoVideo	10	No	0.318
YT8M (Abu-El-Haija et al., 2016)	3.3M	ExoVideo	10	No	0.188
YT-BoundingBoxes (Real et al., 2017)	380K	ExoVideo	10	No	0.250

E.4 Linear Probe Baseline Feature Dimensions

To ensure a fair comparison between model features, we maintain the standard linear probing protocol in self-supervised learning - mean pooling over the token sequence. We report the final resulting feature vector dimensions in Table 15.

While all the video models (including TIME) used in our evaluation are base transformer size-equivalent (ViT-B), the smallest publicly available V-JEPA 2 checkpoint is ViT-L. This discrepancy between model sizes (i.e., 80M parameters for other baselines vs 300M for V-JEPA 2) may provide a competitive performance advantage to the V-JEPA 2 model during evaluation.

F Evaluation Design Choices

F.1 SSv2 "Arrow of Time" class pairs

We provide the action pair details for the 28 classes of the Something-Something v2 (Goyal et al., 2017) "Arrow of Time" subset in Table 16.

F.2 Ego-Exo4D Preprocessing

For Ego-Exo4D (Grauman et al., 2024), we use all available video takes containing *keystep annotations* (i.e., "Bike Repair", "Cooking", and "Health" scenes), resulting in a set containing 24,060

Table 15: **Feature readout protocol.** All models are compared using standard linear probing with mean pooling over the token sequence.

Model	Architecture	Mean-pooled Feature Vector Dimension
TIME	ViT-B	768
VideoMAEv2	ViT-B	768
RVM	ViT-B	768
V-JEPA 2	ViT-L	1024

Table 16: **SSv2 "Arrow of Time" Forward and Backward Action Pairs.**

Forward Action	Backward Action
Moving something up	Moving something down
Pushing from left to right	Pushing from right to left
Pulling from left to right	Pulling from right to left
Moving towards the camera	Moving away from the camera
Moving things closer	Moving things away
Moving closer to something	Moving away from something
Lifting up completely	Putting down completely
Lifting up one end	Putting down one end
Rolling down a slant	Rolling up a slant
Opening something	Closing something
Folding something	Unfolding something
Covering something	Uncovering something
Putting into something	Taking out of something
Plugging into something	Pulling out of something

training and 6,014 validation samples across 620 classes. We extract all action snippets from these Ego-Exo4D scenes containing the 'best Exo' camera view as labeled in the metadata. We include all actions between 0.5s and 5.0s in length. If an action snippet is longer than 5.0s, we center crop these action snippets temporally to capture the most representative portion of the video.

G TIME Model Efficiency Claims

G.1 Training Efficiency

We further explain the model efficiency by comparing to the gains of other video models when trained with more data. For example, as reported by the authors, V-JEPA trained on 2M videos vs trained 22M videos gains 0.7% on average for temporal datasets (SSV2, Jester, Diving48) (Assran et al., 2025). In other words, it takes 20M videos to increase 0.7%. On the datasets that we experimented with TIME (SSV2, Diving48, EgoExo4D – note we exclude Jester as accuracies around 97% are practically saturated), we are able to improve V-JEPA's performance by up to 4.15% using 250K synthetic videos. This points to the fact that TIME is able to extract information in an alternative and complementary manner, which is more efficient than increasing the amount of training data of a specific model.

G.2 Addressing Point Tracking Penalty

Use of point tracking models (i.e., CoTracker3 (Karaev et al., 2024)) involves additional training data used to train the models themselves. However, when put in context numerically, even this additional training is remarkably small compared to existing foundation models. TIME is trained on 250K videos, of 2 seconds each, and CoTracker3 teacher models are primarily trained on 10K videos of 5 seconds each. This is a total of 0.017 years of video, compared to V-JEPA 2, trained on over 100 years of video.

H LIMITATIONS

In this paper we have shown the surprising power of training on motion information to learn video representations efficiently. We consider this work a stepping stone for several lines of future work that include the following. First, this model focuses on temporal information alone, and the integrations we have done with other appearance-based models are straightforward (e.g., in the form of concatenation). Fully integrating the TIME representation into an appearance-based model, while maintaining the temporal separation that has proved so advantageous will be a promising area of research. Second, this model at inference time relies on point-tracking techniques. While recent research (Karaev et al., 2024) has made them faster and more accurate, they might still introduce errors in fast-moving or blurry regions. Integrating this embedding with a point-tracking system or using other motion representations will also be a very promising direction.

I AI USE STATEMENT

In this work, we used generative AI tools for polishing writing, retrieval and discovery (e.g., finding related work), research ideation (e.g., aggregating notes and identifying overlapping challenges from related works), execution (e.g., using AI-powered coding assistant to generate scripts for training models on a compute cluster), generating synthetic datasets (e.g., dataset generation pipelines built with the help of AI-powered coding assistant). We have not used generative AI tools for drafting sections of the paper or proving mathematical claims. We have reviewed all AI-assisted work. We checked LLM-generated research ideas for potential plagiarism through a manual literature survey. LLM-generated code was verified and tested for correctness by evaluating it against the expected outputs. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

J ETHICS STATEMENT

Our proposed method focuses on making the self-supervised methods for learning video representations more dependable and efficient. While there are no direct negative societal impacts arising from this work, the proposed model is still subject to the inherited biases from the synthetic training data. Furthermore, the use of synthetic data for training video foundation models and the sim-to-real transfer are very active areas of research. Therefore, investigating synthetic data biases, assessing model fairness, and knowledge transfer capabilities during real-world deployment (e.g., robotics, autonomous driving) are left for future works.

K REPRODUCIBILITY STATEMENT

We provide training and test details in the main paper. In addition, further information about TIME model training, hyperparameters, datasets and baseline pre-training is available in the appendix. We fully disclose the information needed to reproduce the main experimental results. Our submission includes code and instructions for generating data, training models, preprocessing datasets, extracting features, and running evaluation benchmarks. The computational cost and resources necessary for running the experiments is provided in the "Training Details" section of this paper as well as the appendix. We provide the details of all datasets used in evaluation (Something-Something V2, CLEVRER, Diving48, Ego-Exo4D) together with model architectures included in the experiments (VideoMAEv2, V-JEPA 2, RVM, DINOv3, CLIP).

L SYNTHETIC DATA VISUALIZATIONS

During the experiments we observe that point tracks can capture events such as object collisions, interactions, velocity, scale changes, rotation and acceleration. We include a more extensive visualization of our model trajectory reconstruction progression in Figure 6. We also provide similar visualizations and briefly describe observed motion principles where objects are also segmented by colors (Figures 7, 8, and 9).

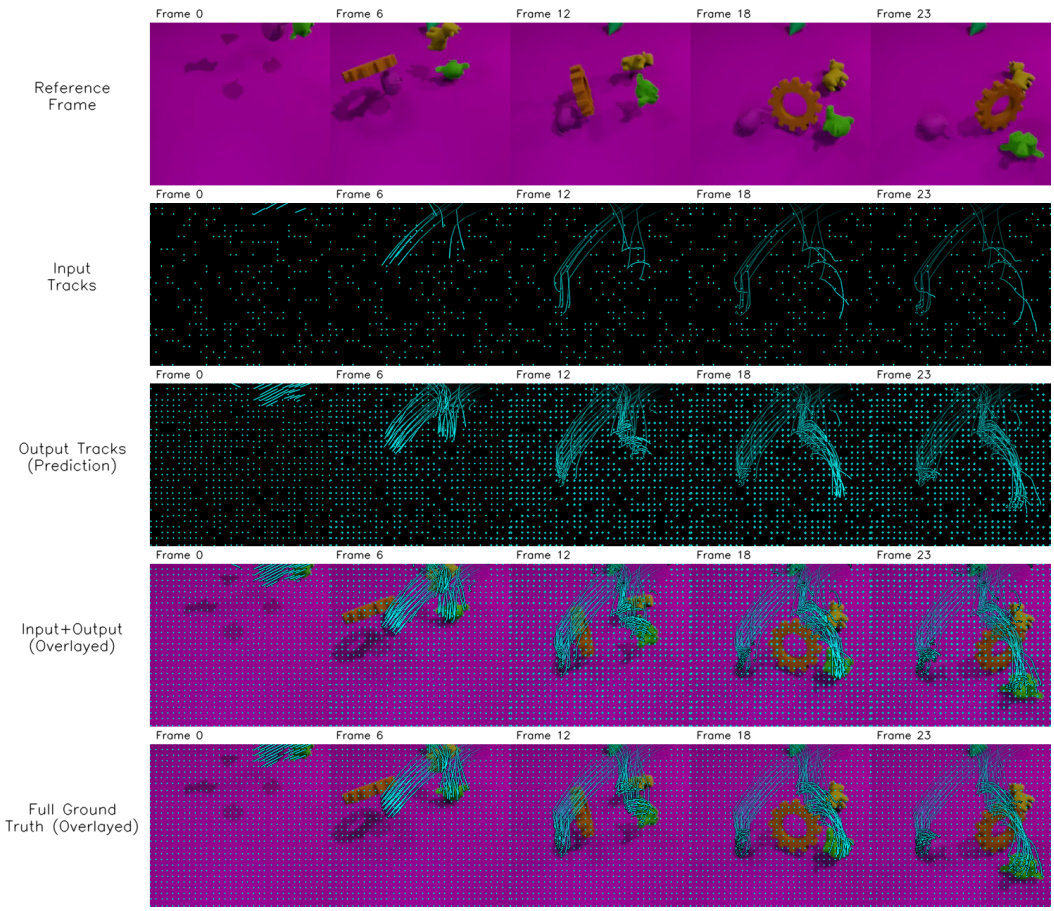

Figure 6: TIME visualization. Our model reconstructs missing trajectories which closely align with the ground truth.

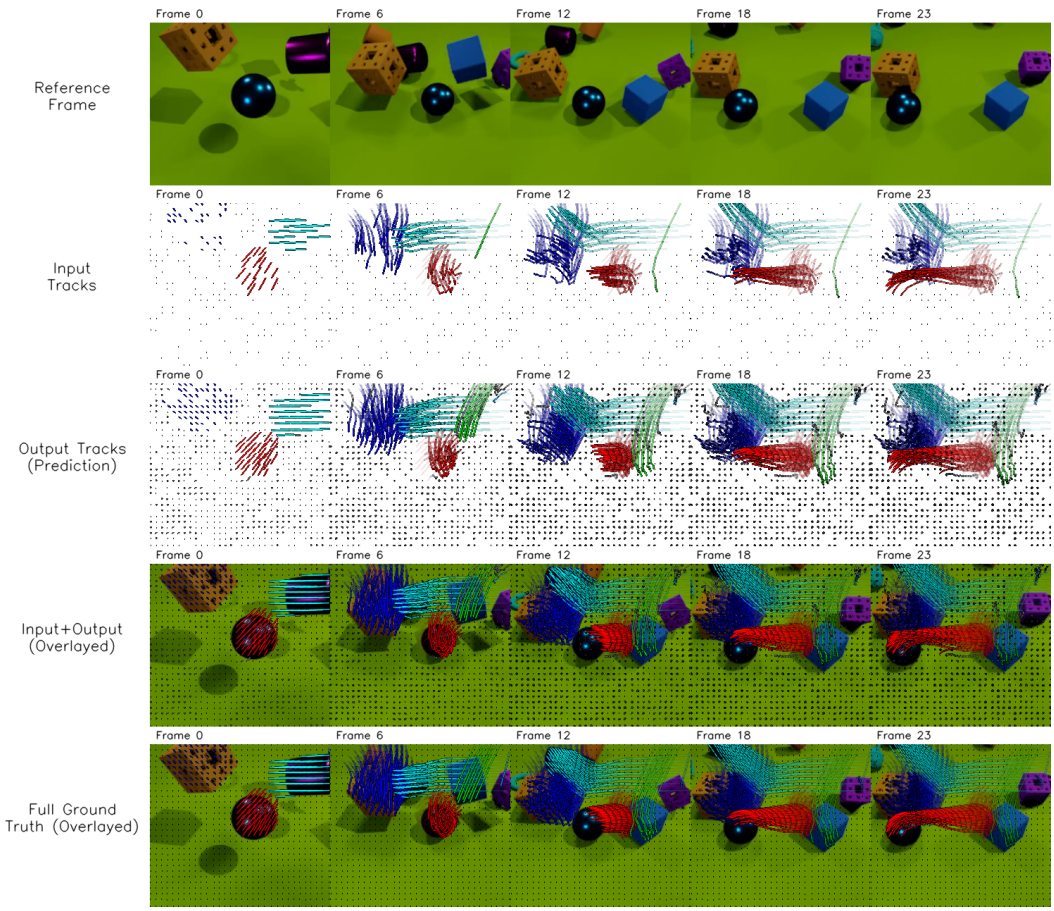

Figure 7: TIME visualization (objects segmented by colors). In this visualization we can observe that the proposed model successfully captures multi-object collisions and velocity changes. Static point tracks that fall onto the background remain static throughout the timeline.

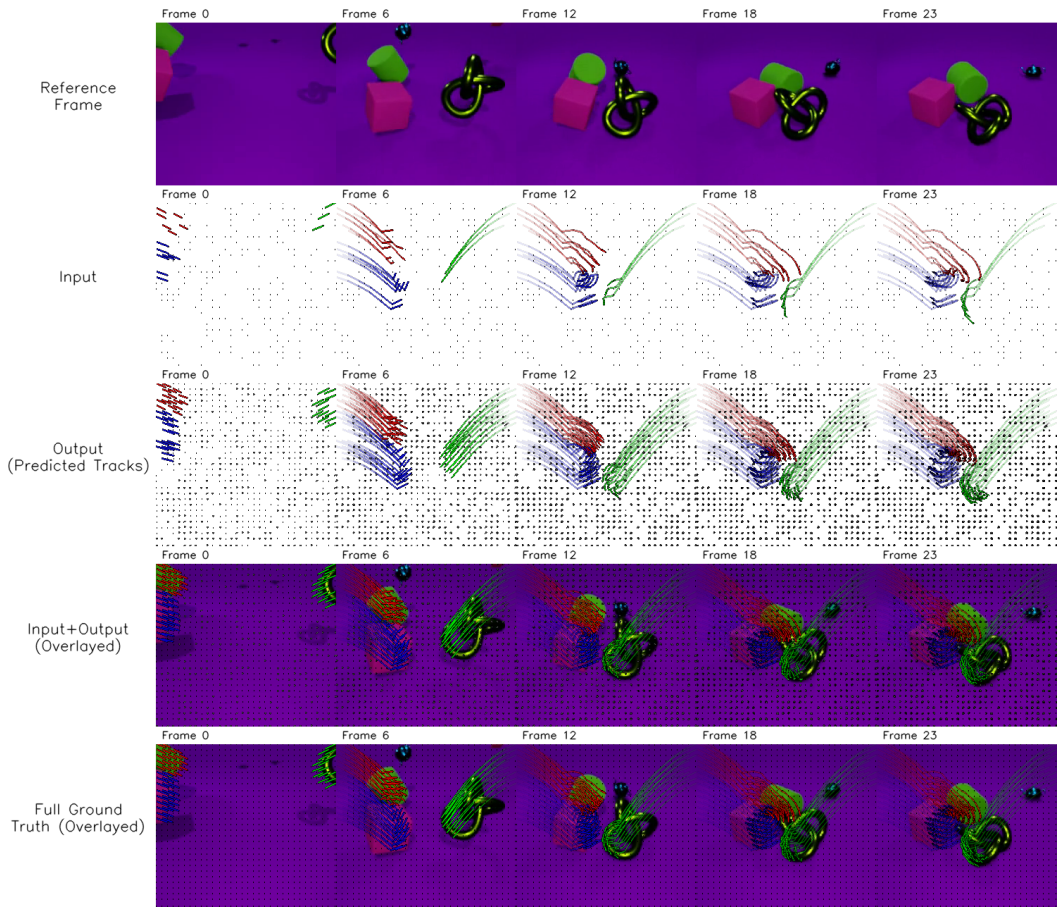

Figure 8: TIME visualization (objects segmented by colors). In this visualization we can observe that the proposed model successfully captures angular rotation of a complex object (e.g., torus knot).

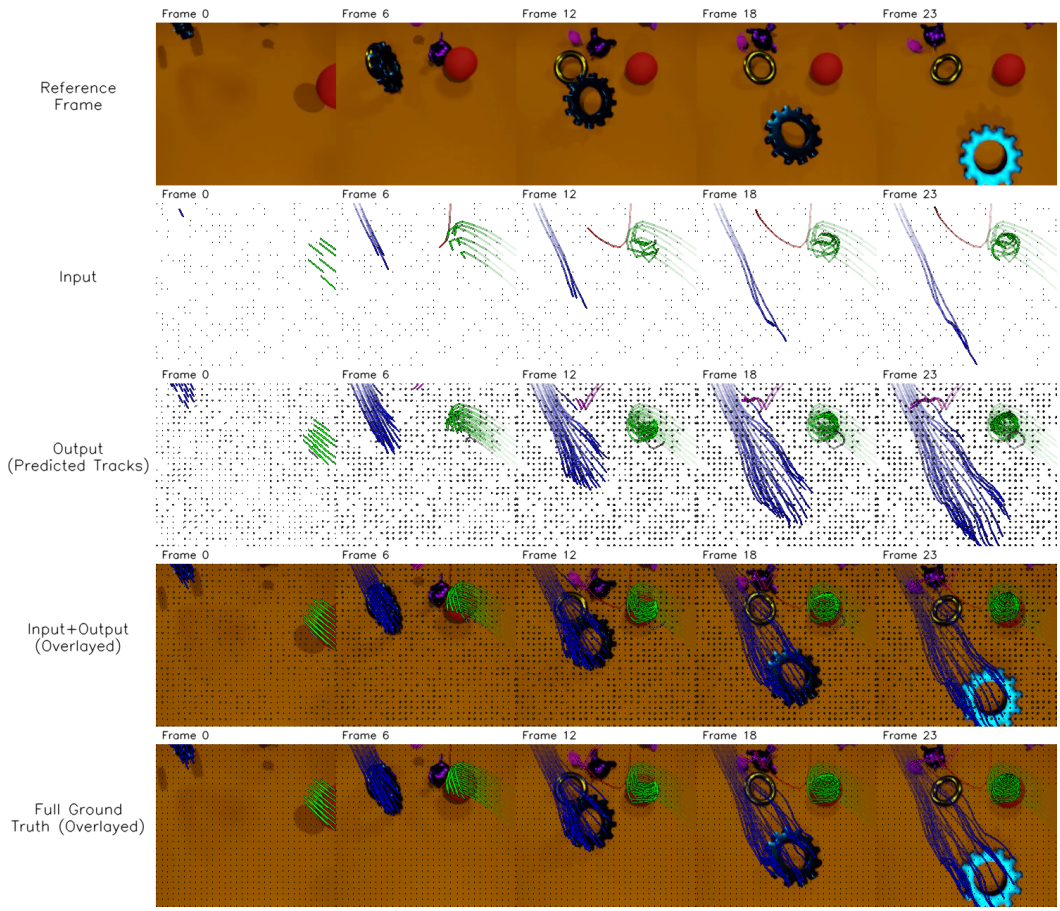

Figure 9: TIME visualization (objects segmented by colors). The proposed model successfully reconstructs object scale, rotation and velocity changes.