End-to-End Driving through Generative Video Pretraining

Florent Bartoccioni◇ Elias Ramzi◇ Victor Besnier Shashanka Venkataramanan Tuan-Hung Vu Yihong Xu Loick Chambon[1]
Spyros Gidaris Serkan Odabas David Hurych Renaud Marlet[2] Alexandre Boulch Mickael Chen⋆ Éloi Zablocki
Andrei Bursuc Eduardo Valle Matthieu Cord[1]

valeo.ai, Paris, France [1] Sorbonne Université, Paris, France [2] ENPC, Paris, France

1. Introduction

We introduce VaVAM, an end-to-end driving Video-Action Model that harnesses the power of generative video pretraining, learning unsupervised rich video representations from large driving video datasets and applies these representations for control via imitation learning.

Unsupervised end-to-end learning of driving policies is key to achieving scalable and adaptable autonomous systems without relying on costly and labor-intensive labeling. Current approaches lean either on reinforcement learning [9, 27, 40] in simulated environments such as CARLA [15], or on imitation learning from real-world expert driving data [5, 11, 20, 22, 33]. While the latter has shown promising results [22, 24], it is limited by the size and diversity of available datasets, which often fail to cover rare but critical edge cases. Some methods mitigate this by leveraging structured annotations (e.g., bounding boxes, HD maps), injecting privileged information to simplify training. However, this requires external labels that we aim to avoid.

Meanwhile, generative video models made remarkable progress [4, 8, 41], with recent models generating photorealistic, coherent videos by exploiting large amounts of raw video data. Specialized models such as GAIA-1 [21] and VISTA [17] excel at predicting future frames in driving videos, suggesting that generative models can learn meaningful representations of the physical world, possibly capturing semantics, dynamics, and geometry relevant to driving. However, it remains unclear whether these representations are useful for real-world driving. Can they support downstream tasks such as motion planning? Can they generalize beyond video prediction to actual control?

VaVAM allows us to investigate those questions. Our experiments yield three key findings: first, a pretrained video model captures meaningful driving semantics and dynamics, even without external labels; second, when used as input to the action expert, these representations significantly boost performance in closed-loop (online) driving tasks; finally, strong generalization and decision-making capabilities are within reach of this strategy, shown by VaVAM achieving state-of-the-art performance in safety-critical scenarios on the NeuroNCAP benchmark [28].

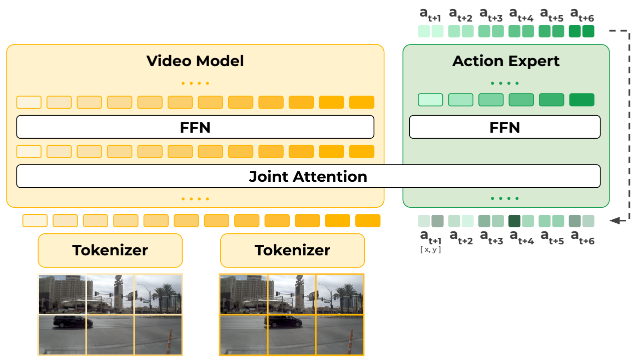


Figure 1. **End-to-end pipeline of VaVAM.** From a context of up to 8 frames, the video model (in yellow) builds a spatio-temporal representation, and then VaVAM's action expert (in green) estimates the dynamic profile of the driving actions to undertake.

Our main contributions are:

- We present training protocols for autoregressive video models, including details of our training data mix on 1,800+ hours of public, real-world driving footage.
- We describe how to adapt the pretrained video model into a full video-to-action pipeline by imitation learning, demonstrating the effectiveness of end-to-end driving from cameras alone.
- VaVAM sets the new state of the art in frontal driving scenarios on NeuroNCAP [28], a closed-loop evaluation setting.
- We will release our training protocols, our code, and trained checkpoints, with open-source and open-weight licenses for the benefit of the research community.

2. VaVAM: architecture and learning objectives

VaVAM is an end-to-end video-to-action model for autonomous driving. It processes raw video inputs to predict future driving trajectories conditioned on high-level commands (e.g., 'turn left', 'go straight'). As shown in Fig. 1, it combines a spatio-temporal video encoder, trained for video modeling, with an action generation module to form a unified perception-to-action pipeline.

Video Model: VaVAM's features. Our goal is to develop a video model that captures the semantic and dynamic structure of driving scenes from raw video inputs, ultimately as a foundation for autonomous driving. We first transform each video frame with a vector-quantized visual tokenizer (LlamaGen-VQGAN [16, 37]), which reduces the task to the sequential prediction of discrete tokens thus easing the capture of long-range spatiotemporal dependencies.

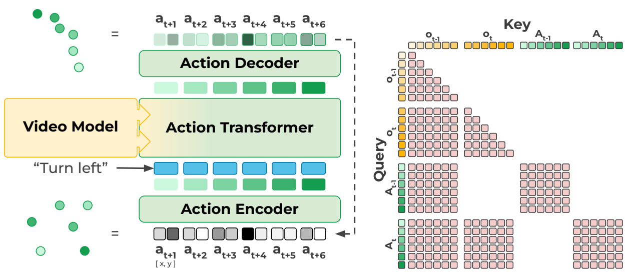
VaVAM central component is an autoregressor for tokens, factorizing the joint distribution of a spatio-temporal token sequence into a product of conditional probabilities. This formulation forces the model to understand both the local appearance and the temporal progression of scenes, capturing how objects move and interact over time. The design intent is that, by mastering this prediction task, the model should internalize key patterns of driving scenes that are crucial for downstream control and planning.

In practice, we use a GPT-2-style [34] architecture with causal attention, disentangled spatial and temporal positional embeddings [21], and tied input-output embeddings. It is trained with a next-token prediction loss, i.e., cross entropy and teacher forcing. Formal details appear in App. C.

VaVAM's action expert. Whether video generation pre-training effectively captures the features essential for safe and reliable driving is a key question. To bridge the gap between pre-trained video representations and driving decisions, we introduce an action expert module, forming VaVAM by complementing the video model with decision-making. The action expert module translates the visual representation into a future trajectory. Drawing inspiration from π_0 [3], we formulate the action prediction as a denoising process: given a noised trajectory, a video context, and a high-level goal, the model learns to iteratively refine it by flow matching (Fig. 2a).

The action expert comprises three components. First, an Action Encoder that projects trajectory tokens into a latent space, incorporating noise-level embeddings (τ) as well as temporal and command embeddings. Second, a Joint Attention Transformer that enables action tokens to attend to video features using a specialized attention mask (Fig. 2b): action tokens attend to video tokens of all current and past frames and to all action tokens within the same frame; video tokens follow strictly sequential causal attention among themselves and cannot attend to action tokens. Third, and finally, an Action Decoder that applies a linear map to the latent representation to predict the denoising flow.

Joint attention between video and action tokens is a key design choice allowing MLPs in the action branch smaller than the dimension of the video model, learnable probing of features from all the video model's layers, and KV caching of the video model's features while the action expert does several forward during denoising. While the original $\pi0$ [3] conditions on single frames for robotic manipulation, we extend it to driving by exploiting the temporal contexts of multiple frames that are crucial for understanding dynamic scenarios.



a_{t+1} a_{t+2} a_{t+3} a_{t+4} a_{t+5} a_{t+6}
Action Decoder
Video Model
Action Transformer
"Turn left"
Action Encoder
a_{t+1} a_{t+2} a_{t+3} a_{t+4} a_{t+5} a_{t+6}
[x,y]
Key
o_{t-1}
o_t
A_{t-1}
A_t
Query

(a) **Flow-matching action expert** (b) **Joint Attention Masking**

Figure 2. **Model details.** (a) The iterative denoising process for driving trajectory estimation: starting from random noise, VaVAM estimates the sequence of driving waypoints (green dots), conditional to high-level commands (e.g., 'turn left') and video model features. (b) The joint attention between the video model tokens o and action tokens A at training time.

Learning Objective. We learn a conditional denoising vector field v_θ, which defines how to progressively transform noisy trajectories back into expert-like behavior. The dataset $\mathcal{D} = \{(O_t, A_t, c_t)\}$ is composed of image sequences $O_t = [o_t, \dots, o_{t-N}]$ observed up to N past frames, expert trajectories $A_t = [a_{t+1}, \dots, a_{t+H}]$ of future positions over horizon H, and commands $c_t \in \{\text{left,right,straight}\}$.

Flow matching defines the forward process as a linear interpolation between the expert action A_t and Gaussian noise $\epsilon \sim \mathcal{N}(0,I)$, controlled by a noise scale $\tau \in [0,1]$:

$$A_t^\tau = \tau A_t + (1-\tau)\epsilon, \quad \epsilon \sim \mathcal{N}(0,I) \tag{1}$$

The model is then trained to predict the denoising vector field v_θ from A_t^τ:

$$L^\tau(\theta) = \mathbb{E}_{p(A_t|O_t,c_t),q(A_t^\tau|A_t)} ||v_\theta(A_t^\tau, O_t, c_t) - u(A_t^\tau|A_t)||^2 \tag{2}$$

where $q(A_t^\tau|A_t)$ is the forward process defined above and $u(A_t^\tau|A_t)$ is the optimal transport vector field. The optimal transport vector field $u(A_t^\tau|A_t)$ represents the ideal direction in which noisy actions should move to become expert actions. The learned vector field v_θ approximates this optimal transport. Note that the video model is kept frozen during the training of the action expert. During inference, we generate action sequences by integrating the learned vector field:

$$A_t^{\tau+\delta} = A_t^\tau + \delta \cdot v_\theta(A_t^\tau, O_t, c_t) \tag{3}$$

using 10 steps of the forward Euler method, starting from random noise $A_t^0 \sim \mathcal{N}(0,I)$.

3. Data and Training Strategy

3.1. Data sources

Our desiderata for the data were to find a large dataset of video data for the pre-training and a sufficient amount of data with perception and trajectories synchronized for fine-tuning.

To that end, we train the video model and VAM on a collection of three datasets: OpenDV [43], a massive dataset of raw front-camera driving videos, and nuPlan [7] and nuScenes [6], dedicated automotive datasets captured with multiple sensors.

OpenDV [43]. Since the video model does not require any annotation, metadata, or additional sensors, we can leverage the web-scale dataset OpenDV. OpenDV is a large-scale dataset [43] which consists of over 1,700 hours of high quality unannotated and unsynchronized driving videos collected from the internet. OpenDV provides broad and diverse visual coverage of real-world driving conditions, which makes it suited to learn our auto-regressive features for the video model.

nuPlan [7] & nuScenes [6] are high quality standard driving datasets, often used to benchmark methods for diverse taks. They provide synchronized camera and ego-trajectory data, enabling supervised training of action models. But are not the same order of magnitude as OpenDV.

For all datasets, we use only the front-facing camera and extract short 4-second video clips at 2 FPS, resulting in eight 512×288 frames per clip. Details are provided in App. D.

3.2. Training

Learning features for the video model. We train the video model in two stages. In the first phase, we pretrain it on the broad data distribution of OpenDV: this is the most computationally expensive step. Following pretraining, we move to a fine-tuning phase where the model is adapted to more specific autonomous driving data. Fine-tuning is performed using the same autoregressive next-token prediction objective, ensuring consistency with pretraining while adapting the model's internal representations to better align with target-domain distributions. We construct a training mix that includes a subset of OpenDV as well as nuPlan and nuScenes that more closely match our benchmark data distribution (the exact proportion is given in App. D). We use an ImageNet pretrained VQGAN-based tokenizer from LlamaGen [37] and a GPT-style transformer [34]. We give all details on input resolution, context length, model sizes, and optimizer in App. E. While recent approaches rely on diffusion models [17, 18], our video model stands out as the only open, large-scale autoregressive model trained on real-world driving data from OpenDV.

Action learning with VaVAM. We learn the action expert from the front-cam and GPS positions of the nuPlan and nuScenes datasets. We predict future trajectory for 3 seconds at 2Hz, i.e., 6-steps, as illustrated on Fig. 2a. VaVAM is initialized with the pre-trained video model and remains frozen. During training we use a block attention pattern (Fig. 2b) for the joint attention, which enables training, and thus inference, with variable context lengths. The action matches the number of layers of the video model, and has a hidden dimension of 192 ($= 768/4$) for the FFN layers, while the tokens are projected for the joint attention to match the hidden dimension of the video model (768). This allows reducing the number of parameters of the action expert (21M v.s. 185M for the video model), and the overall computational cost of VaVAM. We use a beta noise schedule for flow matching [3]. For the optimization we keep the AdamW optimizer, with details given in App. E.

4. Experiments

4.1. Closed-loop evaluation with NeuroNCAP

Protocol. To evaluate safety-critical behavior beyond open-loop metrics, we use NeuroNCAP [28], a photorealistic, NeRF-based simulator supporting data-driven closed-loop evaluation. Unlike synthetic [15] or view-reprojection [1, 2] systems, NeuroNCAP produces novel views from real data and inserts adversarial agents to mimic critical Euro NCAP scenarios: ego-lane obstacles, frontal collisions, and cross-traffic. Driving decisions are executed in simulation, with observations updated accordingly. The ego-vehicle and the adversarial agents are initialized so that, under constant speeds and steering angles, a collision would occur in $\sim$4 seconds. Predicted trajectories from VaVAM are converted into low-level control commands (steering, throttle, brake) via an LQR controller implemented within the NeuroNCAP simulator.

NeuroNCAP reports two core metrics: the *collision rate* (lower is better) and the *NeuroNCAP Score (NNS)* (higher is better) which is derived from the collision rate and severity: zero collisions give a perfect score of 5.0, which is lowered for more collisions or collisions at higher speeds (see App. F).

Baselines. *Rule-based baselines:* Base-U and Base-V use perception, respectively, from UniAD [22] and VAD [24], braking when objects are detected in a corridor 4m wide and $2 \times$ (ego speed) long. *End-to-end planners:* UniAD and VAD, which use 360° camera inputs and predict ego trajectories, with bird's-eye-view representations that require heavy annotation at train time (HD-map, tracking of agents).

These learned baselines apply a post-processing step at inference, refining predicted trajectories by minimizing a cost over candidate options based on smoothness and predicted collisions. This uses a future occupancy map trained with dense human annotations. Designing such post-processing requires supervision and expert-crafted heuristics tailored to specific risk types (e.g., avoiding vehicle crashes), well aligned with the focus of NeuroNCAP. However, this approach is narrow in scope and does not generalize to broader driving contexts involving traffic lights, stop signs, etc. While our method could also benefit from such post-processing, the focus of this work is seeking strong performance without supervision.

4.2. Main results

SOTA comparison. Tab. 1 summarizes the performance of all models on the NeuroNCAP benchmark. Rule-based baselines (Base-U, Base-V) perform strongly in static

Table 1. Performance on NeuroNCAP benchmark: VaVAM obtains SOTA scores on the frontal scenarios. * Static scenarios, sensitive to annotation-dependent post-processing. † Side scenarios, sensitive to multiview-cameras. ‡ Baseline reproduced by us.

MODEL	POST-PROC.	NeuroNCAP Score ↑				Collision rate (%) ↓			
		Avg.	Stat.*	Frontal	Side†	Avg.	Stat.*	Frontal	Side†
Baselines — Trained with hand-labeled annotations, 360° View									
Base-U	N/A	2.65	4.72	1.80	1.43	69.90	9.60	100.00	100.00
Base-V	N/A	2.67	4.82	1.85	1.32	68.70	6.00	100.00	100.00
UniAD	✗	0.73	0.84	0.10	1.26	88.60	87.80	98.40	79.60
VAD	✗	0.66	0.47	0.04	1.45	92.50	96.20	99.60	81.60
UniAD	✓	1.84	3.54	0.66	1.33	68.70	34.80	92.40	78.80
UniAD‡	✓	2.08	3.58	1.18	1.48	61.10	31.20	78.80	73.20
VAD	✓	**2.75**	**3.77**	1.44	**3.05**	**50.70**	**28.70**	73.60	**49.80**
VaVAM — Trained on raw data, Front-cam only									
VaVAM	✗	2.62	3.13	**2.67**	2.07	52.70	47.20	**50.00**	60.80

obstacle scenarios, achieving NeuroNCAP Scores (NNS) above 4.7. However, they fail in dynamic and interactive situations, suffering from a 100% collision rate in frontal and side scenarios. This highlights their inability to anticipate or react to moving hazards.

In the absence of post-processing, the learned models UniAD and VAD perform very poorly overall, with collision rates of 80–100%. Post-processing techniques, such as occupancy filtering of predicted trajectories, considerably improve their performance, especially in static settings. However, even with these corrections, challenges remain for dynamic and frontal collisions. While the original version of VAD performs poorly, the version using UniAD's post-processing presented at the NeuroNCAP benchmark achieves the lowest overall collision rate, outperforming VaVAM by 2 percent points.

In contrast, VaVAM operates end-to-end using only front-facing cameras and without access to labels or inference post-processing. Nevertheless, VaVAM excels in safety-critical frontal collision scenarios reaching SOTA scores, achieving an NNS of 2.67, and surpasses UniAD (which has 360° camera input) in side-collision cases. Remark that UniAD and VaVAM are at comparable model-size scales, with 125 million and 206 million parameters, respectively.

Qualitative results, shown in Fig. 3, further illustrate these differences. In challenging frontal collision situations, UniAD detects the oncoming vehicle but fails to execute a safe maneuver, adhering too rigidly to its planned trajectory. In contrast, VaVAM, without explicit supervision or rule programming for such cases, naturally deviates from its nominal path and performs an evasive maneuver, successfully avoiding the hazard. This highlights the strength of our large-scale self-supervised pretraining approach in enabling flexible, safety-aware behavior in complex real-world driving scenarios.

5. Conclusion

We introduced VaVAM, an end-to-end driving system that combines generative video pretraining with action learning

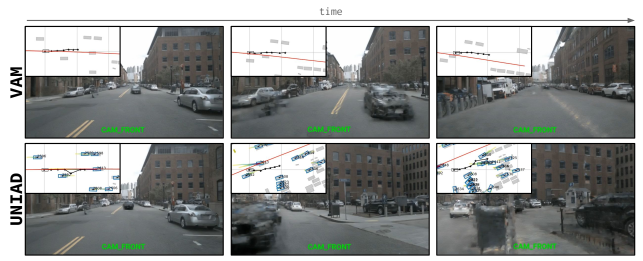


Figure 3. **Driving behavior comparison on NeuroNCAP [28].** UniAD [22] and VaVAM are evaluated in a frontal collision scenario. Images: frontal camera. Inset plots: top view of scene evolution, with objects (gray boxes) and guiding path (red line) highlighted for visualization only. While UniAD detects and forecasts the oncoming vehicle (blue), it fails to react. VaVAM successfully deviates to avoid collision and safely returns to its lane, without explicit supervision.

from demonstrations, without requiring manual annotations. VaVAM achieves strong closed-loop performance and sets a new state of the art in safety-critical scenarios, without relying on post-processing at inference. These findings support the potential of video-based generative modeling as a foundation for scalable driving agents. Future directions include reward-based learning, multi-camera inputs, and improved tokenization. We open-source our models, tools, and training protocols to support the community and foster further progress in autonomous driving research.

Limitations. Our video model relies solely on the front camera, limiting performance in scenarios involving side interactions. Future work could extend VaVAM to multi-camera setups using large-scale datasets such as L2D [38]. The imitation learning training requires the ego position and is currently conducted on smaller, synchronized datasets (nuPlan and nuScenes). Recently, GEM [18] proposed pseudo-labeling to extract the ego-position from raw videos, allowing to leverage GPS pseudo-annotation on OpenDV-scale datasets. This strategy would allow future works to scale up the imitation learning phase.

References

[1] Alexander Amini, Igor Gilitschenski, Jacob Phillips, Julia Moseyko, Rohan Banerjee, Sertac Karaman, and Daniela Rus. Learning robust control policies for end-to-end autonomous driving from data-driven simulation. *RAL*, 2020. 3, 8

[2] Alexander Amini, Tsun-Hsuan Wang, Igor Gilitschenski, Wilko Schwarting, Zhijian Liu, Song Han, Sertac Karaman, and Daniela Rus. Vista 2.0: An open, data-driven simulator for multimodal sensing and policy learning for autonomous vehicles. In *ICRA*, 2022. 3, 7, 8

[3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi 0$: A vision-language-action flow model for general robot control. *CORR*, 2024. 2, 3

[4] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. 1, 7

[5] Thibault Buhet, Émilie Wirbel, Andrei Bursuc, and Xavier Perrotton. PLOP: probabilistic polynomial objects trajectory prediction for autonomous driving. In *CoRL*, 2020. 1

[6] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020. 3, 7, 9

[7] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric M. Wolff, Alex H. Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. *CoRR*, 2021. 3, 8

[8] Hila Chefer, Uriel Singer, Amit Zohar, Yuval Kirstain, Adam Polyak, Yaniv Taigman, Lior Wolf, and Shelly Sheynin. Videojam: Joint appearance-motion representations for enhanced motion generation in video models. *CoRR*, 2025. 1, 7

[9] Raphael Chekroun, Marin Toromanoff, Sascha Hornauer, and Fabien Moutarde. Gri: General reinforced imitation and its application to vision-based autonomous driving. *Robotics*, 2023. 1, 7

[10] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *RSS*, 2023. 7

[11] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *PAMI*, 2023. 1, 7

[12] Felipe Codevilla, Antonio M López, Vladlen Koltun, and Alexey Dosovitskiy. On offline evaluation of vision-based driving models. In *ECCV*, 2018. 8

[13] Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. Exploring the limitations of behavior cloning for autonomous driving. In *ICCV*, 2019. 7

[14] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *CoRL*, 2023. 8

[15] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. CARLA: an open urban driving simulator. In *CoRL*, 2017. 1, 3, 7, 8

[16] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2020. 2, 8

[17] Shenyuan Gao, Jiazhi Yang, Li Chen, Kashyap Chitta, Yihang Qiu, Andreas Geiger, Jun Zhang, and Hongyang Li. Vista: A generalizable driving world model with high fidelity and versatile controllability. In *NeurIPS*, 2024. 1, 3, 8

[18] Mariam Hassan, Sebastian Stapf, Ahmad Rahimi, Pedro M. B. Rezende, Yasaman Haghighi, David Brüggemann, Isinsu Katircioglu, Lin Zhang, Xiaoran Chen, Suman Saha, Marco Cannici, Elie Aljalbout, Botao Ye, Xi Wang, Aram Davtyan, Mathieu Salzmann, Davide Scaramuzza, Marc Pollefeys, Paolo Favaro, and Alexandre Alahi. GEM: A generalizable ego-vision multimodal world model for fine-grained ego-motion, object dynamics, and scene composition control. *CoRR*, 2024. 3, 4, 9

[19] Jeffrey Hawke, Richard Shen, Corina Gurau, Siddharth Sharma, Daniele Reda, Nikolay Nikolov, Przemysław Mazur, Sean Micklethwaite, Nicolas Griffiths, Amar Shah, et al. Urban driving with conditional imitation learning. In *ICRA*, 2020. 7

[20] Anthony Hu, Gianluca Corrado, Nicolas Griffiths, Zak Murez, Corina Gurau, Hudson Yeo, Alex Kendall, Roberto Cipolla, and Jamie Shotton. Model-based imitation learning for urban driving. In *NeurIPS*, 2022. 1, 7

[21] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. GAIA-1: A generative world model for autonomous driving. *CoRR*, 2023. 1, 2, 8

[22] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, Lewei Lu, Xiaosong Jia, Qiang Liu, Jifeng Dai, Yu Qiao, and Hongyang Li. Planning-oriented autonomous driving. In *CVPR*, 2023. 1, 3, 4, 7, 8

[23] Xiaosong Jia, Penghao Wu, Li Chen, Jiangwei Xie, Conghui He, Junchi Yan, and Hongyang Li. Think twice before driving: Towards scalable decoders for end-to-end autonomous driving. In *CVPR*, 2023. 7

[24] Bo Jiang, Shaoyu Chen, Qing Xu, Bencheng Liao, Jiajie Chen, Helong Zhou, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Vad: Vectorized scene representation for efficient autonomous driving. In *ICCV*, 2023. 1, 3, 7, 8

[25] Seungjae Lee, Yibin Wang, Haritheja Etukuru, H. Jin Kim, Nur Muhammad (Mahi) Shafiullah, and Lerrel Pinto. Behavior generation with latent actions. In *ICML*, 2024. 7

[26] Xiaofan Li, Yifu Zhang, and Xiaoqing Ye. Drivingdiffusion: Layout-guided multi-view driving scenarios video generation with latent diffusion model. In *ECCV*, 2024. 8

[27] Xiaodan Liang, Tairui Wang, Luona Yang, and Eric Xing. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. In *ECCV*, 2018. 1, 7

[28] William Ljungbergh, Adam Tonderski, Joakim Johnander, Holger Caesar, Kalle Åström, Michael Felsberg, and Christoffer Petersson. Neuroncap: Photorealistic closed-loop safety testing for autonomous driving. In *ECCV*, 2024. 1, 3, 4, 8

[29] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 9

[30] Jianbiao Mei, Tao Hu, Xuemeng Yang, Licheng Wen, Yu Yang, Tiantian Wei, Yukai Ma, Min Dou, Botian Shi, and Yong Liu. Dreamforge: Motion-aware autoregressive video generation for multi-view driving scenes. *CoRR*, 2024. 8

[31] Jongjin Park, Younggyo Seo, Chang Liu, Li Zhao, Tao Qin, Jinwoo Shin, and Tie-Yan Liu. Object-aware regularization for addressing causal confusion in imitation learning. In *NeurIPS*, 2021. 7

[32] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *CoRR*, 2024. 7

[33] Dean Pomerleau. ALVINN: an autonomous land vehicle in a neural network. In *NIPS*, 1988. 1, 7

[34] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019. 2, 3, 9

[35] Katrin Renz, Long Chen, Ana-Maria Marcu, Jan Hünermann, Benoît Hanotte, Alice Karnsund, Jamie Shotton, Elahe Arani, and Oleg Sinavski. Carllava: Vision language models for camera-only closed-loop driving. *CoRR*, 2024. 7

[36] Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. Gaia-2: A controllable multi-view generative world model for autonomous driving. *CoRR*, 2025. 8

[37] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. Autoregressive model beats diffusion: Llama for scalable image generation. *CoRR*, 2024. 2, 3, 8, 9

[38] Yaak LeRobot team. Lerobot goes to driving school: World's largest open-source self-driving dataset. *https://www.huggingface.com/blog/lerobot-goes-to-driving-school*, 2025. 4

[39] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006. 8

[40] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *CVPR*, 2020. 1, 7

[41] Veo-Team, :, Agrim Gupta, Ali Razavi, Andeep Toor, Ankush Gupta, Dumitru Erhan, Eleni Shaw, Eric Lau, Frank Belletti, Gabe Barth-Maron, Gregory Shaw, Hakan Erdogan, Hakim Sidahmed, Henna Nandwani, Hernan Moraldo, Hyunjik Kim, Irina Blok, Jeff Donahue, José Lezama, Kory Mathewson, Kurtis David, Matthieu Kim Lorrain, Marc van Zee, Medhini Narasimhan, Miaosen Wang, Mohammad Babaeizadeh, Nelly Papalampidi, Nick Pezzotti, Nilpa Jha, Parker Barnes, Pieter-Jan Kindermans, Rachel Hornung, Ruben Villegas, Ryan Poplin, Salah Zaiem, Sander Dieleman, Sayna Ebrahimi, Scott Wisdom, Serena Zhang, Shlomi Fruchter, Signe Nørly, Weizhe Hua, Xinchen Yan, Yuqing Du, and Yutian Chen. Veo 2. *tech report*, 2024. 1, 7

[42] Zhenhua Xu, Yujia Zhang, Enze Xie, Zhen Zhao, Yong Guo, Kwan-Yee K. Wong, Zhenguo Li, and Hengshuang Zhao. Drivegpt4: Interpretable end-to-end autonomous driving via large language model. *RAL*, 2024. 7

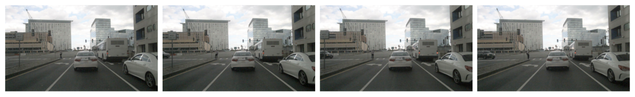
[43] Jiazhi Yang, Shenyuan Gao, Yihang Qiu, Li Chen, Tianyu Li, Bo Dai, Kashyap Chitta, Penghao Wu, Jia Zeng, Ping Luo, Jun Zhang, Andreas Geiger, Yu Qiao, and Hongyang Li. Generalized predictive model for autonomous driving. In *CVPR*, 2024. 3, 8

[44] Tengju Ye, Wei Jing, Chunyong Hu, Shikun Huang, Lingping Gao, Fangzhen Li, Jingke Wang, Ke Guo, Wencong Xiao, Weibo Mao, et al. Fusionad: Multi-modality fusion for prediction and planning tasks of autonomous driving. *CoRR*, 2023. 8

[45] Jiang-Tian Zhai, Ze Feng, Jihao Du, Yongqiang Mao, Jiang-Jiang Liu, Zichang Tan, Yifu Zhang, Xiaoqing Ye, and Jingdong Wang. Rethinking the open-loop evaluation of end-to-end autonomous driving in nuscenes. *CoRR*, 2023. 8

† Corresponding authors; {florent.bartoccioni, elias.ramzi}@valeo.com
* Work done while at valeo.ai, now at H company.

A. Further analysis


(a) Input context frames (real images).

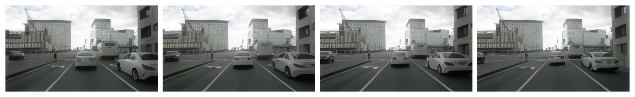

(b) Generated frames with the video model. The input context frames are from the nuScenes dataset [6].

Figure 4. **Video generation.** Given 4 context frames ((a)), the video model predicts 4 future frames with coherent structure and motion.

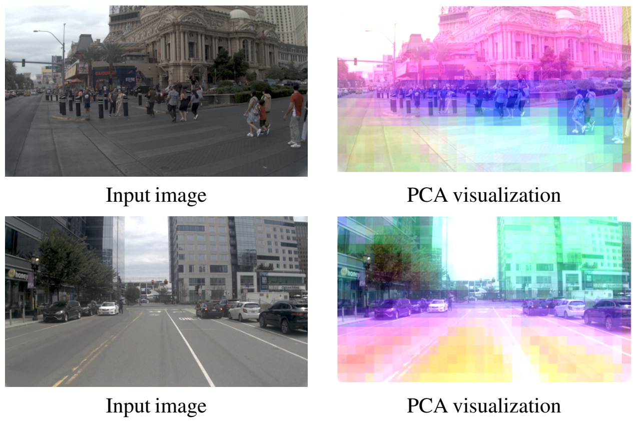


Figure 5. **Semantic structure in video model features.** We map the top-3 PCA components of the video model's layer-22 features to the RGB primaries. Similar colors cluster around the same object classes (e.g., vehicles, pedestrians, road), revealing emergent semantics without supervision. Input images from the nuScenes dataset [6].

Qualitative analysis of the video model. Fig. 4 shows a sample generation: given 4 context frames (top row), the video model generates spatially coherent future frames (bottom row), preserving scene layout and dynamics. In Fig. 5, we visualize the internal representations of the video model using principal component analysis (PCA) to map the top-3 principal components of the features from the 22nd layer onto the primaries of an RGB image. The similar colors clustering around objects of the same class (cars, pedestrians, road) reveal strong, emergent semantic structure on the features, even if training forgoes any explicit labeling.

Flow matching ablation. We adopt flow matching to model action trajectories, as it directly learns vector fields that capture complex, multimodal distributions. Discrete approaches, such as action quantization and categorical sampling [25], struggle with such long-tail distributions and offer limited expressiveness. Simple regression, on the other hand, produces deterministic outputs that average over possible actions (e.g., driving straight when faced with left/right options), leading to unsafe and ambiguous behavior. In contrast, flow matching allows sampling diverse, goal-consistent trajectories by refining noisy inputs through learned gradient fields [10].

Tab. 2 highlights the importance of the flow-matching formulation for trajectory generation. Performance degrades sharply across all NeuroNCAP scenarios, particularly in frontal scenarios — NNS drops from 2.67 to 0.70, collision rate rises from 50.0 to 96.0% — for a simple mean-squared error regression objective (VaVAM-MSE) instead of flow-matching, confirming the latter's advantage in modeling complex and multimodal distributions of driving behavior. In long-tailed datasets, dominated by straight trajectories, the flow-based predictor allows capturing rare but critical maneuvers, such as evasive swerves or hard braking, far more reliably than direct regression.

B. Related work

End-to-end driving primarily follows two paradigms: reinforcement learning (RL) and imitation learning (IL). *RL* trains agents through trial and error towards a reward, typically in simulation [9, 27, 40]. Agents use visual encoders pretrained on tasks such as semantic segmentation [40]. While RL can exploit privileged simulation signals [20], sim-to-real transfer remains challenging due to visual and behavioral domain gaps in current simulators [2, 15]. *IL* [13, 19] bypasses reward design by mimicking expert demonstrations, easing real-world deployment. Behavior cloning [22, 33] treats IL as supervised learning but requires large and diverse datasets to handle rare cases and avoid overfitting [31]. TransFuser [11] and MILE [20] ease data demands by relying on dense annotations (e.g., HD maps and vehicles bounding boxes), but that limits scalability [23]. Recent work such as UniAD [22] integrates perception, prediction, and planning in a unified framework. VAD [24] uses vectorized scene representations for faster inference. Emerging foundation models such as DriveGPT4 [42] and CarLLaVA [35] aim to bypass dense labels entirely using vision-language pretraining. In this work, we retain the simplicity of IL but forgo all human annotations. Instead, we scale unsupervised generative video pretraining, enabling strong closed-loop performance without dense supervision.
Video modeling. Video generative models have seen rapid recent progress [8, 32, 41]. Sora [4] employs a latent transformer diffusion model trained on images and videos at different aspect ratios, enabling very diverse generation. Movie Gen [32] can produce videos of up to 16

Table 2. **Ablation of the action expert learning.** We compare the full VaVAM model against a variant (VaVAM-MSE) where the flow-matching trajectory head is replaced with a standard regression loss. The flow-matching formulation yields significantly better NeuroNCAP scores and lower collision rates, especially in frontal collision scenarios, confirming its ability to model diverse and safe behaviors.

MODEL	POST-PROC.	NEURONCAP SCORE ↑				COLLISION RATE (%) ↓			
		AVG.	STAT.	FRONTAL	SIDE	AVG.	STAT.	FRONTAL	SIDE
VAVAM-MSE	✗	1.50	2.62	0.70	1.18	80.60	66.60	96.0	79.20
VAVAM	✗	2.62	3.13	2.67	2.07	52.70	47.20	50.0	60.80

seconds at 16 fps. Some models specifically target driving scenes [17, 21, 26, 30, 36], such as GAIA-1 [21], which uses 5,000-hour driving data to scale up generation, and GAIA-2 [36], with enhanced controllability and expanded geographic and vehicle diversity. In this work, we show that large-scale generative video pretraining is highly beneficial for autonomous driving. Rather than aiming only at video synthesis, our model leverages the learned visual dynamics and temporal reasoning skills acquired during generative pretraining to boost closed-loop driving performance — without requiring any explicit labels, costly annotations, or task-specific supervision. This suggests that generative video models can serve as a strong foundation for building robust, scalable, end-to-end autonomous driving systems.

Open-loop and closed-loop evaluations. *Open-loop evaluation* [22, 24, 44] compares predicted trajectories to expert demonstrations but resets the system at each step, preventing error accumulation and feedback assessment. It also penalizes all deviations from the expert, even reasonable ones [12, 14, 45]. *Closed-loop evaluation*, in contrast, lets model decisions affect future observations, enabling realistic assessment of driving behavior. However, for end-to-end camera-based systems, it requires rendering plausible views from new positions, a non-trivial task. Simulators such as CARLA [15] offer this but suffer from domain gaps. Vista [1, 2] reprojects real images but does not simulate interactions with dynamic agents. NeuroNCAP [28] overcomes both by using neural rendering with real-world photorealism and dynamic agents. We evaluate VaVAM in closed-loop on NeuroNCAP and show it achieves state-of-the-art performance in safety-critical scenarios.

C. Video model formalism

Each input frame $X_t \in \mathbb{R}^{h\times w\times c}$ is tokenized into a discrete spatial grid of codes using a pretrained LlamaGen-VQGAN [16, 37]. The encoder f_θ produces a latent feature map $e \in \mathbb{R}^{h'\times w'\times d}$, which is discretized into tokens $q^{(i,j)}$ by finding the nearest neighbors in LlamaGen's codebook $\{e_k\}_{k=1}^{K}$:

$$q^{(i,j)} := \underset{k}{\arg\min} \|e^{(i,j)} - e_k\|_2.$$

This results in a compact sequence of tokens that significantly reduce video data dimensionality.

To model temporal dynamics, this multi-frame token sequence is fed to an autoregressive transformer decoder. The model learns to predict the next token in the sequence from all preceding tokens, capturing joint spatio-temporal structure. Formally, for a sequence $\mathcal{Q} = [q^0, \ldots, q^{n-1}]$, we minimize:

$$\mathcal{L}_\theta = -\sum_{i=1}^{n} \log P(q^i | q^0, \ldots, q^{i-1}; \theta).$$

D. Data Details and Processing

OpenDV [43] The OpenDV dataset is the largest driving dataset publicly available to date, with more than 1,700 hours of driving videos, collected at 60 FPS, resulting in over 360 million frames. The dataset comprises single-camera front-cam videos collected from YouTube, with indicated durations of intros (usually 90 seconds) and outros (usually 60 seconds) for trimming, to avoid artifacts such as title sequences and closing credits. Most of the videos are at or close to Full HD (1920×1080) resolution.

We include in our data only the videos at exactly Full HD to avoid issues of aspect ratio distortion. That meant discarding 1.3% of the videos (2.5% of the total duration). Using FFMPEG [39], we extracted the frames for the remaining videos at 10 FPS and 512×288 pixels, discarding intros and outros. We stored the frames in individual JPEG files. We extract overlapping clips of 8 frames at 2 FPS to train the video model. Only a front camera is available for this dataset, without any metadata.

nuPlan [7] The nuPlan dataset contains around 1,200 hours of driving scenarios recorded in Las Vegas (838 hours), Boston, Pittsburgh, and Singapore. In particular, among the 1,200-hour raw data, approximately 94-hour recordings contain sensor information (LiDAR and cameras) with a sampling rate of 10 Hz. Our project only employs the RGB images in 1274 recorded videos and the ego position. More specifically, we only use the front camera instead of the eight cameras that cover the 360-degree view around the ego vehicle. We collect from nuPlan 2,833,723 frames for training and 492,477 for validation, together with trajectory extracted from the ego positions. We also extract overlapping clips of 8 frames at 2 Hz from the original 10 Hz video sequences.

nuScenes [6] The nuScenes dataset contains 1,000 driving scenes of 20 seconds collected in Boston and Singapore. Similarly to nuPlan, while nuScenes includes 6 cameras, LIDAR, RADAR, etc., we restrict its usage to the front camera and ego position in this work. That results in a dataset of 28,130 training frames and 6,019 validation frames. We also extract overlapping clips of 8 frames. The dataset is natively synchronized at 2Hz.

As detailed in subsequent sections, we use the datasets for different steps of the video model and VaVAM training. Specifically, we use OpenDV only for pre-training the video model. A mix of the three datasets to fine-tune the video model. For both training steps, only the front camera is used, making those steps completely unsupervised and, thus, highly scalable. Finally, we use nuScenes and nuPlan to learn VaVAM through imitation learning on the ego trajectory. That training stage requires access to the ground-truth expert trajectory, although, interestingly, recent approaches such as GEM [18], explore the use of pseudo-annotations for the ego trajectory, paving the way to scaling action learning to OpenDV-size datasets.

E. Implementation Details and Training Infrastructure

Image tokenizer In practice, we use a pre-trained image tokenizer, LlamaGen [37], which is based on the VQGAN architecture. Specifically, we use the stride=16 tokenizer, which has 72M parameters. It has a vocabulary size of 16,384 with codewords of 8 dimensions. We use images of size 512×288, resulting in a token map of 32×18, or 576 tokens.

The video model is based on a GPT-2 transformer architecture [34]. We train it with a context length of 8 frames, resulting in a maximum context of 4,608 tokens. It has 24 layers, a vocabulary size of 16,384, with a width of 768. This results in a codebook of size 12.6M. The heads dimensionality is 128. We set a standard multiplication factor of 4 for the FFN hidden dimensionality. We optimize it with AdamW [29], a base learning rate of $4e$-3, a decoupled weight decay of $1e$-7, and $\beta = (0.9, 0.95)$ while clipping the gradient with a norm of 1.0. Finally, we initialize with a standard deviation of $3e$-2. These values are fixed by random search. We train all our models with a batch size of 384. We pre-train the model on approximately 60 million overlapping windows.

VaVAM predicts the trajectory for the 6 next timesteps at 2 Hz, i.e., for 3 seconds. The dimensionality of VaVAM attention layers is identical to its video model companion for the joint attention. However, VaVAM's MLP layers dimensionality is reduced by a factor of 4 with respect to the video model dimensionality for efficient action sampling, i.e., 192.

We use a learning rate equal to 0.0194, an initialization standard deviation of 0.0086, and similar optimizer parameters to the video model. For the flow matching loss, we follow $\pi 0$ and use a beta distribution for the noise schedule and 10 steps for denoising at inference time. We efficiently train our model with different observation context lengths using a block attention pattern (Fig. 2b). That allows training the action expert to handle varying lengths of temporal context from one training clip.

F. Experimental protocol

The NeuroNCAP score is formally defined as:

$$\text{NNS} = \begin{cases} 5.0 & \text{if no collision} \\ 4.0 \cdot \max(0, 1 - v_i / v_r) & \text{otherwise} \end{cases} \tag{4}$$

where v_i is the impact velocity and v_r is the reference impact speed that would occur if no action is performed. Higher scores reflect better collision avoidance and mitigation.

G. Acknowledgements

This work was partially supported by the ANR Multi-Trans project (ANR-21-CE23-0032). It was initially explored using HPC resources from GENCI–CINES (Grant 2023-A0141014181), and most of its results were obtained using HPC resources from GENCI–IDRIS (Grant 2024-GC011015459). We acknowledge the EuroHPC Joint Undertaking for awarding this project access to the EuroHPC supercomputer LEONARDO, hosted by CINECA (Italy) and the LEONARDO consortium, through a EuroHPC AI and Data-Intensive Access call.