# GAZEFLOW: From Human Gaze Behavior to Generative Egocentric Gaze Prediction

**Sheng Zhao**
Department of Computer Science
University of Rochester
szhao32@ur.rochester.edu

**Weikai Lin**
Department of Computer Science
University of Rochester
wlin33@ur.rochester.edu

**Yuhao Zhu**
Department of Computer Science
University of Rochester
yzhu@rochester.edu

## Abstract

Egocentric gaze prediction enables many downstream applications but remains challenging, as human gaze is inherently stochastic. This stochasticity is constrained by structured temporal dynamics alternating between fixations and saccades, top-down influences from tasks, and bottom-up visual saliency. Based on this observation, we introduce GAZEFLOW, a framework that directly models gaze as a joint distribution of temporal gaze positions conditioned upon both top-down and bottom-up information. In particular, GAZEFLOW uses conditional flow matching (CFM): a learned velocity field iteratively transports a Gaussian noise sample into a plausible gaze trajectory drawn from this joint distribution. The velocity field is conditioned on bottom-up visual features extracted by a video encoder and on top-down task information obtained by globally querying these features. On standard datasets, GAZEFLOW achieves state-of-the-art performance on per-frame metrics, and the generated trajectories align better with human gaze temporal dynamics.

## 1 Introduction

The eye provides a direct window into the brain. Egocentric video captures visual experience from the wearer's perspective, offering a natural way to probe this link—provided that gaze is known. However, gaze is not directly observable in such videos without dedicated eye-tracking hardware, which requires calibration and introduces additional cost and complexity in real-world deployment. At the same time, access to gaze would enable a range of applications, including foveated video compression [16, 20, 34, 8], hazard anticipation in autonomous driving [36], and imitation learning in robotics [22]. A natural alternative is to infer where the wearer looks directly from the video itself, a problem known as egocentric gaze prediction [15, 24, 28].

Gaze patterns are notoriously difficult to predict because of the stochasticity, as widely recognized in prior work [24, 15, 28]: even for the same observer viewing the same scene, trajectories are not exactly reproducible. This stochasticity, however, is not purely random; rather, it is jointly constrained by three factors. First, human gaze trajectories exhibit structured dynamics under the oculomotor control, characterized by alternating cycles of saccades and fixational eye movements superimposed with occasional microsaccades [42]; this imposes strong temporal constraints on how gaze can evolve over time. Second, gaze is strongly task-dependent [44, 11, 41, 12]; information about the task, often revealed in future frames, can therefore inform where the observer is likely to look at the current

Preprint.

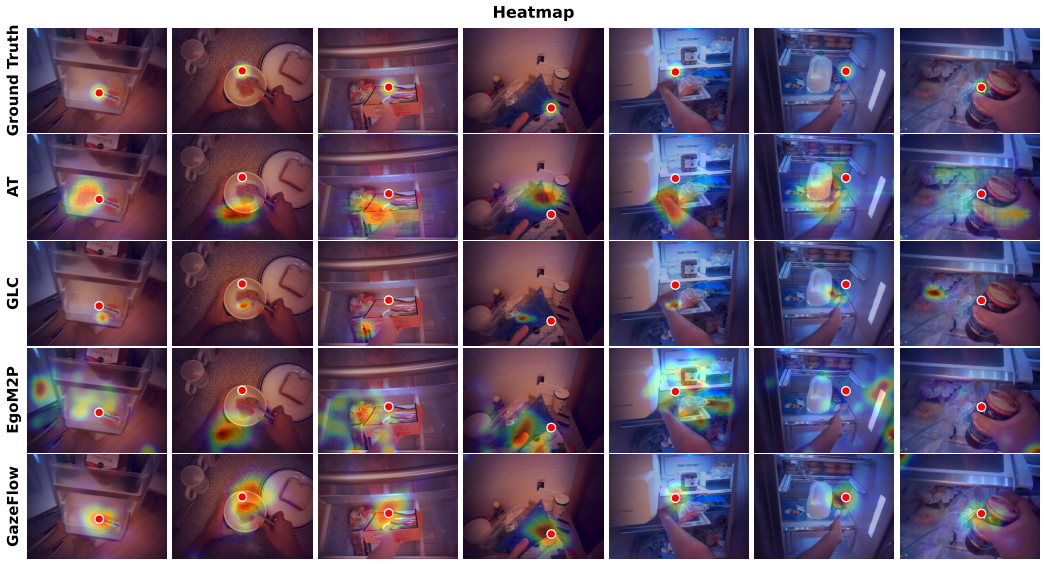


Figure. 1: **Qualitative comparison on EGTEA Gaze+.** Example frames where GAZEFLOW predicts a heatmap tightly localized on the ground-truth gaze positions (red dots) compared to existing methods (AT [15], GLC [24], EgoM2P [28]).

moment. Third, gaze is also influenced by stimulus-driven factors such as conspicuity and visual saliency [17, 46], which requires analyzing the spatio-temporal signal properties in the scene.

Taken together, gaze pattern is best modeled as a *joint* distribution of temporal gaze positions conditioned upon the *entire* video sequence. Any gaze trajectory arising from interacting with the scene captured in the video can then be viewed as a sample from this distribution. Modeling the joint distribution enables us to capture the global temporal dynamics of gaze, while conditioning on the full video allows us to account for both top-down influences, which arise from longer-term contextual and task-related information (on the order of seconds), and bottom-up saliency (e.g., local contrast or flicker), which operate at shorter timescales (tens of milliseconds).

We formalize these observations into a computational framework, GAZEFLOW. In GAZEFLOW, egocentric gaze prediction is formulated as a generative process, where each generated trajectory is a sample from the underlying human gaze distribution. In particular, we use conditional flow matching (CFM), a generative modeling method. We use a video encoder to extract bottom-up spatio-temporal features from video frames. To incorporate top-down influences, the encoded video embeddings are allowed to be globally queried by gaze at any time, yielding task-relevant information. The bottom-up and top-down embeddings are then used to condition the generation process.

We evaluate GAZEFLOW on two common egocentric video datasets, EGTEA Gaze+ [30] and Ego4D [10]. Not only does GAZEFLOW achieve state-of-the-art performance on per-frame metrics, the predicted trajectories also exhibit closer alignment with the temporal dynamics of human gaze. Figure 1 shows the qualitative results. In summary, this paper makes the following contributions:

- Starting from human gaze mechanisms, we first formulate egocentric gaze prediction as a conditional joint-distribution modeling problem.
- We propose a flow matching-based generative algorithm, where gaze trajectory generation is conditioned on both bottom-up (saliency) and top-down (task-specific) information from a video.
- We demonstrate that GAZEFLOW achieves state-of-the-art per-frame prediction accuracy on EGTEA Gaze+ and Ego4D, and is better aligned with the temporal dynamics of human gaze.

Table 1: **Comparison of egocentric gaze prediction paradigms.** $\mathcal{V}$ is the input video and $\mathcal{V}_{\leq t}$ its causal prefix, $\mathbf{g}_t \in \mathbb{R}^2$ is the gaze position at frame $t$, $\mathbf{g}_{1:T}$ the full trajectory, and $q_\theta(\cdot \mid \cdot)$ denotes a model's parametric conditional distribution. MSE: mean squared error, KL: Kullback–Leibler divergence, and BCE: binary cross-entropy. ✓ if satisfied, × if not satisfied, and ∼ if architecturally implied but not enforced.

| | | | | Human gaze mechanism | | | |
|---|---|---|---|---|---|---|---|
| Method | Paradigm | Prediction | Training objective | Bottom-up | Top-down | Joint | Stochastic |
| Li et al. [29] | Regression | $\mathbf{g}_t \mid \mathcal{V}$ | per-frame MSE | × | ∼ | × | × |
| GLC [24] | Marginal density estimation | $q_\theta(\mathbf{g}_t \mid \mathcal{V})$ | per-frame KL | ✓ | × | × | ✓ |
| DFG [49] | | $q_\theta(\mathbf{g}_t \mid \mathcal{V})$ | per-frame KL | ✓ | × | ∼ | ✓ |
| AT [15] | Autoregressive | $q_\theta(\mathbf{g}_t \mid \mathbf{g}_{t-1}, \mathcal{V}_{\leq t})$ | per-frame BCE | ✓ | ∼ | ∼ | ✓ |
| EgoM2P [28] | | $q_\theta(\mathbf{g}_t \mid \mathbf{g}_{<t}, \mathcal{V})$ | masked $L^2$ | ∼ | ∼ | ∼ | ✓ |
| **GAZEFLOW (Ours)** | Distribution mapping | $q_\theta(\mathbf{g}_{1:T} \mid \mathcal{V})$ | $L^2$ velocity field | ✓ | ✓ | ✓ | ✓ |

## 2 Related Work

**Mechanisms of Human Gaze.** Human gaze is shaped by three complementary mechanisms. Bottom-up saliency is driven by low-level visual features such as contrast, orientation, and motion [18, 17]. Top-down task control guides the eyes to anticipate upcoming actions and select objects relevant to the current goal [48, 26, 11, 44, 41]. Oculomotor control further imposes structured temporal dynamics on gaze. These dynamics alternate between saccades, fixations, and microsaccades, constraining how gaze evolves over time [42]. GAZEFLOW addresses these factors jointly: a visual encoder supplies bottom-up evidence, a task-token bank supplies top-down context, and conditional flow matching captures temporal structure.

**Joint Distribution Modeling.** Modeling a high-dimensional joint distribution generally falls into two families. Autoregressive factorization decomposes the joint into per-step conditionals [47, 4], while distribution mapping transports a simple prior to the data via a learned map, as in GANs [9], VAEs [23], diffusion [13], and flow matching [31, 45]. GAZEFLOW adopts the latter, sampling a full trajectory through one video-conditioned Ordinary Differential Equation (ODE) transport (Section 4).

**Egocentric Gaze Prediction.** Existing methods fall into four paradigms summarized in Table 1.

*Regression* directly regresses one gaze point per frame from the video, trained with per-frame MSE. Early egocentric methods regress over hand-crafted cues [29]. This collapses gaze to a single mode and cannot capture its multimodality. *Marginal density estimation* conditions on the video and independently predicts a per-frame heatmap per frame, trained with per-frame KL. GLC [24] produces this heatmap by correlating each frame with a global clip summary through a transformer. CSTS [25] additionally exploits sound, and DFG [49] learns a GAN. Per-frame factorization misses temporal coupling across frames.

*Autoregression* factors the joint distribution into a chain of per-frame conditionals on previously generated gaze. Attention Transition [15] predicts each fixation by recursively conditioning on the previous-frame fixation through an LSTM-based attention-transition module. EgoM2P [28] decodes VQ-VAE-quantized gaze tokens within a multimodal pretraining stack. This captures temporal dependency but accumulates errors under self-rollout. *Distribution mapping* models the full trajectory directly, sampling each trajectory by transporting a simple prior through a learned map. GAZEFLOW adopts conditional flow matching [31, 45]: a learned velocity network defines a video-conditioned velocity field that transports each Gaussian noise sample into a gaze trajectory. This jointly captures temporal coupling, stochasticity, and the bottom-up/top-down factors of natural gaze (Section 3).

## 3 GAZEFLOW

In egocentric gaze prediction task, a head-mounted video $\mathcal{V}$ of $T$ frames and its ground-truth gaze trajectory $\mathbf{g} \in \mathbb{R}^{2T}$ are recorded jointly while human performs a task. Gaze is inherently stochastic and depends on both what the human is doing and the visual saliency in the scene. So $\mathbf{g}$ is one realization of a distribution conditioned on $\mathcal{V}$. We write $p^\star(\mathbf{g} \mid \mathcal{V})$ for the population distribution

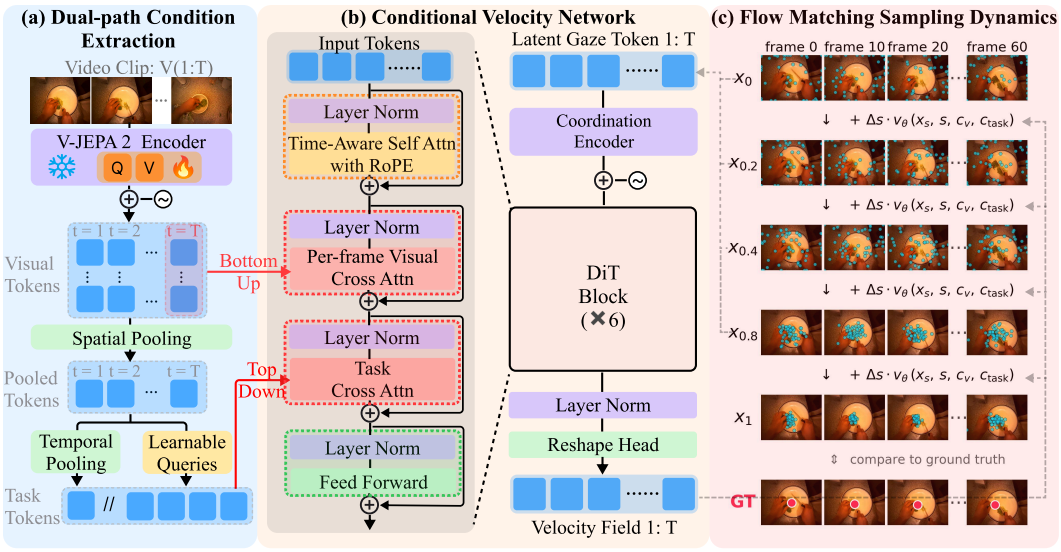


Figure. 2: **GAZEFLOW architecture.** (a) Dual-path condition extraction (Section 3.1). A LoRA-adapted V-JEPA 2 backbone produces bottom-up visual tokens $\mathbf{c}_v$; spatial–temporal pooling and learnable queries form the top-down task-token bank $\mathbf{c}_{\text{task}}$. (b) Conditional velocity network (Section 3.2). Latent gaze tokens encoding $\mathbf{x}_s$ and $s$ pass through six DiT blocks, each containing four sub-layers; a linear head outputs the velocity field $\mathbf{v}_\theta$. (c) Flow-matching sampling dynamics. Euler integration of $\mathbf{v}_\theta$ transports Gaussian noise $\mathbf{x}_0$ to a clean trajectory $\mathbf{x}_1$; the cyan sample cloud at $s \in \{0, 0.2, 0.4, 0.8, 1\}$ contracts toward the ground truth.

over all gaze trajectories given $\mathcal{V}$, and each annotated trajectory in the dataset is one sample drawn from $p^\star$. Our goal is to learn a model distribution $q_\theta(\mathbf{g} \mid \mathcal{V})$ that approximates $p^\star$, the underlying distribution we want to recover. Since we have only samples from $p^\star$, not a closed-form density, we adopt conditional flow matching (CFM) [31, 45], whose training objective is a sample-based regression that fits a velocity field transporting Gaussian noise to $q_\theta$ (see details in Appendix B).

Figure 2 shows the architecture of GAZEFLOW, which contains two main components: First, video feature extraction computes bottom-up visual tokens and top-down task tokens (Section 3.1). Second, the conditional velocity network uses these tokens for calculating a velocity transport that maps a Gaussian noise to target trajectory distribution (Section 3.2). We discuss how our framework can deal with training and inference on long videos (Section 3.3).

### 3.1 Dual-Pathway Video Condition Extraction

The video-conditioning module $E_\psi(\cdot)$ maps an input clip to two conditions. The first condition is bottom-up and frame-local. It comes from a pretrained video encoder. The second condition is top-down and clip-level. It comes from a learned task-token bank. This dual-pathway design follows the gaze mechanisms reviewed in Section 2: Human gazing depends on both low-level visual saliency and high-level task goals. Formally,

$$\mathbf{c} := \big(\mathbf{c}_v, \mathbf{c}_{\text{task}}\big) = E_\psi(\mathcal{V}), \tag{1}$$

where $\mathbf{c}_v$ contains frame-level visual tokens and $\mathbf{c}_{\text{task}}$ contains task-level tokens.

**Bottom-Up Feature Extraction.** The bottom-up path provides visual saliency for gaze prediction. We use V-JEPA 2 ViT-L [1] as the pretrained video encoder. We resize each frame ($T$ in total) to size of $256\times256$. With a patch size of $N = 16\times16$, each frame produces a $16\times16$ grid of spatial tokens. A trainable linear layer projects each token to a width $d = 256$:

$$\mathbf{c}_v = \mathbf{V} \in \mathbb{R}^{T\times N\times d}, \qquad N = 16 \times 16. \tag{2}$$

Each gaze token $\mathbf{h}_t$ (see Section 3.2) is allowed to cross-attend to the spatial tokens from the same frame $t$. This biases $\mathbf{c}_v$ toward frame-specific evidence. Modeling the temporal gaze correlation is handled by the velocity network (Section 3.2).

**Top-Down Task Token Bank.** Frame-level visual tokens alone do not encode task-level intent over the whole prediction horizon. We therefore aggregate the video tokens into a small bank of task tokens. This module provides the top-down task-level signal in GAZEFLOW.

We first average the video tokens over the spatial axis to obtain one summary $\bar{\mathbf{V}}_t$ per frame. We then average the frame summaries over time to obtain one global video token $\mathbf{p}$. This single token is stable, but it has limited representational power and could miss local task information. We therefore add a learnable-query mechanism based on multi-head cross-attention (MHCA) [47]. Specifically, we initialize $K$ learnable task queries $\mathbf{Q}_{\text{task}}$ attend to the $T$ frame summaries:

$$\bar{\mathbf{V}}_t = \frac{1}{N}\sum_{n=1}^{N} \mathbf{V}_{t,n,:}, \qquad \mathbf{p} = \frac{1}{T}\sum_{t=1}^{T} \bar{\mathbf{V}}_t, \qquad \mathbf{q}_{1:K} = \mathrm{MHCA}\big(\mathbf{Q}_{\text{task}}, \bar{\mathbf{V}}\big). \tag{3}$$

Here, $n$ indexes the $N$ spatial tokens within frame $t$. $\bar{\mathbf{V}} = [\bar{\mathbf{V}}_1; \ldots; \bar{\mathbf{V}}_T]$ stacks all the frame summaries. The global token $\mathbf{p}$ captures coarse task context while the queries $\mathbf{q}_{1:K}$ allow for more fine-grained, selective extraction of the task information from the video. The final task-token bank is

$$\mathbf{c}_{\text{task}} = [\mathbf{p}; \mathbf{q}_{1:K}] \in \mathbb{R}^{(K+1)\times d}. \tag{4}$$

We verify this design in three ways. Section 4.3 shows that the task tokens $\mathbf{c}_{\text{task}}$ encode task-level semantics. Section 4.5 shows that the learnable queries $\mathbf{q}_{1:K}$ strengthen task encoding and improve prediction compared with a single global summary. Section 4.4 shows that task information improves gaze prediction. Appendix C gives all the details.

### 3.2 Conditional Velocity Network

We now describe how the bottom-up embeddings $\mathbf{c}_{\text{v}}$ and top-down embeddings $\mathbf{c}_{\text{task}}$ are used to help generate gaze trajectories. In particular, we use conditional flow matching [31, 45], where generation is a transport process. A sample starts as a Gaussian noise $\mathbf{x}_0 \in \mathbb{R}^{2T}$, where $T$ is the number of discrete points (frames) on the gaze trajectory we predict (each of the discrete point has two coordinates, hence $2T$ dimensions). It then passes through noisy trajectory states $\mathbf{x}_s$, and eventually reaches a clean trajectory sample $\mathbf{x}_1$. This transport is realized by the following differential equation:

$$\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \quad \frac{d\mathbf{x}_s}{ds} = \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}), \quad \hat{\mathbf{g}} = \mathbf{x}_1 = \int_0^1 \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c})\mathrm{d}s + \mathbf{x}_0 \sim q_\theta(\mathbf{g} \mid \mathcal{V}). \tag{5}$$

Conditioned on $\mathbf{c}_v$ and $\mathbf{c}_{\text{task}}$ (collectively referred as $\mathbf{c}$), the velocity network $\mathbf{v}_\theta$ decides how the current state should move at flow time $s$. $\mathbf{v}_\theta(\cdot)$ takes the current state $\mathbf{x}_s$, the flow time $s$, and the conditions $\mathbf{c}$ as input. $\mathbf{x}_s$ and $s$ are first embedded into $T$ gaze tokens, each of which corresponds to the gaze at frame $t$. The gaze tokens are then updated in each denoising step by four separate blocks, which are visualized in Figure 2(b).

First, **time-aware self-attention** lets gaze tokens exchange information over time. This is where temporal correlation of gaze positions in the trajectory can be captured. We use 1D RoPE to encode relative frame offsets [43]. Second, **per-frame visual cross-attention** lets each gaze token read the spatial visual tokens of the same frame from $\mathbf{c}_v$. This injects bottom-up visual saliency. Third, **task cross-attention** lets all gaze tokens read the task-token bank $\mathbf{c}_{\text{task}}$. This injects top-down, task-specific information. Fourth, a **feed-forward** layer transform each token independently. We implement the velocity network with DiT-style blocks [37]. Appendix C gives the full design details. Finally, a linear output head maps the latent gaze tokens to velocity in the trajectory space $\mathbb{R}^{2T}$.

At inference time, the integral in Eq. 5 is evaluated numerically with a set of Euler steps, each of which evaluates $\mathbf{v}_\theta(\cdot)$ once. Each integral evaluation uses a noise sample $\mathbf{x}_0$ and gives a $\mathbf{x}_1$, representing one sample of the underlying distribution of the gaze trajectory $q_\theta(\mathbf{g} \mid \mathcal{V})$. Repeating this integration with different noise samples gives a sample distribution of the gaze trajectory.

### 3.3 Training & Inference

**Training.** We train GAZEFLOW on 64-frame clips paired with per-frame gaze. For each clip, we sample $s \sim \mathcal{U}[0,1]$ and $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T})$, and form $\mathbf{x}_s = (1-s)\mathbf{x}_0 + s\mathbf{g}$ with a sample $\mathbf{g} \sim p^\star(\cdot \mid \mathcal{V})$ given by the dataset. The CFM loss regresses $\mathbf{v}_\theta$ on the per-sample velocity $\mathbf{g} - \mathbf{x}_0$:

$$\mathcal{L}_{\text{CFM}}(\theta) \;=\; \mathbb{E}_{\mathcal{V},\, s,\, \mathbf{x}_0,\, \mathbf{g}}\Big[\, \big\| \mathbf{v}_\theta(\mathbf{x}_s,\, s,\, \mathbf{c}) - (\mathbf{g} - \mathbf{x}_0) \big\|_2^2 \Big]. \tag{6}$$

See Appendix D.1 and Appendix D.2 for the full training algorithm and engineering details.

**Long-Video Inference.** We process videos longer than 64 frames with overlapping windows. Each window is encoded once for $(\mathbf{c}_v, \mathbf{c}_{\text{task}})$. We integrate Eq. 5 via Euler steps in parallel with multiple noise samples; each endpoint $\mathbf{x}_1$ is one trajectory sample for that window. For adjacent windows, we linearly blend predictions in the overlap region by frame index. This keeps transitions continuous and avoids encoding the full video at once. Appendix D.3 gives the exact windowing and blending rule.

# 4 Experiments

## 4.1 Experimental Setup

**Training and Inference.** We optimize both the video encoder and the velocity network jointly in an end-to-end manner. The encoder is adapted via LoRA, applying rank-16 low-rank [14] updates to its query and value projections, while the velocity network is trained with full parameters. The model is trained by minimizing the conditional flow-matching objective $\mathcal{L}_{\text{CFM}}$ in Eq. 6 using AdamW [33] ($\beta_1$=0.9, $\beta_2$=0.999, weight decay $10^{-4}$), for 80 epochs with a base learning rate of $10^{-4}$ under a cosine schedule decaying to $10^{-6}$. All experiments are run on 4×NVIDIA H100 GPUs in bfloat16 mixed precision. Remaining hyperparameters and configurations are summarized in Appendix E.4. At inference, we draw $K$=50 trajectory samples per clip in parallel as a Monte Carlo of the learned joint distribution. Each sample is generated by integrating the learned velocity field $\mathbf{v}_\theta$ using the explicit Euler scheme over $S$=50 uniform steps, starting from an independent draw of $\mathcal{N}(\mathbf{0}, \mathbf{I})$. It takes $\sim$3.4 s to inference a clip of $T$=64 frames on a single H100 (Appendix E.5).

**Per-Frame Heatmaps.** At each frame, we convert the $K$=50 sampled gaze points into a per-frame heatmap by placing a Gaussian kernel at each point and averaging. This makes GazeFlow directly comparable to heatmap-output baselines under the heatmap-based per-frame metrics defined below.

**Datasets.** We evaluate on two egocentric gaze datasets. EGTEA Gaze+ [30] contains cooking videos recorded at $640 \times 480$ resolution and 24 fps, with eye-tracker gaze annotations, and provides 8,299 training and 2,022 validation clips. Ego4D [10] spans a broader range of daily activities; we use the subset with gaze annotations, recorded at $1088 \times 1080$ resolution and 30 fps, with 12,178 training and 5,202 validation clips. Data preparation details are provided in Appendix E.1.

**Baselines.** We compare against three families of prior methods. For the first two families, we include representative state-of-the-art models.

- *Marginal density estimation.* GLC [24], a state-of-the-art egocentric gaze model that outputs a per-frame fixation heatmap independently for each frame.
- *Autoregressive.* AT [15], the first egocentric gaze model to learn attention transitions using an LSTM conditioned on previous-frame gaze, and EgoM2P [28], a multimodal model pretrained on large-scale egocentric datasets that decodes gaze as a discrete autoregressive token sequence.
- *Simple Priors.* Center Bias always predicts the gaze position as the image center; Random Walk is a Gaussian random walk whose step variance is set to the human frame-to-frame gaze motion calculated from the dataset.

We retrain AT, use the official checkpoint for GLC, and evaluate EgoM2P in two settings: the official zero-shot checkpoint and LoRA fine-tuning. All baselines are evaluated on both datasets, with full configuration details provided in Appendix E.2.

**Metrics.** We use two commonly used metrics: per-frame prediction accuracy and alignment with human temporal dynamics. The first asks whether gaze is predicted at the correct spatial location on each frame. The second asks whether the predicted trajectory has a human-like temporal structure. Full definitions, interpretations, and computation details of all metrics are provided in Appendix E.3.

**(1) Per-Frame Accuracy.** We report the *Area Under the Curve* (AUC) [5, 19], *F1* [24], *Precision* (P) [24], *Recall* (R) [24], and *Average Angular Error* (AAE) [15].

**(2) Temporal Dynamics Alignment.** We compare each predicted trajectory against the human trajectory using the *Average Displacement Error* (ADE) [21] and *Dynamic Time Warping distance* (DTW) [38, 21], each reported as mean and best-of-$K$ over $K$=50 samples. Additionally, we report the mean and median *per-frame displacement* as ratios to the human reference, and the *Jensen–Shannon divergence* (JSD) [7] between the predicted and human displacement histograms.

Table 2: **Per-frame accuracy on EGTEA Gaze+ and Ego4D.** GAZEFLOW achieves the best performance; results on more metrics are in Table 11 in Appendix E.6. Best per column is in bold.

| | | EGTEA Gaze+ | | | | | Ego4D | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Paradigm | Method | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ |
| *Simple Priors* | Center Bias | 0.906 | 0.185 | 0.162 | 0.215 | 16.07 | 0.930 | 0.222 | 0.194 | 0.258 | 13.92 |
| | Random Walk | 0.824 | 0.147 | 0.090 | 0.413 | 16.34 | 0.763 | 0.109 | 0.059 | **0.662** | 14.38 |
| *Marginal density* | GLC [24] | 0.953 | 0.421 | 0.348 | 0.531 | 10.21 | 0.947 | 0.355 | 0.267 | 0.530 | 11.52 |
| *Autoregressive* | EgoM2P (zero-shot) [28] | 0.921 | 0.280 | 0.195 | 0.498 | 13.18 | 0.911 | 0.253 | 0.180 | 0.422 | 13.86 |
| | EgoM2P (LoRA) [28] | 0.921 | 0.293 | 0.205 | 0.515 | 12.81 | 0.930 | 0.297 | 0.217 | 0.470 | 12.69 |
| | AT [15] | 0.956 | 0.419 | 0.339 | 0.547 | 9.88 | 0.954 | 0.346 | 0.268 | 0.487 | 12.01 |
| *Distribution Mapping* | **GAZEFLOW (ours)** | **0.964** | **0.491** | **0.414** | **0.603** | **9.01** | **0.958** | **0.392** | **0.307** | 0.540 | **10.46** |

Table 3: **Alignment with human gaze dynamics on EGTEA Gaze+.** Disp Mean and Disp Med are per-step gaze displacements shown as ratios to the human trajectory (subheader H, measured in pixels; closer to $1\times$ is better). Best per column in bold.

| | | ADE ($\times 10^2$ px)↓ | | DTW ($\times 10^2$ px)↓ | | Disp Mean | Disp Med | JSD↓ |
|---|---|---|---|---|---|---|---|---|
| Paradigm | Method | Mean | Best | Mean | Best | (H: 12.5 px) | (H: 4.0 px) | |
| *Simple Priors* | Center Bias | 1.35 | 1.35 | 1.35 | 1.35 | $0.00\times$ | $0.00\times$ | 0.125 |
| | Random Walk | 2.07 | 1.05 | 1.97 | 0.92 | $2.54\times$ | $7.43\times$ | 0.294 |
| *Marginal density* | GLC [24] | 1.13 | 1.01 | 1.03 | 0.87 | $5.77\times$ | $11.75\times$ | 0.352 |
| *Autoregressive* | EgoM2P (zero-shot) [28] | 1.47 | 0.97 | 1.35 | 0.83 | $2.20\times$ | $3.05\times$ | 0.058 |
| | EgoM2P (LoRA) [28] | 1.50 | 0.96 | 1.37 | 0.81 | $2.69\times$ | $3.03\times$ | 0.058 |
| | AT [15] | 1.32 | 1.12 | 1.22 | 0.99 | $9.70\times$ | $24.55\times$ | 0.476 |
| *Distribution Mapping* | **GAZEFLOW (ours)** | **1.04** | **0.60** | **0.95** | **0.48** | $\mathbf{0.82\times}$ | $\mathbf{1.08\times}$ | **0.002** |

**Protocol.** All per-frame evaluations follow the GLC-format protocol [24]. For each validation clip, we sample uniformly spaced frames, compute metrics against the ground-truth fixation maps, average over frames within the clip, and then average over clips. The main comparisons in Section 4.2 use the LoRA variant of GAZEFLOW on the full validation split. Ablations in Section 4.4 and design analyses in Section 4.5 use the frozen-encoder variant on a fixed validation subset, since repeating $K{=}50$ LoRA sampling per variant is computationally expensive. Details on subset construction, frozen-encoder evaluation, and sample-count selection are provided in Appendix E.3 and Appendix E.8.

### 4.2 GAZEFLOW Outperforms Prior Methods

Joint distribution modeling achieves the best per-frame accuracy across datasets. Table 2 shows that GAZEFLOW outperforms all baselines on both EGTEA Gaze+ and Ego4D. Compared to the strongest baseline, GAZEFLOW improves F1 by $16.6\%$ and $10.4\%$ on the two datasets, reflecting better binarized fixation prediction under the GLC evaluation protocol. It also reduces AAE by $8.8\%$ and $9.2\%$ on the two datasets, corresponding to lower angular error of the predicted fixation with respect to human gaze. Table 11 shows the full breakdown.

GAZEFLOW also matches human gaze dynamics much more closely than prior methods. Table 3 reports the alignment-with-human-dynamics metrics. GAZEFLOW achieves the lowest mean and best-of-$K$ ADE and DTW, indicating smaller path-wise error to the ground-truth trajectory under Euclidean and temporally aligned comparisons, respectively. On best-of-$K$, GAZEFLOW reduces this error by roughly $1.6\times$ compared to the strongest baseline, EgoM2P with LoRA, with ADE decreasing from $0.96$ to $0.60$ and DTW from $0.81$ to $0.48$. Our per-step displacement statistics are also closest to the human reference: the median is $1.08\times$ human and the mean follows the same pattern, whereas GLC and AT are $11.75\times$ and $24.55\times$, respectively. The displacement JSD is $0.002$, the lowest among all methods and closest to the human reference distribution.

### 4.3 Task Tokens Encode Task Semantics

We have hypothesized that task information can be beneficial to gaze prediction, since human gazing is task dependent. This hypothesis motivates us to extract the task-dependent tokens $\mathbf{c}_{\text{task}}$, which are used to condition the flow matching. We now show that $\mathbf{c}_{\text{task}}$ does carry task-level semantics.

Table 4: **Linear probing of the task-token bank** on EGTEA action/verb/noun labels. The ∪ indicates a clip is counted correct if at least one probe is correct.

| | Action (106) | | Verb (19) | | Noun (53) | |
|---|---|---|---|---|---|---|
| Probe input | T1 | T5 | T1 | T5 | T1 | T5 |
| $\mathbf{q}_{1:K}$ alone | 40.5 | 66.6 | 64.1 | 95.9 | 47.9 | 78.7 |
| $\mathbf{p}$ alone | 49.1 | 76.6 | 68.3 | 98.2 | 58.7 | 87.3 |
| $\mathbf{p} \cup \mathbf{q}_{1:K}$ | **54.6** | **80.2** | **76.0** | **98.7** | **64.4** | **89.8** |

Table 5: **Design alternatives to GAZEFLOW.** *w/o Learnable Queries*: drop the $K{=}4$ learnable queries (keep only the global mean-pool token). *Concat*: replace task cross-attention with channel-wise concatenation. *Learnable PE*: replace RoPE with a learnable absolute position embedding.

| Variant | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ |
|---|---|---|---|---|---|
| **Default** | **0.970** | **0.493** | **0.401** | **0.638** | **8.81** |
| w/o Learnable Queries | 0.967 | 0.484 | 0.395 | 0.626 | 8.93 |
| Concat | 0.966 | 0.469 | 0.379 | 0.615 | 9.10 |
| Learnable PE | 0.967 | 0.476 | 0.380 | 0.635 | 8.94 |

Each video clip in EGTEA Gaze+ is annotated at three label hierarchies: 19 verbs (e.g., "cut"), 53 nouns (e.g., "tomato"), and 106 actions formed as verb-noun pairs (e.g., "cut tomato"). We freeze GAZEFLOW, extract its task-token bank for each clip, and train a linear classifier to predict the action, verb, and noun labels. We report Top-$k$ accuracy, which denotes the percentage of validation clips for which the ground-truth class appears among the linear classifier's $k$ highest-scoring predictions. As shown in Table 4, the global token $\mathbf{p}$ is already strongly predictive, reaching $49.1\%$, $68.3\%$, and $58.7\%$ Top-1 accuracy on action, verb, and noun, respectively. The learnable task queries $\mathbf{q}_{1:K}$ are weaker in isolation, but provide complementary information: combining $\mathbf{p}$ and $\mathbf{q}_{1:K}$ improves Top-1 by $+5.5$, $+7.7$, and $+5.7$ points. These results show that task-relevant semantics emerge in the task-token bank despite the absence of explicit supervision, consistent with the design in Eq. 4.

### 4.4 Ablation Studies

We isolate the contribution of each architectural component by removing one component at a time, as shown in Table 6. The four ablation rows test the four architectural choices made in Section 3: joint-distribution modeling via flow matching, the bottom-up pathway, the top-down pathway, and 1D temporal RoPE for time-aware self-attention (SA).

**Flow Matching vs. Regression.** Replacing the flow-matching head with a deterministic regression head over the same encoder and conditioning inputs reduces AUC from $0.970$ to $0.903$ and inflates the displacement JSD against human gaze by roughly $66\times$. This is the most direct evidence for the central claim of Section 3.2: gaze should be modeled as a joint distribution over the full trajectory. A regression head, which neither models a distribution nor couples frames, predicts each gaze point in isolation and therefore fails to capture the temporal coupling between frames.

**Dual-Path Conditioning.** Dropping the per-frame visual cross-attention sub-layer of the velocity network, which carries the bottom-up condition $\mathbf{c}_v$ from Section 3.1, lowers F1 from $0.493$ to $0.431$. Dropping the task cross-attention sub-layer, which carries the top-down condition $\mathbf{c}_{\text{task}}$, lowers F1 to $0.476$. Both pathways are individually necessary, supporting the dual-pathway design in Section 3.1.

**Temporal RoPE.** Removing the relative-time RoPE from the time-aware self-attention sub-layer of the velocity network has little effect on per-frame accuracy but weakens the displacement JSD by roughly $2.5\times$ and worsens the median displacement to $1.4\times$ of that of human. This confirms that the relative-offset encoding is what enables time-aware self-attention to capture inter-frame temporal coupling, as claimed in Section 3.2.

### 4.5 Design Choice Analysis

**Temporal Gaze Correlation.** Time-aware gaze-token self-attention is where our denoise block models the temporal coupling needed for joint-distribution modeling, and its temporal positional encoding decides how each gaze token reasons about its position relative to other frames. We compare two choices: *(i)* RoPE (default, details in Appendix C.4), which injects each pairwise frame offset directly into the attention dot product, and *(ii)* a learnable absolute position embedding, which assigns a learned vector to each frame index. Replacing RoPE with the absolute embedding lowers F1 from $0.493$ to $0.476$ (Table 5). This is because gaze dynamics are properties of how far apart two frames are, not of where a frame sits inside an arbitrary clipped window.

Table 6: **Ablation study. Full** is the default GAZEFLOW; each remaining row ablates one architectural component while keeping all others identical, removing either the bottom-up visual feature ($\mathbf{c}_v$, frame-local visual cross-attention), the top-down task guidance ($\mathbf{c}_{\text{task}}$, task cross-attention), the relative-time RoPE in time-aware self-attention, or the joint-distribution flow-matching head (replaced with a deterministic regression head over the same conditioning). Per-column best in **bold**.

| Variant | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ | ADE↓ | DTW↓ | Disp Mean (H: 12.5 px) | Disp Med (H: 4.0 px) | JSD↓ |
|---|---|---|---|---|---|---|---|---|---|---|
| **Full** | **0.970** | **0.493** | **0.401** | **0.638** | **8.81** | **0.57** | **0.45** | **0.91×** | **1.08×** | **0.0011** |
| - Bottom-up Visual Feature | 0.961 | 0.431 | 0.333 | 0.613 | 9.22 | 0.65 | 0.53 | 0.89× | 1.17× | 0.0012 |
| - Top-down Task Guidance | 0.965 | 0.476 | 0.380 | 0.638 | 9.14 | 0.58 | 0.46 | 0.92× | 1.09× | 0.0013 |
| - Temporal RoPE | 0.968 | 0.477 | 0.388 | 0.620 | 8.94 | 0.59 | 0.46 | 1.21× | 1.40× | 0.0027 |
| - Joint Distribution Modeling | 0.903 | 0.451 | 0.394 | 0.526 | 9.12 | 0.84 | 0.79 | 0.71× | 2.01× | 0.0729 |

**Top-Down Task Guidance.** Top-down task information from the task-token bank is injected into gaze tokens via a per-block task cross-attention sub-layer. We compare two forms of task guidance: *(i)* cross-attention (default; Appendix C.6), where each gaze token queries the bank at every denoiser block, and *(ii)* channel-wise concatenation of the bank to the gaze-token input. Concatenation degrades all metrics, including F1 ($0.493 \rightarrow 0.469$) and AAE ($8.81 \rightarrow 9.10$); see Table 5 for details.

Removing the $K{=}4$ learnable queries, leaving only the global mean-pool token, also reduces F1 to $0.484$ (Table 5), confirming that the queries enable task-aware selective extraction beyond the low-bandwidth global token alone.

**Task-Token Bank Bandwidth.** We sweep $K \in \{1, 2, 4, 8, 16\}$, the size of the learnable task queries, and the results are shown in Table 12 in Appendix E.7. The performance peaks at $K = 4$, indicating that too few queries cannot represent the diversity of task-relevant factors, while too many dilute the per-query specialization that cross-attention relies on.

### 4.6 Interpretations

**Interpreting Self-Attention.** Self-attention in our velocity network decides, when predicting gaze at frame $i$, how strongly to draw on gaze at each other frame $j$. For each head we measure how attention mass distributes over $j$ and find two regimes: *look-back* heads concentrate on past frames and *look-ahead* heads concentrate on future frames. This pattern parallels two well-known modes of human gaze in egocentric tasks — anticipating upcoming events [35, 27] and lagging ongoing ones to verify completion [3, 11]. A third *split-attention* category and full visualizations are in Appendix A.1.

**Interpreting Bottom-Up Saliency.** The visual cross-attention sub-layer provides bottom-up guidance that decides, when predicting gaze at frame $t$, which spatial regions of the video to focus on. Using off-the-shelf hand and active-object segmentation algorithm [32], we measure how much attention mass each attention head places on the hand, the active object, and the rest of the scene. We find that the heads cluster into hand-focused, object-focused, and coupled groups, and the cluster also varies with the action being performed. This finding is consistent with the manipulation literature, in which gaze concentrates on the hands and the objects they act on [26, 11]. Details are in Appendix A.2.

### 4.7 Limitations

GAZEFLOW's failures concentrate on clips with saccades whose endpoint lies outside the camera's field of view: the saccade target was never captured by the recorded video on which the model conditions, so there is no visual evidence for us to make the prediction (Appendix A.3).

GAZEFLOW models the conditional distribution of gaze trajectories given a fixed input video, so the head movement that produced that video is part of the conditioning. The gaze distribution we recover is therefore of the residual eye movement, not the full distribution that would arise across observers freely interacting with the same scene, where head movement itself varies across trials. We leave to future work to jointly model head and eye movements.

Additionally, GAZEFLOW's prediction is also conditioned upon the *entire* video sequence, which limits its applicability to scenarios that requires real-time gaze prediction from egocentric videos, where one have access only to past and the current frames.

## 5 Conclusion

GAZEFLOW is designed to account for the three factors that influence the stochasticity of human gaze: structured temporal dynamics, top-down task influence, and bottom-up scene saliency. GAZEFLOW sets a new state of the art on per-frame prediction accuracy while simultaneously achieving the closest alignment with human gaze dynamics. We hope GAZEFLOW can broaden the use of gaze in applications that have traditionally required dedicated eye-tracking hardware.

## References


[1] Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

[2] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[3] Dana H. Ballard, Mary M. Hayhoe, and Jeff B. Pelz. Memory representations in natural tasks. *Journal of Cognitive Neuroscience*, 7:66–80, 1995.

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[5] Zoya Bylinskii, Tilke Judd, Aude Oliva, Antonio Torralba, and Frédo Durand. What do different evaluation metrics tell us about saliency models? *IEEE transactions on pattern analysis and machine intelligence*, 41(3):740–757, 2019.

[6] Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

[7] Bent Fuglede and Flemming Topsøe. Jensen-shannon divergence and hilbert space embedding. *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, page 31, 2004.

[8] Wilson S Geisler and Jeffrey S Perry. Real-time foveated multiresolution system for low-bandwidth video communication. In *Human vision and electronic imaging III*, volume 3299, pages 294–305. SPIE, 1998.

[9] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[10] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18995–19012, 2022.

[11] Mary Hayhoe and Dana Ballard. Eye movements in natural behavior. *Trends in cognitive sciences*, 9(4):188–194, 2005.

[12] John M Henderson. Gaze control as prediction. *Trends in cognitive sciences*, 21(1):15–23, 2017.

[13] Jonathan Ho, Ajay Jain, and P. Abbeel. Denoising diffusion probabilistic models. *ArXiv*, abs/2006.11239, 2020.

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. 2022.

[15] Yifei Huang, Minjie Cai, Zhenqiang Li, and Yoichi Sato. Predicting gaze in egocentric video by learning task-dependent attention transition. In *Proceedings of the European conference on computer vision (ECCV)*, pages 754–769, 2018.

[16] Laurent Itti. Automatic foveation for video compression using a neurobiological model of visual attention. *IEEE transactions on image processing*, 13(10):1304–1318, 2004.

[17] Laurent Itti and Christof Koch. A saliency-based search mechanism for overt and covert shifts of visual attention. *Vision research*, 40(10-12):1489–1506, 2000.

[18] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11): 1254–1259, 1998.

[19] Tilke Judd, Frédo Durand, and Antonio Torralba. A benchmark of computational models of saliency to predict human fixations. 2012.

[20] Anton S Kaplanyan, Anton Sochenov, Thomas Leimkühler, Mikhail Okunev, Todd Goodall, and Gizem Rufo. Deepfovea: Neural reconstruction for foveated rendering and video compression using learned statistics of natural videos. *ACM Transactions on Graphics (TOG)*, 38(6):1–13, 2019.

[21] Ozgur Kara, Harris Nisar, and James Matthew Rehg. Diffeye: Diffusion-based continuous eye-tracking data generation conditioned on natural images. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[22] Heecheol Kim, Yoshiyuki Ohmura, and Yasuo Kuniyoshi. Gaze-based dual resolution deep imitation learning for high-precision dexterous robot manipulation. *IEEE Robotics and Automation Letters*, 6:1630–1637, 2021.

[23] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[24] Bolin Lai, Miao Liu, Fiona Ryan, and James M Rehg. In the eye of transformer: Global–local correlation for egocentric gaze estimation and beyond. *International Journal of Computer Vision*, 132(3):854–871, 2024.

[25] Bolin Lai, Fiona Ryan, Wenqi Jia, Miao Liu, and James M Rehg. Listen to look into the future: Audio-visual egocentric gaze anticipation. In *European Conference on Computer Vision*, pages 192–210. Springer, 2024.

[26] Michael Land, Neil Mennie, and Jennifer Rusted. The roles of vision and eye movements in the control of activities of daily living. *Perception*, 28(11):1311–1328, 1999.

[27] Michael F Land and Mary Hayhoe. In what ways do eye movements contribute to everyday activities? *Vision research*, 41(25-26):3559–3565, 2001.

[28] Gen Li, Yutong Chen, Yiqian Wu, Kaifeng Zhao, Marc Pollefeys, and Siyu Tang. Egom2p: Egocentric multimodal multitask pretraining. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10830–10843, 2025.

[29] Yin Li, Alireza Fathi, and James M. Rehg. Learning to predict gaze in egocentric video. *2013 IEEE International Conference on Computer Vision*, pages 3216–3223, 2013.

[30] Yin Li, Miao Liu, and James M Rehg. In the eye of beholder: Joint learning of gaze and actions in first person video. In *Proceedings of the European conference on computer vision (ECCV)*, pages 619–635, 2018.

[31] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[32] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chun yue Li, Jianwei Yang, Hang Su, Jun-Juan Zhu, and Lei Zhang. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. pages 38–55, 2024.

[33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[34] Amrita Mazumdar, Brandon Haynes, Magda Balazinska, Luis Ceze, Alvin Cheung, and Mark Oskin. Perceptual compression for video storage and processing systems. In *Proceedings of the ACM symposium on cloud computing*, pages 179–192, 2019.

[35] Neil Mennie, Mary M. Hayhoe, and Brian T. Sullivan. Look-ahead fixations: anticipatory eye movements in natural tasks. *Experimental Brain Research*, 179:427–442, 2007.

[36] Andrea Palazzi, Davide Abati, Francesco Solera, Rita Cucchiara, et al. Predicting the driver's focus of attention: The dr (eye) ve project. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1720–1733, 2018.

[37] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[38] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268, 2009.

[39] Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM journal on control and optimization*, 30(4):838–855, 1992.

[40] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya K. Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloé Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao Wu, Ross B. Girshick, Piotr Doll'ar, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *ArXiv*, abs/2408.00714, 2024.

[41] Constantin A Rothkopf, Dana H Ballard, and Mary M Hayhoe. Task and context determine where you look. *Journal of vision*, 7(14):16–16, 2007.

[42] Michele Rucci, Ehud Ahissar, and David Burr. Temporal coding of visual space. *Trends in cognitive sciences*, 22(10):883–895, 2018.

[43] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[44] Benjamin W Tatler, Mary M Hayhoe, Michael F Land, and Dana H Ballard. Eye guidance in natural vision: Reinterpreting salience. *Journal of vision*, 11(5):5–5, 2011.

[45] Alexander Tong, Kilian FATRAS, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *Transactions on Machine Learning Research*.

[46] Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[48] Alfred L Yarbus. Eye movements and vision. *Eye movements and vision.*, page 171, 1967.

[49] Mengmi Zhang, Keng Teck Ma, Joo Hwee Lim, Qi Zhao, and Jiashi Feng. Deep future gaze: Gaze anticipation on egocentric videos using adversarial networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3539–3548, 2017.

# A Additional Visualization Results

This appendix collects visual analyses that supplement the main text.

## A.1 Per-Head Patterns of the Time-aware Self-Attention

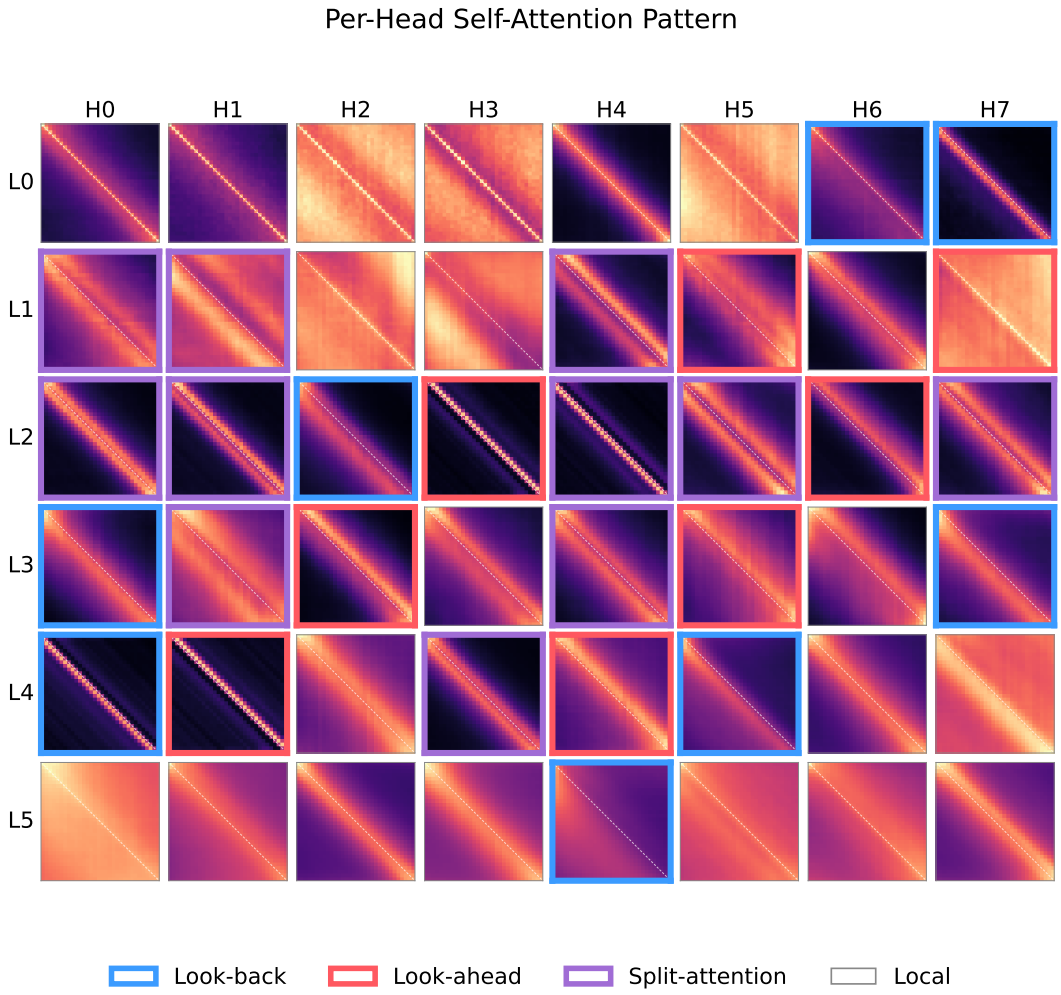


Figure. 3: **Per-head patterns of the time-aware self-attention aggregated across the EGTEA Gaze+ validation set.** Rows index denoiser layers, columns index attention heads. Each cell shows the canonical attention pattern (query frame on the vertical axis, source frame on the horizontal, both normalized to clip-relative position); brighter pixels correspond to higher attention weight after row-wise softmax. The dashed diagonal marks *source* = *query*, separating two halves with opposite temporal meaning: mass in the *lower triangle* (*source* $<$ *query*) is attention to *past* frames (look-back), mass in the *upper triangle* (*source* $>$ *query*) is attention to *future* frames (look-ahead), and mass on both sides marks split-attention heads with peaks straddling the current frame. Cell-frame color encodes this role: look-back (blue), look-ahead (red), split-attention (purple), local (grey).

The two figures above underpin the head-level claim in Section 4.6. We forward EGTEA Gaze+ validation clips through the trained main model at the midpoint of denoising and capture the gaze self-attention matrix of every (layer, head). Because clip length varies, each matrix is resized to a common grid where both axes are normalized to clip-relative position; the per-head pattern shown is the average across clips. We classify each head from its 1-D attention-vs-offset curve into one of four temporal roles: *look-back*, *look-ahead*, *split-attention* (an inverted-M shape with peaks on both sides of the current frame), or *local*; the same colour scheme is used in both figures and the

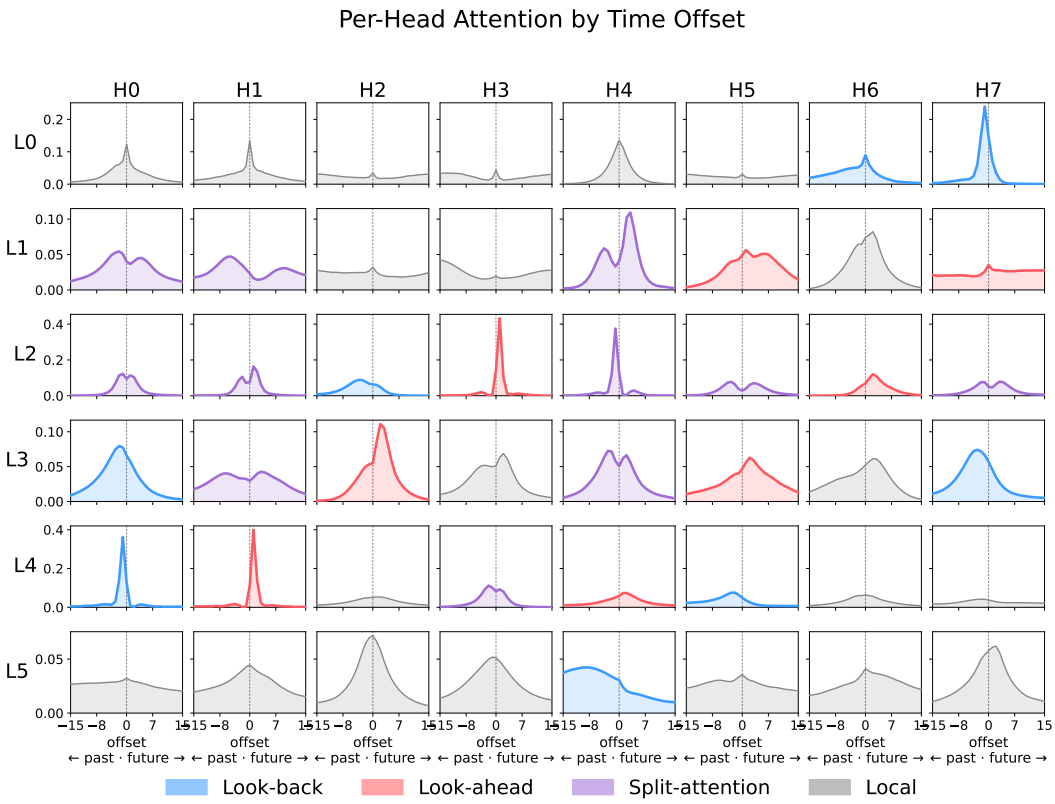


Figure. 4: **Per-head time-aware self-attention as a function of temporal offset.** Same heads as Fig. 3, summarized as a 1-D curve by averaging the attention pattern along each diagonal. The horizontal axis is the offset between source and query frame (negative = past, positive = future); the vertical dashed line marks zero offset. Look-ahead heads peak right of zero, look-back heads peak left, and split heads exhibit two peaks straddling zero with a clear valley at the current frame.

legend. Figure 3 presents the full 2-D pattern per head, which exposes the spatial structure of each role: causal heads light up the lower triangle, anticipatory heads light up the upper triangle, and split heads light up both triangles. Figure 4 collapses the same data to a 1-D temporal-offset curve, where the three role types are immediately distinguishable from peak position.

### A.2 Per-Head Semantic Attribution

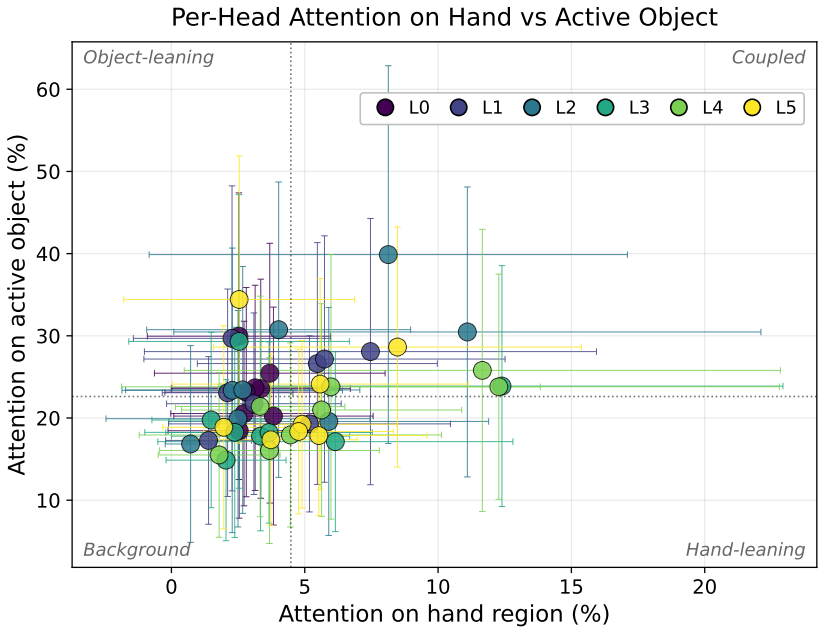


Figure. 5: **Hand vs. active-object attention mass per (layer, head)** on the EGTEA Gaze+ validation set. Each marker is one of the 48 visual-cross-attention heads, plotted by mean attention mass on the hand (x-axis) and active-object (y-axis) regions; markers are coloured by layer, error bars are clip-wise standard deviation. Dotted lines mark the global means (hand $= 4.5\%$, object $= 22.6\%$). Heads in the lower-right quadrant focus on the hand, those in the upper-left on the active object, and those in the upper-right couple both.

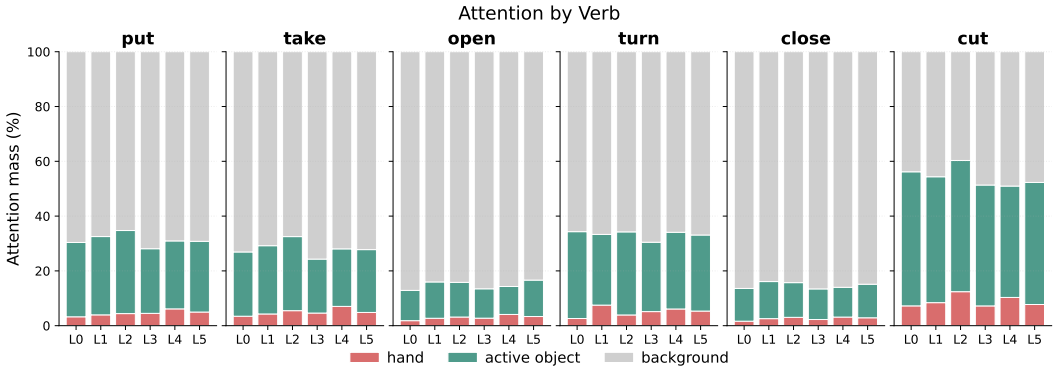


Figure. 6: **Layer-wise hand / active-object / background attention mass conditioned on verb.** For each of the six most frequent EGTEA verbs we average the per-region attention mass across heads at each layer. Manipulation verbs ("cut", "put", "take", "turn") keep 25–50% of the mass on the active object at every layer, while "open" and "close" route most mass to the background regardless of layer.

The two figures above underpin the head-level claim in the Section 4.6. For each clip in the EGTEA Gaze+ validation set we extract per-frame Hand and Active-Object masks via GroundingDINO [32] and SAM 2 [40], downsampled to the $16\times16$ V-JEPA 2 token grid. We then run a forward pass of the trained main model and capture the per-(layer, head) attention weights of the per-frame visual cross-attention sub-layer (Section 3.2) over the spatial tokens. For each clip, layer, and head we sum the attention mass inside the hand mask, inside the active-object mask, and outside both (background); the three values are normalized to 1 per (clip, layer, head) and then averaged across clips. Figure 5

plots each head as one point in the (hand, object) attention plane: heads in the lower-right quadrant concentrate on the hand, those in the upper-left on the active object, and those in the upper-right couple both, all relative to the global means (hand $= 4.5\%$, object $= 22.6\%$). Figure 6 groups clips by the first verb of their EGTEA action label, and the per-verb pattern reflects what each action requires from vision. "cut", "put", "take", and "turn" all act on an already-engaged object, so object-local evidence suffices and the heads keep 25–50% of the mass on the active object across all six layers. "open" and "close" must first locate a fixture (a drawer, lid, or container) inside the workspace, which demands workspace-wide visual support, and the heads accordingly route most of the mass to the background regardless of layer.

### A.3 Failure Case Analysis

GAZEFLOW's failure cases concentrate on clips containing saccades whose endpoint lies outside the camera's field of view (Figure 7). A large saccade traverses tens of visual degrees in 30–80 ms [42], so at our 24–30 fps recording rate the eye can leave the captured frame within a single sample. Once the eye target is outside the camera field of view (FOV), the model has no visual evidence on which to ground a prediction: GAZEFLOW is conditioned solely on the recorded video, and the saccade endpoint by definition was never captured by it. The predicted trajectory cloud therefore stays anchored to features that remain visible, while the eye-tracker continues to report an angle whose corresponding pixel is clamped to the image boundary as an in-flight transit. Closing this gap would require a wider-FOV camera that captures the eventual saccade target, or a saccade-aware model that recognizes when the gaze has left the conditioning support, both of which we leave to future work.


Figure. 7: **Failure cases.** Three rows from two validation clips, each row showing 8 evenly-spaced frames along time. Cyan dots are the $K{=}50$ GAZEFLOW predictions at that frame; the filled red circle is the recorded ground-truth gaze when it lies within the frame, and the hollow red ring with a yellow halo flags frames where the recording is clamped to the image boundary (off-frame). When the wearer saccades to a target outside the camera's view, the predicted cloud stays anchored to the visible scene while the ground-truth marker drifts to the boundary.

# B Conditional Flow Matching: Derivation

This appendix shows that the conditional flow matching loss $\mathcal{L}_{\mathrm{CFM}}$ used in Section 3.3 (Eq. 6) is a sample-based $L^2$ regression on a velocity field whose global minimum yields, at the population level, $q_\theta = p^\star$. Section B.1 sets up the notation used throughout this appendix; Section B.2 gives the derivation that minimizing $\mathcal{L}_{\mathrm{CFM}}$ recovers $p^\star$; and Section B.3 describes the Euler ODE sampler used at inference.

## B.1 Notation

Table 7: Notation used throughout this appendix.

| Symbol | Meaning |
|---|---|
| $T$ | Number of frames in the window ($T$=64) |
| $\mathcal{V}$ | Egocentric video clip / conditioning input |
| $\mathbf{g} \in \mathbb{R}^{2T}$ | Ground-truth gaze trajectory (sample from $p^\star$) |
| $p^\star(\cdot \mid \mathcal{V})$ | True conditional gaze trajectory distribution |
| $q_\theta(\cdot \mid \mathcal{V})$ | Learned model distribution |
| $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T})$ | Source noise sample |
| $\mathbf{x}_s$ | Interpolated state at flow time $s \in [0,1]$ |
| $s \sim U[0,1]$ | Flow time |
| $\mathbf{c} = E_\psi(\mathcal{V})$ | Video conditioning passed to $\mathbf{v}_\theta$ |
| $\mathbf{v}_\theta(\mathbf{x}, s, \mathbf{c})$ | Learned velocity field (denoiser output) |
| $\mathbf{u}_s^\star(\mathbf{x} \mid \mathcal{V})$ | Marginal target velocity field |
| $p_s(\cdot \mid \mathcal{V})$ | Marginal probability path at time $s$ |
| $\mathbf{m} \in \{0,1\}^T$ | Per-frame validity mask |
| $S$ | Number of Euler ODE steps ($S$=50) |
| $K$ | Number of i.i.d. samples drawn per clip |

## B.2 Why $\mathcal{L}_{\mathrm{CFM}}$ recovers $p^\star$

**Velocity matching equates $q_\theta$ with $p^\star$.** We parametrize $q_\theta$ as the law of $\mathbf{x}_1$ given by the ODE

$$\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T}), \qquad \frac{d\mathbf{x}_s}{ds} \;=\; \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}), \qquad \mathbf{x}_1 \sim q_\theta(\cdot \mid \mathcal{V}), \tag{7}$$

where $\mathbf{v}_\theta(\mathbf{x}, s, \mathbf{c})$ is the learned velocity field with conditioning $\mathbf{c} = E_\psi(\mathcal{V})$. By mass conservation under the flow generated by $\mathbf{v}_\theta$, the time-marginal density $q_s(\mathbf{x} \mid \mathcal{V})$ obeys the continuity equation

$$\partial_s\, q_s(\mathbf{x} \mid \mathcal{V}) \;+\; \nabla_{\mathbf{x}} \cdot \big(q_s(\mathbf{x} \mid \mathcal{V})\, \mathbf{v}_\theta(\mathbf{x}, s, \mathbf{c})\big) \;=\; 0, \qquad q_0 \;=\; \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T}). \tag{8}$$

The path $\{q_s\}_{s\in[0,1]}$, and in particular $q_\theta = q_1$, is therefore fully determined by $\mathbf{v}_\theta$. On the data side, for each pair $(\mathbf{x}_0, \mathbf{g}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T}) \otimes p^\star(\cdot \mid \mathcal{V})$, define the linear interpolant

$$\mathbf{x}_s \;:=\; (1-s)\,\mathbf{x}_0 + s\,\mathbf{g}, \qquad s \in [0,1]. \tag{9}$$

At $s = 0$, $\mathbf{x}_s = \mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T})$, so $p_0 = \mathcal{N}(\mathbf{0}, \mathbf{I}_{2T})$; at $s = 1$, $\mathbf{x}_s = \mathbf{g} \sim p^\star$, so $p_1 = p^\star(\cdot \mid \mathcal{V})$. The conditional density on the line segment is a Dirac

$$p_s(\mathbf{x} \mid \mathbf{x}_0, \mathbf{g}) \;=\; \delta\big(\mathbf{x} - (1-s)\,\mathbf{x}_0 - s\,\mathbf{g}\big), \tag{10}$$

and the marginal density factors over the noise-data coupling:

$$p_s(\mathbf{x} \mid \mathcal{V}) \;=\; \int p_s(\mathbf{x} \mid \mathbf{x}_0, \mathbf{g})\, p_0(\mathbf{x}_0)\, p^\star(\mathbf{g} \mid \mathcal{V})\, d\mathbf{x}_0\, d\mathbf{g}. \tag{11}$$

Differentiating Eq. 9 in $s$ gives the per-sample velocity

$$\frac{d\mathbf{x}_s}{ds} \;=\; \mathbf{g} - \mathbf{x}_0, \tag{12}$$

and the marginal velocity field $\mathbf{u}_s^\star$ that generates $p_s$ via $\partial_s p_s + \nabla_{\mathbf{x}} \cdot (p_s \mathbf{u}_s^\star) = 0$ is the density-weighted average of conditional velocities [31]:

$$\mathbf{u}_s^\star(\mathbf{x} \mid \mathcal{V}) \;=\; \int \frac{p_s(\mathbf{x} \mid \mathbf{x}_0, \mathbf{g})}{p_s(\mathbf{x} \mid \mathcal{V})}\, (\mathbf{g} - \mathbf{x}_0)\, p_0(\mathbf{x}_0)\, p^\star(\mathbf{g} \mid \mathcal{V})\, d\mathbf{x}_0\, d\mathbf{g} \;=\; \mathbb{E}\big[\mathbf{g} - \mathbf{x}_0 \,\big|\, \mathbf{x}_s = \mathbf{x},\, \mathcal{V}\big]. \tag{13}$$

The model density $q_s$ (Eq. 8) and the target density $p_s$ both satisfy a continuity equation with the same initial condition $\mathcal{N}(\mathbf{0}, \mathbf{I}_{2T})$. By PDE uniqueness, the velocity field uniquely determines the density path:

$$\mathbf{v}_\theta(\mathbf{x}, s, \mathbf{c}) \;=\; \mathbf{u}_s^\star(\mathbf{x} \mid \mathcal{V}) \;\; \forall s, \mathbf{x} \;\;\Longrightarrow\;\; q_s = p_s \;\; \forall s \;\;\Longrightarrow\;\; q_\theta = p_1 = p^\star.$$

Hence $q_\theta = p^\star$ at the population level iff $\mathbf{v}_\theta \equiv \mathbf{u}_s^\star$.

**Sample-based regression preserves the global minimum.** Velocity matching has reduced distribution matching to pointwise matching of two vector fields, for which the natural loss is squared $L^2$ error. We thus define the *marginal flow matching loss* as the $L^2$ regression of $\mathbf{v}_\theta$ onto $\mathbf{u}_s^\star$:

$$\mathcal{L}_{\mathrm{FM}}(\theta) \;=\; \mathbb{E}_{\mathcal{V}}\, \mathbb{E}_{s\sim U[0,1]}\, \mathbb{E}_{\mathbf{x}_s\sim p_s(\cdot|\mathcal{V})}\Big[\, \big\|\, \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}) - \mathbf{u}_s^\star(\mathbf{x}_s \mid \mathcal{V})\, \big\|_2^2 \,\Big]. \tag{14}$$

At its global minimum, $\mathcal{L}_{\mathrm{FM}}(\theta^\star) = 0 \Leftrightarrow \mathbf{v}_{\theta^\star} \equiv \mathbf{u}_s^\star$ a.e., which by the previous paragraph implies $q_{\theta^\star} = p^\star$. However, $\mathcal{L}_{\mathrm{FM}}$ is intractable: evaluating $\mathbf{u}_s^\star$ at any $\mathbf{x}_s$ requires the conditional expectation in Eq. 13, which has no closed form. We therefore replace the marginal target with the per-sample velocity $\mathbf{g} - \mathbf{x}_0$ from Eq. 12:

$$\mathcal{L}_{\mathrm{CFM}}(\theta) \;=\; \mathbb{E}_{\mathcal{V}}\, \mathbb{E}_{s,\,\mathbf{x}_0,\,\mathbf{g}}\Big[\, \big\|\, \mathbf{v}_\theta\big((1-s)\,\mathbf{x}_0 + s\,\mathbf{g},\; s,\; \mathbf{c}\big) - (\mathbf{g} - \mathbf{x}_0)\, \big\|_2^2 \,\Big], \tag{15}$$

which uses only quantities sampled from data and matches main-text Eq. 6. The network sees only $(\mathbf{x}_s, s, \mathbf{c})$, not $(\mathbf{x}_0, \mathbf{g})$, so by the tower property

$$\mathcal{L}_{\mathrm{CFM}}(\theta) \;=\; \mathbb{E}_{\mathcal{V}}\, \mathbb{E}_{s,\,\mathbf{x}_s}\, \mathbb{E}_{(\mathbf{x}_0,\mathbf{g})|\mathbf{x}_s,\,\mathcal{V}}\Big[\, \big\|\, \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}) - (\mathbf{g} - \mathbf{x}_0)\, \big\|_2^2 \,\Big],$$

and the inner $L^2$ regression at fixed $(\mathbf{x}_s, s, \mathbf{c})$ is minimized by the conditional expectation

$$\mathbf{v}_\theta^\star(\mathbf{x}_s, s, \mathbf{c}) \;=\; \mathbb{E}\big[\mathbf{g} - \mathbf{x}_0 \,\big|\, \mathbf{x}_s,\, \mathcal{V}\big] \;=\; \mathbf{u}_s^\star(\mathbf{x}_s \mid \mathcal{V}), \tag{16}$$

where the second equality is Eq. 13. Hence $\mathcal{L}_{\mathrm{CFM}}$ has the same global minimum as $\mathcal{L}_{\mathrm{FM}}$; in fact $\nabla_\theta \mathcal{L}_{\mathrm{CFM}} = \nabla_\theta \mathcal{L}_{\mathrm{FM}}$ [31]. Optimizing the tractable sample-based regression $\mathcal{L}_{\mathrm{CFM}}$ thus drives $\mathbf{v}_\theta \to \mathbf{u}_s^\star$ and $q_\theta \to p^\star$, recovering the population gaze distribution.

**Masked variant for invalid frames.** EGTEA Gaze+ contains blink intervals where the fixation label is undefined. We use a per-frame validity mask $\mathbf{m} \in \{0,1\}^T$ (broadcast along the coordinate axis to $\mathbb{R}^{T\times 2}$) and replace $\mathcal{L}_{\mathrm{CFM}}$ with

$$\mathcal{L}_{\mathrm{masked}}(\theta) \;=\; \mathbb{E}\left[\frac{\big\|\, \mathbf{m} \odot \big(\mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}) - (\mathbf{g} - \mathbf{x}_0)\big)\big\|_2^2}{\max(\|\mathbf{m}\|_1, 1)}\right]. \tag{17}$$

Since $\mathbf{m}$ is independent of $\theta$ and zeroes out invalid coordinates, the inner $L^2$ regression at every valid coordinate is unchanged: the optimal $\mathbf{v}_\theta^\star$ on valid coordinates remains $\mathbf{u}_s^\star$, and invalid coordinates impose no constraint on $\mathbf{v}_\theta$.

### B.3 Inference: Euler ODE sampling

We integrate Eq. 7 with $S$ uniform Euler steps:

$$\mathbf{x}_{k+1} \;=\; \mathbf{x}_k + \tfrac{1}{S}\, \mathbf{v}_\theta\big(\mathbf{x}_k,\, k/S,\, \mathbf{c}\big), \qquad k = 0, \ldots, S{-}1. \tag{18}$$

We use $S = 50$. Drawing $K$ independent $\mathbf{x}_0^{(k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and running this in parallel yields $K$ i.i.d. samples from $q_\theta(\cdot \mid \mathcal{V})$.

## C Velocity Network Architecture

This section specifies every component of the velocity network $\mathbf{v}_\theta$. Section C.1 gives a one-page forward-pass overview with tensor shapes. Section C.2–Section C.4 describe the non-attention components (coordinate / timestep / self-conditioning fusion, encoder, RoPE). Section C.5 specifies the Task Injection module that produces the clip-level task tokens. Section C.6 composes everything by specifying each of the four sub-layers of a single denoiser block, including explicit per-head factorization of the attentions; Sub-layer 3 consumes the task tokens defined in the previous subsection.

### C.1 Overview and Forward-Pass Tensor Trace

In one forward pass, the network maps a noisy trajectory state $\mathbf{x}_t \in [-1,1]^{T\times 2}$, flow time $t \in [0,1]$, and video conditioning $\mathcal{V}$ to a velocity estimate $\mathbf{v}_\theta \in \mathbb{R}^{T\times 2}$. Table 8 gives the tensor-shape trace.

Table 8: Forward-pass tensor trace of the velocity network. $T$=64, $N$=256, $D$=1024, $d$=256, $K$=4, $H$=8, $L$=6.

| # | Input | Output | Module |
|---|---|---|---|
| 1 | $(T,2)$ noisy + $(T,2)$ self-cond | $(T,d)$ | Coordinate encoder (MLP) |
| 2 | $t \in [0,1]$ | $(d)$, broadcast to $(T,d)$ | Sinusoidal + MLP; additive |
| 3 | $(T,N,D)$ raw encoder features | $(T,N,d)$ | V-JEPA 2 encoder + linear |
| 4 | $(T,N,d)$ | $((K{+}1),d)$ | Task Bank |
| 5 | $(T,d)$, $(T,N,d)$, $((K{+}1),d)$ | $(T,d)$ | $L$ DenoiserBlock layers |
| 6 | $(T,d)$ | $(T,2)$ | LayerNorm + Linear output head |

Steps 1–2 produce the initial gaze tokens, step 3 produces the visual context, step 4 produces the clip-level task tokens, and step 5 iteratively refines the gaze tokens by reading from both contexts. Steps 3–4 are *per clip*: they are computed once and reused across all $L$ layers of step 5 and (at inference) across all $S$ Euler integrator steps. Only step 5 re-executes per Euler step.

### C.2 Input Pipeline

The three pieces consumed at the input side—noisy gaze, flow time, and self-conditioning estimate—are fused into the initial gaze token sequence $\mathbf{h}^{(0)} \in \mathbb{R}^{T\times d}$ that feeds the first DenoiserBlock.

**Coordinate encoder.** The noisy gaze $\mathbf{x}_t \in \mathbb{R}^{T\times 2}$ and the previous-step self-conditioning estimate $\hat{\mathbf{x}}_1^{\text{prev}} \in \mathbb{R}^{T\times 2}$ are concatenated along the feature dimension and passed through a two-layer MLP with SiLU activation:

$$\mathbf{c}_t = [\mathbf{x}_t\,;\,\hat{\mathbf{x}}_1^{\text{prev}}] \in \mathbb{R}^{T\times 4}, \tag{19}$$

$$\mathbf{h}^{\text{coord}} = \mathbf{W}_2\,\sigma_{\text{SiLU}}(\mathbf{W}_1\,\mathbf{c}_t + \mathbf{b}_1) + \mathbf{b}_2 \in \mathbb{R}^{T\times d}, \tag{20}$$

with $\mathbf{W}_1 \in \mathbb{R}^{d\times 4}$ and $\mathbf{W}_2 \in \mathbb{R}^{d\times d}$. The same MLP is used throughout training whether or not self-conditioning is active: when the Bernoulli coin-flip (Section D.2) sets $\hat{\mathbf{x}}_1^{\text{prev}} = \mathbf{0}$, the last two coordinates of $\mathbf{c}_t$ are zero and the linear layer behaves as if they were not there.

**Timestep embedding.** The continuous flow time $t \in [0,1]$ is first rescaled to $\hat{t} = 1000\,t$ (matching the integer-timestep convention of DDPM [13] so that the same sinusoidal frequency range is useful), then embedded with a sinusoidal basis followed by a two-layer MLP:

$$\text{PE}(\hat{t})_j = \begin{cases} \sin\!\big(\hat{t}\cdot 10000^{-2j/d}\big) & j < d/2 \\ \cos\!\big(\hat{t}\cdot 10000^{-2(j-d/2)/d}\big) & j \geq d/2 \end{cases} \tag{21}$$

$$\mathbf{t}_{\text{emb}} = \mathbf{W}_4\,\sigma_{\text{SiLU}}(\mathbf{W}_3\,\text{PE}(\hat{t}) + \mathbf{b}_3) + \mathbf{b}_4 \in \mathbb{R}^d. \tag{22}$$

The timestep embedding is broadcast across frames and added to the coord tokens to form the initial gaze state:

$$\mathbf{h}_t^{(0)} = \mathbf{h}_t^{\text{coord}} + \mathbf{t}_{\text{emb}}, \qquad t = 1,\ldots,T. \tag{23}$$

### C.3 Encoder and Projection

**Feature extraction.** We use V-JEPA 2 ViT-L [1] as the visual encoder, trained end-to-end with the rest of the network. Given an egocentric video clip sampled at 24 fps, we resize frames to 256×256 (the pre-training resolution) and extract last-layer hidden states:

$$\mathbf{V}_t^{\text{raw}} \in \mathbb{R}^{N\times D}, \qquad t = 1,\ldots,T, \tag{24}$$

where $N = 256$ is the number of spatial tokens (16×16 grid) and $D$=1024 is the encoder hidden dimension.

**Backbone selection.** We use V-JEPA 2 ViT-L because its features capture both fine-grained motion and clip-level action structure [1]: the former supplies the local visual evidence read by our per-frame cross-attention (bottom-up), and the latter the task-level structure pooled by our task-token bank (top-down). A linear probe with frozen-features supports this story (Table 9): at the native resolution $256\times256$, V-JEPA 2 binds Qwen3-VL-8B [2] in AUC and exceeds the best Qwen3-VL layer in F1 (0.378 vs. 0.370) and Precision (0.292 vs. 0.280), while a scale-matched input $240\times320$ degrades every metric.

Table 9: **Linear probing of frozen backbone features on EGTEA Gaze+**. Top block: V-JEPA 2 ViT-L at its native pre-training resolution and at a 4:3 input that matches the dataset aspect ratio. Bottom block: Qwen3-VL-8B at four LM layers, native resolution. All probes are ridge regressions on the frozen spatial token grid; per-column best in bold.

| Backbone | Resolution | Layer | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ |
|---|---|---|---|---|---|---|---|
| **V-JEPA 2 ViT-L** | **256×256** (native) | last | **0.944** | **0.378** | **0.292** | 0.535 | 11.39 |
| V-JEPA 2 ViT-L | 240×320 (matched) | last | 0.913 | 0.296 | 0.216 | 0.469 | 12.87 |
| Qwen3-VL-8B | native | 2 | 0.944 | 0.370 | 0.280 | **0.549** | 11.37 |
| Qwen3-VL-8B | native | 4 | 0.944 | 0.370 | 0.280 | 0.547 | 11.37 |
| Qwen3-VL-8B | native | 8 | 0.943 | 0.369 | 0.278 | **0.549** | 11.34 |
| Qwen3-VL-8B | native | 16 | 0.942 | 0.368 | 0.279 | 0.541 | **11.24** |

**Trainable linear projection.** The raw encoder features are projected from $D{=}1024$ to the denoiser working dimension $d{=}256$ by a learned linear layer:

$$\mathbf{V}_t \;=\; \mathbf{W}_{\text{proj}}\,\mathbf{V}_t^{\text{raw}} + \mathbf{b}_{\text{proj}} \;\in\; \mathbb{R}^{N\times d}, \qquad \mathbf{W}_{\text{proj}} \in \mathbb{R}^{d\times D}. \tag{25}$$

The projection is trained jointly with the rest of the network; A 2D sinusoidal spatial positional encoding $\mathbf{P}^{\text{sp}} \in \mathbb{R}^{H'\times W'\times d}$ with $H' = W' = 16$ is added to $\mathbf{V}_t$ before it enters any cross-attention:

$$\mathbf{V}_t \leftarrow \mathbf{V}_t + \mathbf{P}^{\text{sp}} \quad \text{(broadcast over the } N{=}H'W' \text{ tokens)}. \tag{26}$$

We write $\mathbf{V} \in \mathbb{R}^{T\times N\times d}$ for the stack over frames. $\mathbf{V}_t$ then passes through a small per-frame Transformer encoder (two pre-norm layers, $H{=}8$ heads, expansion factor 4, GELU) that lets its $N$ spatial tokens exchange information within each frame. This pre-aggregation is computed once per clip and shared across all $L$ denoiser layers and all $S$ Euler steps; like the linear projection, it operates independently on each frame's grid and never crosses the temporal axis.

### C.4 Rotary Position Embedding

For the frame-to-frame self-attention (Sub-layer 1 below) we use 1D RoPE [43] indexed by the frame position $p \in \{0,\dots,T{-}1\}$, with base $\theta = 10000$. RoPE rotates each query/key pair $(q_{2j}, q_{2j+1})$ by $\phi_{p,j} = p\cdot\theta^{-2j/d_h}$ before the attention dot-product; values are not rotated. By construction the logit $a_{p,p'} = \langle \text{RoPE}(\mathbf{q}_p), \text{RoPE}(\mathbf{k}_{p'})\rangle/\sqrt{d_h}$ depends only on the relative offset $p-p'$, which is the structure gaze dynamics require (consecutive-frame coupling and saccade step length scale with $|p-p'|$, not with absolute $p$). Section 4.4 confirms this empirically: replacing RoPE with learned absolute PE degrades behavioral metrics.

### C.5 Task Injection Module

The Task Injection module produces the clip-level task tokens $\mathbf{c}_{\text{task}} \in \mathbb{R}^{(K+1)\times d}$ that every denoiser block (Section C.6) consumes via its Sub-layer 3 cross-attention. It combines two complementary summaries of the encoder features at different granularities: a parameter-free *global-pool token* and a set of $K{=}4$ *learnable task queries*.

**Space-marginal pre-reduction.** Starting from the projected encoder tokens $\mathbf{V} \in \mathbb{R}^{T\times N\times d}$ (Section C.3), we first marginalize over the spatial axis to obtain a per-frame summary

$$\bar{\mathbf{V}}_t \;=\; \frac{1}{N}\sum_{n=1}^{N} \mathbf{V}_{t,n,:} \;\in\; \mathbb{R}^d, \qquad \bar{\mathbf{V}} \;=\; [\bar{\mathbf{V}}_1;\dots;\bar{\mathbf{V}}_T] \;\in\; \mathbb{R}^{T\times d}. \tag{27}$$

Both the pool token and the learnable queries read from $\bar{\mathbf{V}}$, not from the full spatiotemporal grid; this is a deliberate design choice to keep the task module focused on clip-level temporal structure while the per-frame visual cross-attention inside the denoiser block handles spatial localization.

**Global-pool token.** A single token formed by further marginalizing over time:

$$\mathbf{p} \;=\; \frac{1}{T}\sum_{t=1}^{T} \bar{\mathbf{V}}_t \;\in\; \mathbb{R}^{1\times d}. \tag{28}$$

This is the unconditional clip-mean token. It is parameter-free and well-defined from the first training step, which makes it an effective stable anchor in the low-data regime (see Section 4.4).

**Learnable task queries.** A learnable parameter $\mathbf{Q}_{\text{task}} \in \mathbb{R}^{K\times d}$ is projected onto the time-marginal $\bar{\mathbf{V}}$ via a single multi-head cross-attention layer (distinct from the per-denoiser-block task cross-attentions defined later):

$$\tilde{\mathbf{Q}}_{\text{task}} \;=\; \text{LayerNorm}(\mathbf{Q}_{\text{task}}) \;\in\; \mathbb{R}^{K\times d}, \tag{29}$$

$$\mathbf{q}_k^{(h)} \;=\; \text{softmax}\left(\frac{\mathbf{W}_q^{\text{task},h}\,\tilde{\mathbf{Q}}_{\text{task},k}\,(\mathbf{W}_k^{\text{task},h}\,\bar{\mathbf{V}})^\top}{\sqrt{d_h}}\right)\mathbf{W}_v^{\text{task},h}\,\bar{\mathbf{V}} \;\in\; \mathbb{R}^{d_h}, \tag{30}$$

$$\mathbf{q}_k \;=\; \mathbf{W}_o^{\text{task}}\,[\mathbf{q}_k^{(1)};\,\ldots;\,\mathbf{q}_k^{(H)}] \;\in\; \mathbb{R}^{d}, \qquad k=1,\ldots,K, \tag{31}$$

with per-head projections $\mathbf{W}_q^{\text{task},h}, \mathbf{W}_k^{\text{task},h}, \mathbf{W}_v^{\text{task},h} \in \mathbb{R}^{d_h\times d}$ and output projection $\mathbf{W}_o^{\text{task}} \in \mathbb{R}^{d\times d}$. The bottleneck of $K=4$ queries forces each query to specialize on a different task-relevant temporal factor.

**Two-granularity concatenation.** The pool token and the learnable queries are concatenated along the token axis to form the task tokens:

$$\mathbf{c}_{\text{task}} \;=\; \big[\,\mathbf{p};\;\mathbf{q}_1;\;\ldots;\;\mathbf{q}_K\,\big] \;\in\; \mathbb{R}^{(K+1)\times d}. \tag{32}$$

The concatenation order is fixed for checkpoint compatibility but does not matter mathematically, since the downstream task cross-attention in the denoiser block (Eq. (35)) is permutation-invariant in its key/value set.

### C.6 DenoiserBlock: Four Sub-layers

The denoiser block is the consumer that ties together every tensor defined in the previous subsections: gaze tokens $\mathbf{h}$ from the input pipeline, projected encoder tokens $\mathbf{V}$ from Section C.3, and task tokens $\mathbf{c}_{\text{task}}$ from Section C.5. Each of the $L$=6 blocks applies four pre-norm residual sub-operations to the gaze tokens $\mathbf{h} \in \mathbb{R}^{T\times d}$. All attentions use $H=8$ heads with $d_h = d/H = 32$ and layer-specific QKV/output projections (multi-head attention, `MHA`, follows the standard definition).

**Sub-layer 1: Frame-to-frame self-attention with RoPE.**

$$\mathbf{h} \;\leftarrow\; \mathbf{h} + \text{MHA}\big(\text{RoPE}(\mathbf{Q}),\,\text{RoPE}(\mathbf{K}),\,\mathbf{V};\,\mathbf{h}\big), \tag{33}$$

where $\mathbf{Q},\mathbf{K},\mathbf{V}$ are linear projections of $\text{LayerNorm}(\mathbf{h})$ and RoPE is applied to queries and keys but not to values. This is the only pathway along which gaze tokens at different frames exchange information, and is therefore where the joint structure of $p(\mathbf{g}\mid\mathcal{V})$ enters the network.

**Sub-layer 2: Per-frame visual cross-attention.** Each gaze token $\mathbf{h}_t$ cross-attends *only* to its own frame's $N$ spatial tokens $\mathbf{V}_t$:

$$\mathbf{h}_t \;\leftarrow\; \mathbf{h}_t + \text{MHA}\big(\text{LayerNorm}(\mathbf{h}_t),\,\mathbf{V}_t,\,\mathbf{V}_t\big). \tag{34}$$

Long-range temporal aggregation is left to Sub-layer 1; this sub-layer queries the encoder precisely at the temporal location being denoised. The sub-layer is not gated: visual conditioning is the dominant spatial signal and runs at full strength from initialization.

**Sub-layer 3: Task cross-attention.** Each gaze token cross-attends to the clip-level task tokens $\mathbf{c}_{\text{task}} \in \mathbb{R}^{(K+1)\times d}$ defined in Section C.5:

$$\mathbf{h} \leftarrow \mathbf{h} + \sigma(g^{(\ell)}) \cdot \text{MHA}\big(\text{LayerNorm}(\mathbf{h}),\, \mathbf{c}_{\text{task}},\, \mathbf{c}_{\text{task}}\big). \tag{35}$$

The per-layer gate $g^{(\ell)} \in \mathbb{R}^d$ is a channel-wise learnable vector applied element-wise to the cross-attention residual (broadcast over the $T$ gaze tokens). All channels are initialized at $-2$ so that $\sigma(g_i^{(\ell)}) \approx 0.12$ uniformly at the start of training: the task signal starts weak and ramps up as the Task Injection module becomes informative, letting the rest of the network train without interference from a randomly initialized task pathway. At convergence, different layers settle at different mean gate values (the channel-averaged $\sigma$ typically lies in $[0.2, 0.5]$, with non-trivial per-channel spread).

**Sub-layer 4: Feed-forward network.** A two-layer GELU MLP with expansion factor 4:

$$\mathbf{h} \leftarrow \mathbf{h} + \mathbf{W}_6\, \text{GELU}\big(\mathbf{W}_5\, \text{LayerNorm}(\mathbf{h})\big), \tag{36}$$

$\mathbf{W}_5 \in \mathbb{R}^{4d\times d}$, $\mathbf{W}_6 \in \mathbb{R}^{d\times 4d}$ (biases and dropout omitted for brevity).

**Output head.** After the last block, $\mathbf{v}_\theta = \mathbf{W}_{\text{out}}\, \text{LayerNorm}(\mathbf{h}^{(L)}) + \mathbf{b}_{\text{out}} \in \mathbb{R}^{T\times 2}$.

# D Training and Sampling Algorithms

This section consolidates how GAZEFLOW is trained and how a trained model is used at inference. Appendix D.1 gives the full procedure as a single self-contained algorithm that covers both phases. Appendix D.2– Appendix D.4 detail the ingredients the algorithm uses: self-conditioning, sliding-window handling of long clips and the parameter EMA.

## D.1 Training and Inference Algorithm

Algorithm 1 states the two phases in one listing to emphasize the shared variable definitions (interpolant, target velocity, self-conditioning estimate) and to make clear which quantities are precomputed once per clip versus re-evaluated per Euler step.

Two observations about Algorithm 1. (i) The encoder forward and the Task Injection module are called once per clip at inference but once per optimizer step at training, because the encoder is itself being optimized. (ii) Self-conditioning at inference uses the value computed in the *previous* Euler step; the initial value $\hat{\mathbf{x}}_1^{(k)} = \mathbf{0}$ matches the $\hat{\mathbf{x}}_1^{\text{prev}} = \mathbf{0}$ branch of training, so the network's expected input distribution at $s = 0$ is consistent between training and inference.

## D.2 Self-Conditioning Details

Self-conditioning [6] feeds the model's own previous *clean-endpoint* estimate back as additional input, enabling iterative refinement across ODE steps without the cost of an autoregressive decoder. Algorithm 1 shows where this fits; here we record the subtleties.

**Training schedule.** On each training step we flip a fair Bernoulli coin. With probability $0.5$ we run the network once with the hint set to zero to obtain a preliminary velocity $\hat{\mathbf{v}}$, convert it to a clean-endpoint estimate via the exact inverse-flow formula

$$\hat{\mathbf{x}}_1^{\text{prev}} \;=\; \text{clamp}\big(\mathbf{x}_s + (1-s)\,\hat{\mathbf{v}},\; [-1, 1]\big), \tag{37}$$

stop the gradient, and then forward again with $\hat{\mathbf{x}}_1^{\text{prev}}$ concatenated into the coordination encoder input (making its input width $c = 4$ instead of $c = 2$, Eq. (20)). With the other $0.5$ probability, $\hat{\mathbf{x}}_1^{\text{prev}}$ is set to $\mathbf{0}$ (no hint). This mixture ensures that at inference the network is well-defined both at the first integration step ($s_0 = 0$, no previous estimate) and at later steps ($s_i > 0$ for $i \geq 1$, where one is available). The wall-clock overhead of the two-pass schedule is approximately $16\%$ in our setup.

**Inference update rule.** During Euler integration, $\hat{\mathbf{x}}_1^{(k)}$ is refreshed at every step from the *same* inverse-flow estimate (inference phase of Algorithm 1):

$$\hat{\mathbf{x}}_1^{(k,\text{next})} \;=\; \text{clamp}\big(\mathbf{x}_{s_i}^{(k)} + (1-s_i)\,\mathbf{v}_{\bar{\theta}}(\mathbf{x}_{s_i}^{(k)}, s_i, \mathbf{c}_v, \mathbf{c}_{\text{task}}, \hat{\mathbf{x}}_1^{(k)}),\; [-1, 1]\big). \tag{38}$$

As $s_i$ approaches 1, the inverse-flow factor $(1 - s_i)$ shrinks and $\hat{\mathbf{x}}_1^{(k)}$ converges to $\mathbf{x}_{s_i}^{(k)}$ itself; the self-conditioning input therefore matters most at early steps when the network has seen little structure, and fades out naturally near the end of integration.

**Algorithm 1** Training and inference for GAZEFLOW.

**Require:** Dataset $\mathcal{D}$; encoder $E_\psi$; velocity net $\mathbf{v}_\theta$; EMA weights $\bar{\theta}$; ODE steps $S$; samples per clip $K_{\text{samples}}$; self-cond probability $p_{\text{sc}}$=0.5; EMA decay $\beta$=0.995.

**Training phase.**

**repeat**
  Sample $(\mathcal{V}, \mathbf{g}, \mathbf{m}) \sim \mathcal{D}$; $\mathbf{c}_v \leftarrow E_\psi(\mathcal{V})$; $\mathbf{c}_{\text{task}} \leftarrow \text{TaskInjection}(\mathbf{c}_v)$.
  Sample $s \sim U[0,1]$, $\mathbf{x}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; set $\mathbf{x}_s, \mathbf{u}_s$ from Eqs. (9), (12).
  **if** Bernoulli$(p_{\text{sc}}) = 1$ **then** ▷ two-pass self-conditioning
    $\hat{\mathbf{v}} \leftarrow \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}_v, \mathbf{c}_{\text{task}}, \mathbf{0})$ **(no grad)**; $\hat{\mathbf{x}}_1^{\text{prev}} \leftarrow \text{clamp}(\mathbf{x}_s + (1-s)\hat{\mathbf{v}})$.
  **else**
    $\hat{\mathbf{x}}_1^{\text{prev}} \leftarrow \mathbf{0}$.
  **end if**
  $\hat{\mathbf{v}} \leftarrow \mathbf{v}_\theta(\mathbf{x}_s, s, \mathbf{c}_v, \mathbf{c}_{\text{task}}, \hat{\mathbf{x}}_1^{\text{prev}})$; $\mathcal{L} \leftarrow \|\mathbf{m} \odot (\hat{\mathbf{v}} - \mathbf{u}_s)\|_2^2 / \max(\|\mathbf{m}\|_1, 1)$.
  AdamW update on $(\psi, \theta)$; $\bar{\theta} \leftarrow \beta\bar{\theta} + (1-\beta)\theta$.
**until** convergence

**Inference phase** (parallel $K_{\text{samples}}$-trajectory Euler ODE).

$\mathbf{c}_v \leftarrow E_\psi(\mathcal{V})$; $\mathbf{c}_{\text{task}} \leftarrow \text{TaskInjection}(\mathbf{c}_v)$. ▷ once per clip
**for** $k = 1, \ldots, K_{\text{samples}}$ **in parallel do**
  $\mathbf{x}_0^{(k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$; $\hat{\mathbf{x}}_1^{(k)} \leftarrow \mathbf{0}$.
  **for** $i = 0, \ldots, S-1$ **do**
    $s_i \leftarrow i/S$; $\mathbf{v} \leftarrow \mathbf{v}_{\bar{\theta}}(\mathbf{x}_{s_i}^{(k)}, s_i, \mathbf{c}_v, \mathbf{c}_{\text{task}}, \hat{\mathbf{x}}_1^{(k)})$.
    $\hat{\mathbf{x}}_1^{(k)} \leftarrow \text{clamp}(\mathbf{x}_{s_i}^{(k)} + (1-s_i)\mathbf{v})$; $\mathbf{x}_{s_{i+1}}^{(k)} \leftarrow \mathbf{x}_{s_i}^{(k)} + (1/S)\mathbf{v}$.
  **end for**
**end for**
**return** $\{\text{clamp}(\mathbf{x}_1^{(k)})\}_{k=1}^{K_{\text{samples}}}$.

**Initial condition.** We initialize $\hat{\mathbf{x}}_1^{(k)} = \mathbf{0}$ at $s = 0$ (inference phase of Algorithm 1). Because the zero-hint branch during training is triggered on exactly half of the steps, the network has seen the $(\mathbf{x}_s, s, \mathbf{0})$ input distribution during training and does not suffer a train–inference gap at the first step.

### D.3 Sliding-Window Inference for Long Clips

The velocity network is trained on fixed windows of $W = 64$ frames. At inference, clips longer than $W$ are split into overlapping windows of stride $W_s = W/2 = 32$, and Algorithm 1 is run independently inside each window. In the overlap region between consecutive windows $i$ and $i{+}1$, predictions are linearly blended along the frame axis $\tau$ to avoid window seams:

$$\mathbf{x}_\tau^{\text{blend}} \;=\; (1-\alpha_\tau)\,\mathbf{x}_\tau^{(i)} + \alpha_\tau\,\mathbf{x}_\tau^{(i+1)}, \qquad \alpha_\tau \;=\; \frac{\tau - \tau_{\text{start}}^{(i+1)}}{W_s} \in [0,1], \tag{39}$$

where $\tau_{\text{start}}^{(i+1)}$ is the first frame of window $i{+}1$ and $\alpha_\tau$ ramps from 0 to 1 across the overlap. Clips shorter than $W$ are zero-padded before inference and truncated back to their original length afterwards.

### D.4 Exponential Moving Average

We maintain an exponential moving average of the denoiser parameters [39]:

$$\bar{\theta}^{(i)} \;=\; \beta\,\bar{\theta}^{(i-1)} + (1-\beta)\,\theta^{(i)}, \tag{40}$$

with decay $\beta = 0.995$. All reported evaluations and all sampled trajectories use the EMA weights $\bar{\theta}$, not the online $\theta$.

## E Experimental Details

This appendix supplements Section 4.1 and the experimental sections of the main text. Section E.1 details the two datasets; Section E.2 describes the baselines and their training recipes; Section E.3

specifies the metrics, the GLC-format evaluation protocol, and the trajectory-to-saliency conversion; Section E.4 lists our training and sampling hyperparameters; Section E.5 reports parameters, FLOPs, wall-clock, and peak memory on a single H100; Section E.6 expands Table 2 with the full per-frame metric breakdown; Section E.7 sweeps the number of learnable task queries used to build the task-token bank; and Section E.8 sweeps the trajectory sample count $K$ used at inference.

### E.1 Datasets

**EGTEA Gaze+.** EGTEA Gaze+ [30] contains 86 cooking videos (106 action classes with 19 verbs and 53 nouns) recorded at $640 \times 480$ / 24 fps together with synchronized gaze from an SMI head-mounted eye tracker at 30 Hz. We follow the official train / val split from Li et al. [30], yielding 8,299 training and 2,022 validation clips after aligning video and gaze timelines. Eye-tracker samples are classified into Fixation, Saccade, and Blink events by the authors' BeGaze output; we carry forward all event types into the trajectory target but exclude Blink frames from the loss via a binary validity mask $\mathbf{m}$.

**Ego4D (Aria-glass gaze subset).** The Ego4D [10] benchmarks include a subset of clips captured with Project Aria glasses equipped with an integrated eye tracker, providing per-frame gaze at $1088 \times 1080$ / 30 fps. We use the released train / val partition (12,178 / 5,202 clips). Raw gaze is already aligned to the video grid by the Aria SDK; we rescale coordinates to the native $1088 \times 1080$ resolution, normalize to $[-1, 1]$ for training. No action/verb labels are used for Ego4D; it is strictly a cross-dataset generalization target.

**Windowing.** All experiments operate on fixed $T = 64$-frame windows drawn from each clip: clips shorter than $T$ are right-padded with the last valid frame and their padding positions are zeroed in the validity mask; clips longer than $T$ are trained with random crops and evaluated with a sliding window (stride 32, linear blend in the overlap, Appendix D.3). Clip-length statistics and per-split window counts are summarized in Table 10.

Table 10: Dataset summary.

| Dataset | Resolution | fps | Train clips | Val clips |
|---|---|---|---|---|
| EGTEA Gaze+ | $640 \times 480$ | 24 | 8,299 | 2,022 |
| Ego4D (Aria gaze) | $1088 \times 1080$ | 30 | 12,178 | 5,202 |

### E.2 Baselines

**Attention Transition (AT).** AT [15] is trained in three stages: (SP) a two-stream CNN predicts a per-frame heatmap, (LSTM) a recurrent module classifies each frame as Fixation or Saccade, and (LF) a late-fusion CNN re-weights SP by the fixation–saccade posterior. On EGTEA we retrain the full SP+LSTM+LF pipeline with the authors' released recipe on the 8,299 train clips. On Ego4D, fixation/saccade event labels are not provided, so only Stage SP is retrained and the LSTM and LF stages are disabled. AT inference heatmaps are resized to $480 \times 640$ before evaluation.

**GLC.** GLC [24] is a transformer saliency model with a "global–local correlation" head; we use the authors' released checkpoints for EGTEA and Ego4D without modification. Predictions are produced at $64 \times 64$ and upscaled to $480 \times 640$ for evaluation to match our saliency protocol.

**EgoM2P.** EgoM2P [28] is an autoregressive multimodal foundation model pretrained on eight egocentric datasets that emits discrete gaze tokens one at a time. We report two variants: (i) the released zero-shot checkpoint, and (ii) a LoRA-finetuned variant with rank $r = 16$ applied to the query and value projections of all transformer layers, trained on each dataset's train split with the same optimizer as our own model. For both variants we draw $K_{\text{samples}}$=50 trajectories per clip at temperature 1.0, matching our sampler.

**Prior-only references.** Two trivial priors anchor the lower end of the scale in Table 2. *Center Bias* is a static 2D Gaussian centered at the image center with per-axis standard deviation matched to the empirical fixation spread of each dataset's train split; it is evaluated with the same saliency protocol as every other baseline and has no trainable parameters. *Random Walk* takes the previous frame's ground-truth fixation (or the clip centroid at $t$=0) and samples the next predicted fixation from an

isotropic Gaussian with $\sigma$ matched to the empirical frame-to-frame displacement of the train split; it is explicitly a first-order Markov prior and carries no visual input at all. Both references are reported only for the headline saliency tables and are excluded from the trajectory-dynamics comparison.

### E.3 Metrics and Evaluation Protocol

The metrics below mirror the two-axis structure of Section 4.1: *per-frame prediction accuracy* scores each predicted heatmap against the per-frame ground-truth gaze, while *alignment with human gaze temporal dynamics* scores sampled trajectories against the human trajectory either directly or as a step-size distribution.

**Per-frame prediction accuracy.** In egocentric gaze datasets, each frame is annotated with a ground-truth gaze point. Following the GLC protocol [24], we place a Gaussian kernel at this point to form a fixation map, which is then compared with the predicted heatmap. *Area Under the Curve* (AUC, the AUC-Judd variant [5, 19]) treats the predicted heatmap as a continuous per-pixel score and measures how well it ranks ground-truth gaze locations above non-gaze locations. *F1*, *Precision*, and *Recall* instead binarize the predicted heatmap and the ground-truth fixation map at a fixed threshold and measure the overlap between the two. *Average Angular Error* (AAE) [15] measures the angular deviation between the predicted gaze point and the ground-truth gaze point.

Per-frame metrics aggregate hierarchically: per frame, then averaged within each clip (the clip-mean), then averaged across clips (the global mean), which follows GLC's original protocol and guarantees that each clip contributes equally regardless of length. For F1, Precision, and Recall, the threshold is selected by sweeping over the heatmap; following Lai et al. [24], we report Adaptive F1 (the best F1 over the sweep) with Precision and Recall taken at the same threshold. Beyond AUC-Judd, we additionally report four standard saliency metrics following Bylinskii et al. [5]: NSS (the mean normalized saliency at fixation locations), the linear correlation coefficient CC, the histogram similarity SIM, and the KL divergence from the ground-truth to the predicted distribution.

**Alignment with human gaze temporal dynamics.** Per-frame metrics score each frame independently and therefore do not measure whether the predicted gaze forms a natural scanpath over time. The metrics below evaluate how closely the dynamics of sampled trajectories match those of human gaze. *Average Displacement Error* (ADE) and *Dynamic Time Warping distance* (DTW) [38, 21] compare each predicted trajectory against the human trajectory directly; DTW is the minimum Euclidean alignment cost between the two under a monotone temporal warping, length-normalised by the trajectory length. For both, we report the mean over $K = 50$ samples and the best-of-$K$ value.

We additionally compare step-size statistics. For each trajectory, we measure how far the gaze moves between consecutive frames, and build a displacement histogram, which represents the empirical distribution of these per-frame distances, aggregated across all frames and clips. We report the mean and median displacement as ratios to the human reference, and the Jensen–Shannon divergence (JSD) between the predicted and human displacement histograms. The JSD uses base $e$ on a 100-bin histogram over the union range of all method and human displacements, which captures distribution shape rather than only its first two moments.

**Trajectory-to-heatmap conversion.** Trajectory generators – our model, EgoM2P, and pseudo-trajectory baselines – are converted to per-frame heatmaps so that all methods are comparable on the saliency metrics above. For each frame $t$ we (i) denormalize all $K_{\text{samples}}$ predicted gaze points from $[-1, 1]$ to pixel coordinates (the native evaluation resolution of the target dataset); (ii) place a Gaussian kernel with $\sigma = 25$ px at each prediction; (iii) average over $K_{\text{samples}}$ to obtain $\mathbf{S}_t$. The bandwidth $\sigma$=25 px matches the one used by the released GLC predictions.

**Frozen-encoder protocol for ablations.** The ablation study (Section 4.4) and the design-choice analysis (Section 4.5) evaluate the frozen-encoder variant of GAZEFLOW on a fixed validation subset, rather than the LoRA variant on the full validation split. The choice is justified on two grounds. First, end-to-end LoRA training together with $K$=50 sampling for every architectural variant is computationally prohibitive. Second, every component varied in those studies (the rotary positional encoding, the task-injection module, the spatial encoder, self-conditioning, and the cross-attention versus concatenation choice) operates on top of the encoder representation rather than altering the encoder itself, so encoder adaptation is orthogonal to the architectural choice under study; the relative ranking of variants is therefore unaffected by whether the encoder is frozen or LoRA-adapted.

**Validation subset.** The subset is a random sampled subset of the EGTEA Gaze+ validation split, drawn once with a fixed random seed and held identical across every variant in the ablation and design-choice studies. After applying the GLC-format Fixation filter, the subset contains 1,536 frame-level paired observations per metric.

**Significance testing.** For the main comparison (Table 2), every ablation row of Table 6, and every design-choice variant in Section 4.5, we report a paired Wilcoxon signed-rank test against the corresponding reference: GAZEFLOW for the baselines in Table 2, the full model for the ablation rows, and the cross-attention variant for the concatenation alternative. We choose the Wilcoxon test rather than a paired $t$-test because the per-frame score distributions are heavy-tailed and not well approximated by a Gaussian, and we pair the two scores frame by frame to control for the substantial variance in clip difficulty (a clip with cluttered fine-grained manipulations is harder for every method, which inflates the unpaired variance). Each test is evaluated on the frame-level scores per metric described above, and we reject the null hypothesis of zero median difference at $\alpha = 0.05$. All comparisons reported in Section 4.2, Section 4.4, and Section 4.5 pass this test at $p < 0.05$ on almost every metric, with the vast majority reaching $p < 10^{-4}$ and only one or two GAZEFLOW-vs-baseline pairs in Table 2 sitting in the $[10^{-4}, 0.05)$ range. The exception is *Recall* on Ego4D in the main comparison (Table 2), where Random Walk is not beaten by GAZEFLOW.

### E.4 Training and Sampling Hyperparameters

**Video feature extraction.** The visual encoder is V-JEPA 2 ViT-L [1], fine-tuned with LoRA jointly with the velocity network ("LoRA configuration" below). Frames are resized to $256 \times 256$ and read at 24 fps on EGTEA Gaze+ and 30 fps on Ego4D, and we take the last hidden state of the encoder. This yields $N = 256$ spatial tokens of dimension $D = 1024$ per frame.

**Optimizer and schedule.** AdamW with $\beta_1$=0.9, $\beta_2$=0.999, weight decay $10^{-4}$; learning rate $10^{-4}$ with cosine annealing to $10^{-6}$ over 80 epochs; effective batch size 16 (per-GPU batch 4 with gradient-accumulation factor 4 on 4× H100 via DDP); EMA decay $\beta_{\text{EMA}}$=0.995; gradient clipping at norm 1.0. Early stopping uses patience 15 on validation loss computed at end of each epoch.

**LoRA configuration.** The V-JEPA 2 ViT-L encoder is fine-tuned with LoRA [14] rank $r$=16, $\alpha$=32, dropout 0.05, applied to the query and value projections of every transformer block. All other encoder weights are frozen; only LoRA adapters, the input projection $\mathbf{W}_{\text{proj}}$, and the velocity network are trainable.

**Flow-matching parameters.** We use the OT-form conditional interpolant $\mathbf{x}_s = (1-(1-\sigma_{\min})s)\,\mathbf{x}_0 + s\,\mathbf{g}$ with $\sigma_{\min} = 10^{-3}$, which differs from the linear interpolant in Eq. (9) only by an $O(\sigma_{\min})$ correction below pixel precision; self-conditioning is applied with probability 0.5 during training and deterministically on at inference; training uses the uniform schedule $s \sim \mathcal{U}(0, 1)$ as in Lipman et al. [31]. Self-conditioning adds ∼16% wall-clock overhead per step.

**Sampling.** Euler ODE integration with $S = 50$ steps and $K_{\text{samples}} = 50$ trajectories per clip, processed in batches of 16. Sliding window $W = 64$, stride 32, linear blend in overlap (Appendix D.3).

### E.5 Efficiency Analysis

We report model size, inference FLOPs, wall-clock, and memory measured on a single NVIDIA H100 (80 GB) in BF16. Unless noted, the test clip is a $T$=64-frame (∼2.7 s at 24 fps) EGTEA segment sampled with $K$=50 trajectories and $S$=50 Euler steps.

**Model size.** The velocity network has ∼10.4M parameters: ∼7.9M in the six DenoiserBlocks (each with the four sub-layers of Section 3.2), ∼1.6M in the spatial-token encoder, ∼0.5M jointly across the visual-token projection $\mathbf{W}_{\text{proj}}$ and the learnable-query cross-attention that builds $\mathbf{c}_{\text{task}}$, and the remainder (∼0.4M) across coordinate / timestep / self-conditioning encoders and the output layer. The V-JEPA 2 ViT-L visual encoder contributes ∼327.8M parameters, of which the LoRA adapters add ∼1.9M trainable parameters. Total trainable parameters: ∼12.2M.

**Inference FLOPs.** A single velocity-network forward pass at $T$=64, $K$=1 costs ∼100 GFLOPs. Generating $K$=50 samples with $S$=50 Euler steps requires $S \cdot K$=2,500 such calls, totaling ∼ 250 TFLOPs per clip. The V-JEPA 2 backbone encoder is amortised once per clip and contributes a comparatively small additional cost (<5%).

**Wall-clock and memory.** On a single H100, end-to-end inference of one 2.7 s clip takes $\sim$3.37 s: $\sim$450 ms for the V-JEPA 2 forward pass with LoRA, plus $\sim$2.92 s for the full $K{=}50 \times S{=}50$ Euler sampling (mean over five runs). Peak GPU memory is 9.2 GB. Sampling all 2,022 EGTEA validation clips at $K{=}50$ takes $\sim$30 min on a single H100; the 5,202 Ego4D validation clips take $\sim$80 min.

**Training time.** End-to-end LoRA training takes $\sim$3 days on EGTEA Gaze+ and $\sim$5 days on Ego4D, both on $4\times$ H100 GPUs with DDP and BF16 mixed precision.

### E.6 Full Main Comparison Table

Table 11 expands Table 2 with all nine standard metrics, split into the saliency block (NSS, AUC, CC, KL, SIM) and the localization block (F1, P, R, AAE).

Table 11: Full saliency and localization comparison on EGTEA Gaze+ and Ego4D.

| | **EGTEA Gaze+** | | | | | | | | | **Ego4D** | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Saliency | | | | | Localization | | | | Saliency | | | | | Localization | | | |
| Method | NSS↑ | AUC↑ | CC↑ | KL↓ | SIM↑ | F1↑ | P↑ | R↑ | AAE↓ | NSS↑ | AUC↑ | CC↑ | KL↓ | SIM↑ | F1↑ | P↑ | R↑ | AAE↓ |
| *Prior-only reference* | | | | | | | | | | | | | | | | | | |
| Center Bias | 0.679 | 0.906 | 0.095 | 6.635 | 0.110 | 0.185 | 0.162 | 0.215 | 16.07 | 0.780 | 0.930 | 0.122 | 5.873 | 0.136 | 0.222 | 0.194 | 0.258 | 13.92 |
| Random Walk | 0.623 | 0.824 | 0.096 | 3.926 | 0.101 | 0.147 | 0.090 | 0.413 | 16.34 | 0.113 | 0.763 | 0.017 | 4.256 | 0.072 | 0.109 | 0.059 | **0.662** | 14.38 |
| *Marginal density estimation* | | | | | | | | | | | | | | | | | | |
| GLC [24] | 2.216 | 0.953 | 0.279 | 3.288 | 0.249 | 0.421 | 0.348 | 0.531 | 10.21 | 1.960 | 0.947 | 0.256 | 3.438 | 0.231 | 0.355 | 0.267 | 0.530 | 11.52 |
| *Autoregressive* | | | | | | | | | | | | | | | | | | |
| EgoM2P (zero-shot) [28] | 1.586 | 0.921 | 0.231 | 2.946 | 0.189 | 0.280 | 0.195 | 0.498 | 13.18 | 1.499 | 0.911 | 0.218 | 3.150 | 0.178 | 0.253 | 0.180 | 0.422 | 13.86 |
| EgoM2P (LoRA) [28] | 1.735 | 0.921 | 0.247 | 2.737 | 0.198 | 0.293 | 0.205 | 0.515 | 12.81 | 1.886 | 0.930 | 0.267 | 2.715 | 0.206 | 0.297 | 0.217 | 0.470 | 12.69 |
| AT [15] | 2.722 | 0.956 | 0.378 | 1.949 | 0.269 | 0.419 | 0.339 | 0.547 | 9.88 | 2.289 | 0.954 | 0.327 | **2.156** | 0.197 | 0.346 | 0.268 | 0.487 | 12.01 |
| *Flow matching (ours)* | | | | | | | | | | | | | | | | | | |
| **GAZEFLOW (ours)** | **3.534** | **0.964** | **0.438** | **1.936** | **0.357** | **0.491** | **0.414** | **0.603** | **9.01** | **2.676** | **0.958** | **0.358** | 2.213 | **0.282** | **0.392** | **0.307** | 0.540 | **10.46** |

### E.7 Capacity Sweep over Task Queries

Table 12 reports the full sweep over $K \in \{1, 2, 4, 8, 16\}$ with the global pool token kept (referenced from Section 4.5 (i)). $K{=}4$ is optimal on the per-frame accuracy metrics (AUC, F1) and the curve is inverted-U on both: smaller $K$ underspecifies the task code, larger $K$ dilutes per-query specialization.

Table 12: **Capacity sweep over learnable task queries**; the global pool token is kept. Per-column best in bold.

| $K$ | AUC↑ | F1↑ | P↑ | R↑ | AAE↓ |
|---|---|---|---|---|---|
| 1 | 0.968 | 0.492 | **0.404** | 0.630 | **8.68** |
| 2 | 0.967 | 0.486 | 0.392 | 0.639 | 8.87 |
| **4** | **0.970** | **0.493** | 0.401 | 0.638 | 8.81 |
| 8 | 0.968 | 0.483 | 0.390 | 0.635 | 8.79 |
| 16 | 0.968 | 0.480 | 0.381 | **0.650** | 8.75 |

### E.8 Sample Count $K$ Sweep

Table 13 sweeps the sample count $K \in \{1, 5, 10, 30, 50\}$ for GAZEFLOW (LoRA, full EGTEA Gaze+ validation split), using the same $K{=}50$ samples as Table 3 and subsampling the first $K$. The mean over $K$ is essentially flat (1.05 $\rightarrow$ 1.04 for ADE, 0.96 $\rightarrow$ 0.95 for DTW), so the gains in best-of-$K$ are an order-statistic effect rather than improved per-sample quality. Best-of-$K$ has effectively converged by $K{=}30$ (ADE 0.64 vs. 0.60 at $K{=}50$), so the $K{=}50$ used in the main comparison is a safe operating point.

Table 13: **Sample count $K$ sweep on EGTEA Gaze+ (GAZEFLOW LoRA, full validation split).** ADE and DTW are reported in $\times 10^2$ px following Table 3; "Mean" and "Best" are mean-over-$K$ and best-of-$K$. At $K{=}1$ the two coincide.

| | ADE ($\times 10^2$ px)↓ | | DTW ($\times 10^2$ px)↓ | |
|---|---|---|---|---|
| $K$ | Mean | Best | Mean | Best |
| 1 | 1.05 | 1.05 | 0.96 | 0.96 |
| 5 | 1.05 | 0.80 | 0.95 | 0.68 |
| 10 | 1.05 | 0.73 | 0.95 | 0.60 |
| 30 | 1.04 | 0.64 | 0.95 | 0.51 |
| 50 | 1.04 | 0.60 | 0.95 | 0.48 |