EgoM2P: Egocentric Multimodal Multitask Pretraining
– Supplementary Material –

A. Implementation Details

A.1. Tokenizers

We tokenize spatiotemporal multimodalities into discrete tokens. For RGB and depth video, we use the state-of-the-art (SOTA) Cosmos tokenizer [3]. For gaze dynamics and camera trajectory, we train modality-specific tokenizers based on vector-quantized autoencoder.

For the gaze data $\mathbf{X}^{\text{gaze}} \in \mathbb{R}^{T\times 2}$, we first apply a convolution with a kernel size of 2 to temporally downsample it while mapping the channel dimension from 2 to 768. Then, we use 12 Transformer blocks (ViT-B) with self-attention to encode the data. Following best practices, we use cosine-sine similarity codebook with normalized codes. During training, we update the codebook entries to make sure all entries are used effectively. We track the exponential moving average of the codebook entry usage and replace under-utilized codes with the EMA dead code threshold. Quantized discrete tokens are then fed into the decoder with 12 Transformer blocks (ViT-B). Due to the hardware constraints, gaze data obtained from headsets, especially HoloLens [75], contains considerable invalid numbers. We choose not to discard these sequences, mask out invalid numbers, and use them as input. While calculating the reconstruction loss, we use masked L2 loss:

$$\mathcal{L}_{\text{gaze}} = \frac{\sum_{i=1}^{N} m_i (y_i - \hat{y}_i)^2}{\sum_{i=1}^{N} m_i}$$

where y_i is the ground truth, $\hat{y}_i$ is the predicted value, and m_i is the binary mask indicating valid values (1 for valid, 0 for invalid). The denominator ensures normalization by the number of valid elements. By leveraging the smoothness of deep networks, invalid gaze could also be predicted.

For the camera trajectory data $\mathbf{X}^{\text{cam}} \in \mathbb{R}^{T\times 9}$, we select the first two columns of the rotation matrix and the camera translation and stack them together. The only difference with the gaze tokenizer is the temporal convolution. This convolution performs temporal downsampling with a factor of 2 and maps the channel from 9 to 768. The training details are listed in Tab. A.1. "Batch size" refers to the number of samples per GPU. The "Learning rate" is determined by multiplying the "Base Learning rate" by the "Total batch size" and then dividing by 256 following [28].

The codebook size is 64000 for video modalities, while the gaze and camera modalities have a codebook size of 256. The number of parameters for gaze and camera tokenizers is approximately 180 million. The camera trajectory tokenizer trains in 1 day, while the gaze tokenizer takes 12 hours, both using 4 NVIDIA GH200 superchips, each with an H100 GPU and 96GB of RAM.

Configuration	Gaze Dyn.	Camera Traj.
Codebook size	256	
Temporal compression	2	
Code latent dimension	32	
EMA dead code threshold	2	
Codebook EMA	0.99	
l_2-normalized codes [120]	✓	
Codebook weight	1.0	
Commitment weight β	1.0	
Encoder architecture	ViT-B	
Decoder architecture	ViT-B	
Loss function	Masked MSE	MSE
Optimizer	AdamW [63]	
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.95$	
Weight decay	0.05	
Base learning rate	5e-5	2.5e-5
Learning rate	1e-4	5e-5
Batch size	128	
Total batch size	512	
Max gradient norm	1	
Learning rate sched.	Cosine decay	
Training epochs	200	
Warmup epochs	5	
Data type	float32	

Table A.1. **Tokenizer training settings.**

A.2. Multimodal Token Sampling

First, we randomly sample a dataset from eight curated egocentric datasets with probability proportional to the number of samples they contain. Then, we sample tokens from available multimodalities with a family of symmetric Dirichlet distributions:

1. **Sampling a Dirichlet Distribution:** Let $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, $\boldsymbol{\alpha}_3$, $\boldsymbol{\alpha}_4$ be the four concentration parameters of the Dirichlet dis-

tributions. We select $\boldsymbol{\alpha}_i$ with uniform probability:

$$\begin{aligned}\boldsymbol{\alpha}_1 &= (0.01, 0.01, 0.01, 0.01),\\ \boldsymbol{\alpha}_2 &= (0.1, 0.1, 0.1, 0.1),\\ \boldsymbol{\alpha}_3 &= (1, 1, 1, 1),\\ \boldsymbol{\alpha}_4 &= (10, 10, 10, 10).\end{aligned}$$

If the dataset contains missing modalities, the selected concentration parameter vector $\boldsymbol{\alpha}_i$ is adjusted to include only the parameters corresponding to the available modalities.

2. **Sampling a Probability Vector:** When $\boldsymbol{\alpha}_i$ is sampled, we then sample a probability vector $\boldsymbol{\theta}$ from the chosen Dirichlet distribution:

$$\boldsymbol{\theta} \sim \text{Dirichlet}(\boldsymbol{\alpha}_i).$$

Here, $\boldsymbol{\theta}$ represents a probability distribution over available modalities.

3. **Sampling Tokens from the Modalities:** Finally, given $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_n)$, the number of tokens for each modality i is $T_i = 2048 \times \theta_i$. Within this cap, tokens from each modality are sampled randomly. 2048 is the maximum number of input and target tokens.

A.3. EgoM2P Pretraining Details

For the model architecture, we adapt the T5-Base model. It has 12 Transformer blocks in both the encoder and decoder. The latent dimension is 768. The model uses 12 attention heads in its multi-head attention mechanism. For each modality embedding layer, it maps each token in the codebook to a 768-dimensional space. In Tab. A.2, we detail the training hyperparameters. The model is trained with distributed data parallel.

The total number of parameters of *EgoM2P* is approximately 400 million. During training, the maximum number of sampled tokens for both input and target is 2048. The total number of training tokens is 400 billion, randomly sampled and masked from our database of 4 billion tokens.

The model training takes 16 hours using 256 NVIDIA GH200 superchips. All networks, including tokenizers, are trained from scratch without initializing parameters from existing large models.

A.4. Missing Modality Handling Details

Unlike 4M, which relies on an aligned multimodal dataset containing all modalities, our approach uses missing modality masking. This allows us to scale training across multiple real-world multimodal egocentric datasets, even when the modalities are unaligned. See Alg. A.1 for the pseudo-code.

A.5. Inference Details

As an order-agnostic autoregressive model, *EgoM2P* supports parallel decoding in each decoding step. All target tokens can be decoded at the same time, however, increasing the decoding step s to 3 to 6 generally produces higher-quality predictions. In each decoding step, previously decoded target tokens are conditioned to ensure prediction consistency. See Alg. A.2 for the pseudo-code.

Configuration	**EgoM2P**
Training tokens	400B
Warmup tokens	10B
Optimizer	AdamW [63]
Opt. momentum	$\beta_1, \beta_2 = 0.9, 0.99$
Base learning rate	1e-4
Total batch size	1024
Weight decay	0.05
Max gradient norm	1
Learning rate sched.	Cosine decay
Max input token number	2048
Max target token number	2048
Data type	bfloat16

Table A.2. **Pretraining settings.**

Algorithm A.1 Handling Unaligned Multimodalities

```
Input: Data iterators $\{D_i\}_{i=1}^N$, dataset sampling probabilities $\mathbf{p} = \{p_i\}_{i=1}^N$, training modalities $\{m_j\}_{j=1}^M$
$i \sim$ Categorical($\mathbf{p}$)    ▷ Sample dataset index $i$
/* Sample input and target tokens for i (Sec. A.2) */
$\boldsymbol{\alpha} \sim [\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \boldsymbol{\alpha}_3, \boldsymbol{\alpha}_4]$    ▷ Sample Dir. concentr. param
$\boldsymbol{\alpha} \leftarrow$ Adjust($\boldsymbol{\alpha}$, missing modalities)    ▷ Exclude params for missing modalities
Sample input token counts $ic$ for each modality    ▷ Ensure modality token limits per sample
Sample target token counts $tc$    ▷ Modality token limits per sample are adjusted by subtracting $ic$
$\mathbf{im} \leftarrow \mathbf{1}, \mathbf{tm} \leftarrow \mathbf{1}$    ▷ Init input and target mask
Random sample $ic$ tokens, set input mask $\mathbf{im}$ to 0
Random sample $tc$ tokens (non-overlapping with $ic$), set target mask $\mathbf{tm}$ to 0
$\mathbf{x} \leftarrow \{\text{data}, \mathbf{im}, \mathbf{tm}\}$    ▷ For each existing modality
$\mathbf{x}' \leftarrow \{\}$
for $j = 1$ to $M$ do    ▷ Iterate over $M$ modalities
    $\mathbf{x}'[j][\text{data}] \leftarrow \mathbf{0}$    ▷ Tensor initialized to 0
    $\mathbf{x}'[j][\mathbf{im}] \leftarrow \mathbf{1}$    ▷ Denotes not used in input tokens
    $\mathbf{x}'[j][\mathbf{tm}] \leftarrow \mathbf{1}$    ▷ Denotes not used in target tokens
end for
$\mathbf{x}'$.update($\mathbf{x}$)    ▷ Replace placeholders with real data with random input/target masks sampled by the Dirichlets
Select 2048 input and target tokens with priority given to $\mathbf{im} = 0$ and $\mathbf{tm} = 0$
No Enc. self-attention and cross-attention when $\mathbf{im} = 1$
Dec. self-attention processes visible same-modal tokens
```

Algorithm A.2 *EgoM2P* Inference

```
Input: input tokens $\mathbf{I}$, target modality $t$, decoding steps $s$, target token number $n$, guidance scale $w$
$\mathbf{T} \leftarrow \mathbf{0}$                                   ▷ Init target tokens prediction
$\mathbf{im} \leftarrow \mathbf{1}$          ▷ Init input mask for $t$. 1: not used as input
$\mathbf{tm} \leftarrow \mathbf{0}$          ▷ Init target mask for $t$. 0: need to predict
for $j = 1$ to $s$ do
    $n_j \leftarrow n/s$          ▷ Num of tokens to decode in this step
    /* Pass 1: conditional distribution prediction */
    context $\leftarrow$ Enc($\{\mathbf{I}, \mathbf{T}[\mathbf{1} - \mathbf{im}]\}$)
    $\mathbf{s}_j \leftarrow$ Sample($n_j, \{i \mid \mathbf{tm}[i] = 0\}$)          ▷ Randomly sample $n_j$ indices from unpredicted tokens
    $\mathbf{p}_{\text{cond}}[\mathbf{s}_j] \leftarrow$ Dec($\mathbf{T}[\mathbf{s}_j]$, context)
    /* Pass 2: unconditional distribution prediction */
    context' $\leftarrow$ Enc($\{\mathbf{T}[\mathbf{1} - \mathbf{im}]\}$)          ▷ Mask all input tokens
    $\mathbf{p}_{\text{uncond}}[\mathbf{s}_j] \leftarrow$ Dec($\mathbf{T}[\mathbf{s}_j]$, context')
    $\mathbf{p}[\mathbf{s}_j] \leftarrow \mathbf{p}_{\text{uncond}} + (\mathbf{p}_{\text{cond}} - \mathbf{p}_{\text{uncond}}) * w$          ▷ CFG
    $\mathbf{T}[\mathbf{s}_j] \leftarrow$ Nucleus Sampling $\sim \mathbf{p}[\mathbf{s}_j]$
    $\mathbf{im}[\mathbf{s}_j] \leftarrow \mathbf{0}$
    $\mathbf{tm}[\mathbf{s}_j] \leftarrow \mathbf{1}$
end for
Return: $\mathbf{T}$                                   ▷ Target tokens prediction
```

B. Ablation Studies

B.1. Number of Visible Tokens

First, we do an ablation study on the maximum number of visible input and target tokens. 4M [8, 76] set this to 256 and argue that "the challenge of the multimodal masked modeling task is mainly determined by how many visible input tokens are used; having fewer tokens makes the task more difficult. This is because the modalities provide a lot of spatial information about each other, so it's important to reduce the number of visible tokens to keep the task challenging enough." However, in our task, each video sample has more than 5000 tokens. We find that increasing the maximum input and target token numbers helps the multimodal masked modeling on videos. We follow 4M to report the validation set loss as metrics to show how well the pretraining is. Refer to Tab. B.1. In order to balance the maximum number of tokens and training efficiency, we choose the maximum number of input/target tokens as 2048.

B.2. Dataset Sampling Weights

Secondly, in our experiment, we observed that the sampling weights assigned to different datasets and modalities significantly impact both training stability and model performance. This is particularly important because our datasets are highly imbalanced; for instance, EgoExo4D [30] contains 160 times more samples than H2O [49]. Training a large model on such skewed datasets can result in two major issues: Smaller datasets may be overlooked, or smaller datasets may suffer from overfitting.

Input/Target Tokens	Avg. Loss
1024	5.80
2048	**4.93**

Table B.1. **Maximum visible token ablation.**

Method	EgoExo4D [30]			ADT [80] (*unseen*)		
	ATE↓	RTE↓	RRE↓	ATE↓	RTE↓	RRE↓
EgoM2P w/o EgoGen	0.028	0.005	0.561	0.053	0.009	0.593
EgoM2P	**0.017**	**0.004**	**0.429**	**0.032**	**0.006**	**0.490**

Table B.2. **Ablation of EgoGen [52] on camera tracking.**

To address these challenges, we found that sampling datasets with probabilities proportional to their sizes helps achieve better balance. To further explore this, we conducted three ablation studies:

1. Sampling datasets based on probabilities proportional to their sizes.
2. Sampling datasets using a uniform distribution.
3. Sampling datasets with probabilities proportional to the logarithm of their sizes.

We observe that across all datasets in our database, the first sampling method, which selects datasets based on probabilities proportional to their sizes, consistently results in the lowest validation loss. For large-scale datasets such as EgoExo4D, the third method, which samples datasets with probabilities proportional to the logarithm of their sizes, achieves a lower validation loss compared to the second method, which employs a uniform distribution. However, both the second and third methods tend to overfit during the early stages of training on smaller-scale datasets.

B.3. EgoGen [52] Contributions

Collecting large-scale egocentric datasets with accurate 3D ground-truth annotations is expensive, time-consuming, and non-scalable. Compared to the internet-scale third-person view video data, EgoGen [52] offers a practical solution to scale up the training data of egocentric foundation models. We perform ablation studies on the contributions of egocentric synthetic data from EgoGen [52]: We remove EgoGen from the training set, fix other settings, and test the model on the same test set. In Tab. B.2 and B.3, we show that cheap and high-quality egocentric synthetic data can boost performance on egocentric camera tracking and egocentric video depth estimation, complementing expensive real data.

C. Tokenization Error Analysis

EgoM2P predicts tokens in a quantized form, which leads to the propagation of quantization errors throughout the

Method	H2O [49] Abs Rel ↓	H2O [49] $\delta_{1.25}$ ↑	HOI4D [60] (*unseen*) Abs Rel ↓	HOI4D [60] (*unseen*) $\delta_{1.25}$ ↑
EgoM2P w/o EgoGen	0.062	94.9	0.067	97.1
EgoM2P	**0.055**	**96.0**	**0.061**	**98.0**

Table B.3. **Ablation of EgoGen [52] on depth estimation.**

	EgoExo4D (cam tracking) ATE↓	RTE↓	RRE↓	EgoExo4D (gaze pred.) MSE↓
EgoM2P	0.017	0.004	0.429	0.0162
Quantization error	0.005	0.001	0.272	0.0000188

Table C.1. **Quantization error analysis.**

pipeline. To examine the impact of tokenizers, we present the reconstruction errors in Table C.1. We compare the quantization error with the *EgoM2P* prediction error in the egocentric camera tracking and gaze estimation tasks. Quantization error is not the bottleneck in our model for Euclidean data. The relatively high rotation error (RRE) highlights challenges of VQ for 4D egocentric data in non-Euclidean space.

D. Post-Training Details

With the evolving quantity of egocentric datasets, *EgoM2P* can be easily adapted to unseen datasets by post-training. In the main paper (Sec. 4.5), we leverage post-training on the training sets of unseen datasets, including ADT [80], HOI4D [60], and ASE [6], and demonstrate that the post-trained *EgoM2P* outperforms SOTA specialist baselines. The ADT training set consists of 10,885 paired RGB and camera trajectory samples, while the HOI4D set includes 11,740 paired RGB and depth video samples. Additionally, the ASE set contains 26,000 paired RGB and depth video samples. All samples are randomly selected from the original dataset, ensuring no overlap with the test set. All modalities are standardized as described in Sec. 3.1. We initialize the post-training with the pretrained *EgoM2P*, and continue to train it for 50B tokens. The number of warmup tokens is 5B, and other settings are the same as Tab. A.2. We observe that post-training on ADT [80] tends to overfit more quickly compared to the other two datasets. Consequently, we select the best model for each task based on its respective validation loss during post-training.

E. Egocentric 4D Reconstruction

Given ground-truth camera intrinsics and an egocentric video, we compare *EgoM2P* with the SOTA baseline MegaSaM [54] for 4D reconstruction. Unlike MegaSaM, which relies on SOTA monocular depth estimators and expensive geometry optimization, *EgoM2P* efficiently reconstructs dynamic egocentric scenes. For a 2-second video at 8 FPS, *EgoM2P* completes the reconstruction in less than 1 second, whereas MegaSaM requires 71 seconds. We provide a qualitative comparison in Fig. E.1.

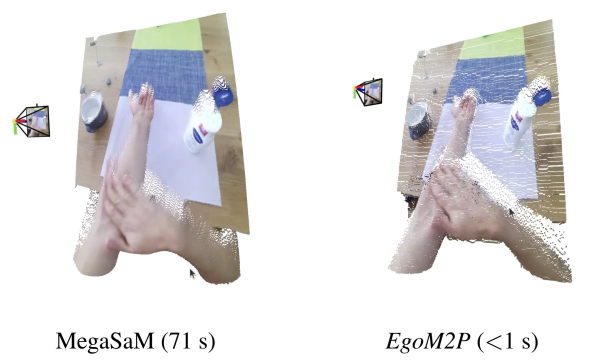

Figure E.1. Dynamic 4D Reconstruction from Monocular Egocentric Videos.