# REWIND: Real-Time Egocentric Whole-Body Motion Diffusion with Exemplar-Based Identity Conditioning

Supplementary Material

## S.1. Video Results

The attached file presents the video results of our main qualitative comparisons shown in Fig. 3-4 in the main paper. In the video, our method is shown to estimate significantly more accurate and natural motions compared to the existing baselines (EgoWholeMocap [16] and EgoPoseFormer [17]).

## S.2. Results with Varying Numbers of Example Poses

In Table S1, we present additional results on exemplar-based identity conditioning with varying numbers of example poses. For our main experiments (Sec. 4.4), we use 10 example poses ($N_{ex} = 10$). We observed that using fewer than 10 example poses ($N_{ex} = 5$) leads to a degradation in motion estimation quality. Conversely, significantly increasing the number of example poses ($N_{ex} = 25$) slightly improves body motion accuracy, but does not enhance hand motion accuracy. Based on these findings, we selected $N_{ex} = 10$ for our main experiments, as it provides a good balance between motion accuracy and the ease of example pose acquisition.

Table S1. **Results with varying numbers of example poses.** Except for the number of example poses ($N_{ex}$), we use the same experimental settings as those in Examplar$^{\dagger}$ from Table 1b in the main paper.

| $N_{ex}$ | MPJPE $_{\text{Body}}$ | PMPJPE $_{\text{Body}}$ | MPJPE $_{\text{Hand}}$ | PMPJPE $_{\text{Hand}}$ |
|---|---|---|---|---|
| 5 | 41.23 | 30.03 | 17.83 | 8.61 |
| 10 | 38.99 | 28.52 | **17.33** | **8.34** |
| 25 | **37.77** | **27.91** | 17.77 | 8.58 |

## S.3. Implementation Details

In this section, we provide additional implementation details of our whole-body motion estimation model.

### S.3.1. Input Encoding

Recall that our network takes as input a sequence of egocentric observations $\Phi^{1:T}$, consisting of stereo images and camera poses, along with a sequence of diffused keypoints $\tilde{\mathbf{J}}_k^{1:T}$ and the corresponding diffusion time $k$. We first describe how each of the conditioning inputs is encoded.

**Egocentric images.** We first estimate 2D keypoints from the egocentric images to encode the geometric information. In particular, we use an EfficientNet-based encoder [13] and a CNN-based decoder [8] to estimate 2D heatmaps. Our encoder consists of four stacks of EfficientNet [13] blocks, each containing three mobile inverted bottlenecks [11] with width multipliers of $[16, 24, 40]$ and depth multipliers of $[1, 2, 2]$, followed by Hard Swish [5] activation. Our decoder consists of three stacks of convolutional blocks, each containing two 2D convolutional layers, followed by batch normalization [6] and ReLU [1] activation.

**Camera poses.** Recall that each camera pose corresponding to viewpoint $v$ is represented by the camera rotation $\mathbf{R}_v \in \mathbb{R}^{3\times3}$ and translation $\mathbf{t}_v \in \mathbb{R}^{3\times1}$. We first convert the camera rotation to a 6D rotation representation [18] and concatenate it with the camera translation vector. We then apply a two-layer MLP, with output feature dimensions set to 256 and 512 for the student and teacher models, respectively. We use Swish [10] activation for the first layer, while the second layer has no activation.

**Diffusion timestep.** We encode the input diffusion timestep based on the sinusoidal functions, as proposed in [3, 4]. We then apply a two-layer MLP with the same network architecture as the camera pose encoder.

**Upper body poses.** Our hand model additionally uses 3D upper body keypoints as conditioning inputs. We flatten the upper body keypoints and apply a two-layer MLP with the same network architecture as the camera pose encoder. Note that we use the ground truth upper body keypoints during training, while during testing, we use the keypoints predicted by the body model.

### S.3.2. Frame Feature Extraction

We now extract frame-wise features by aggregating the conditioning input features. In particular, we concatenate the estimated stereo 2D keypoints with the confidence scores to the corresponding diffused keypoints $\tilde{\mathbf{J}}_k^t$ at each timestep $t$. We additionally concatenate the features of (1) stereo camera poses, (2) the diffusion timestep, and (3) an upper body pose (only for the hand model) to the corresponding diffused keypoints. We then apply Graph Transformer blocks [3], which consist of graph convolution [2] and self-attention [15] layers, on the human skeletal graph. For the teacher network, we use three Graph Transformer blocks, with feature dimensions set to 512 and the number of attention heads set to 4. For the student network, we use a single block with a feature dimension of 256 and 2 attention heads to enable faster inference. Note that we use a linear layer to estimate poses from these intermediate frame-based features to incorporate auxiliary reconstruction loss (to be explained in Sec. S.3.4).

#### S.3.3. Temporal Feature Extraction

Given the frame-based features extracted for each timestep $t$, we apply our Causal Relative-Temporal Transformer (Sec. 3.2) to extract temporal features. For the teacher model, we use three relative-temporal attention layers with 4 attention heads and a window size of 20. For the student model, we use a single relative-temporal attention layer with 2 attention heads with a window size of 8. We set the output feature dimensions to 512 and 256 for the teacher and student models, respectively. Finally, we apply a linear layer to map the output temporal features to the sequence of whole-body keypoints.

#### S.3.4. Network training.

We follow DDPM [4] for training our diffusion model. For the teacher network, we diffuse the ground truth keypoints with a randomly sampled diffusion timestep $k \in [1, K]$ and feed them as inputs to the network. For the student network, the diffusion timestep is set to the maximum value $k = K$ to enable single-step sampling. For noise scheduling, we use cosine scheduling from $\beta_1 = 0.0001$ to $\beta_K = 0.02$ with the maximum diffusion timestep set as $K = 1000$ (refer to [4] for details on the noise scheduling hyperparameter $\beta_k$).

We train our diffusion network for 2M steps using an Adam optimizer with a learning rate of $5 \times 10^{-5}$. We use a single batch consisting of $T = 50$ consecutive frames for training, though our network can inherently generalize to motions longer than the training sequences due to the proposed architecture (Sec. 3.2). For the training loss, we mainly adopt the loss function from MDM [14], which includes: (1) $\mathcal{L}_{simple}$, the L2 distance between the predicted and ground truth motion signals at $k = 0$, (2) $\mathcal{L}_{vel}$, the L2 distance between the predicted and ground truth motion velocities, and (3) $\mathcal{L}_{foot}$, which regularizes the slided foot keypoints (refer to [14] for computation details). We additionally use $\mathcal{L}_{frame}$, an auxiliary L2 loss between the poses predicted from intermediate frame-based features and the ground truth poses. Our final loss function, $\mathcal{L}_{total}$, is defined as:

$$\mathcal{L}_{total} = \mathcal{L}_{simple} + \lambda_{vel}\,\mathcal{L}_{vel} + \lambda_{foot}\,\mathcal{L}_{foot} + \lambda_{frame}\,\mathcal{L}_{frame}. \quad (1)$$

For $\lambda_{vel}$, $\lambda_{foot}$ and $\lambda_{frame}$, we initially use values of 300, 100, and 1, respectively. However, we observe that the loss terms involving motion velocities ($\mathcal{L}_{vel}$ and $\mathcal{L}_{foot}$) converge to very small values in the later stages of training. Thus, we increase the values for $\lambda_{vel}$ and $\lambda_{foot}$ to 4K and 20K, respectively, in the last 10K training steps. Note that, for training the student model, we additionally use the distillation loss $\mathcal{L}_{distill}$ (Sec. 3.3) with the weighting hyperparameter $\lambda_{distill}$ set to 1.

**Network inference.** We use DDIM [12] for network inference, with the number of sampling steps set to 10 for the teacher network and 1 for the student network.

### S.4. Details on Inverse Kinematics

To use our motion tracking results for driving meshes or avatars (e.g., through linear blend skinning), we optionally perform inverse kinematics to convert the estimated 3D keypoints to joint rotations. To this end, we train a simple inverse kinematics network that takes as inputs the 3D whole-body keypoints along with the stereo camera translations (for estimating head poses) per frame and outputs joint rotations. We build our network upon the Graph Transformer [3], similar to the frame-based feature extraction module in our main diffusion model. We use five Graph Transformer blocks [3], with output feature dimensions and the number of attention heads set to 512 and 4, respectively. After the last layer, we use a linear layer to map the final features to the joint rotations in a 6D rotation representation [18]. For network training, we use L2 loss between the predicted and ground truth joint rotations. We train the network with an Adam optimizer and a learning rate of $5 \times 10^{-5}$.

### S.5. Details on Example Pose Estimation

To estimate 3D example poses of the target identity from monocular images, we perform parametric body model fitting to the pseudo ground truth 2D keypoints and depth estimated by Sapiens [7], an off-the-shelf foundational human model. In particular, we fit the parametric body model using the loss $\mathcal{L}_{opt}$ defined as:

$$\mathcal{L}_{opt} = \mathcal{L}_{2D} + \lambda_{depth}\,\mathcal{L}_{depth} + \lambda_{reg}\,\mathcal{L}_{reg} + \lambda_{height}\,\mathcal{L}_{height}. \quad (2)$$

where $\mathcal{L}_{2D}$ is the L2 loss between the 2D projection of the predicted 3D keypoints and the pseudo ground truth 2D keypoints. $\mathcal{L}_{depth}$ is the L2 loss between the predicted and the pseudo ground truth depth maps. $\mathcal{L}_{reg}$ is the L2 loss between the current body model parameters and the mean body model parameters in the training set, penalizing deviations from the mean parameters. We also incorporate $\mathcal{L}_{height}$, which measures the L2 distance between the predicted and the ground truth height of the target identity to reduce the scale ambiguity. We set $\lambda_{depth}$, $\lambda_{reg}$, and $\lambda_{height}$ to 100, 300, and 1, respectively. We perform 3K optimization iterations using the AdamW optimizer [9] with an initial learning rate of $5 \times 10^{-3}$. The learning rate is decayed by 0.023% after each optimization iteration.

#### S.5.1. Details on Baseline Comparisons

**EgoWholeMocap [16].** EgoWholeMocap is the first egocentric whole-body motion estimation method, making it the most relevant baseline for our work. It estimates frame-based 3D poses through 2.5D heatmap estimation and undistortion using the camera parameters, followed by

temporal refinement with an unconditional motion diffusion model, where its DDPM [4]-based motion sampling is guided by the initial 3D poses and their uncertainty scores. In particular, given the clean motion signal $\hat{\mathbf{x}}_0$ estimated by the diffusion model at each diffusion timestep $k$, it defines the mean of the Gaussian distribution for sampling $x_{k-1}$ as:

$$\hat{\mathbf{x}}_0 + \mathbf{w}(\mathbf{x}_e - \hat{\mathbf{x}}_0), \tag{3}$$

where $\mathbf{x}_e$ is a sequence of initially estimated whole-body poses, and $\mathbf{w}$ is a weighting vector computed from their uncertainty scores (refer to Eq. 5 in the original paper [16]).

Note that its original method considers monocular image inputs. To make a fairer comparison to our method, which uses stereo image inputs, we modify the method to (1) estimate 2.5D heatmaps from each of the input stereo images, (2) convert them to 3D poses using the known camera parameters, and (3) perform diffusion-based motion sampling guided by these stereo initial 3D pose estimates by modifying Eq. 3:

$$\hat{\mathbf{x}}_0 + \frac{\mathbf{w}_L}{2}(\mathbf{x}_{e_L} - \hat{\mathbf{x}}_0) + \frac{\mathbf{w}_R}{2}(\mathbf{x}_{e_R} - \hat{\mathbf{x}}_0), \tag{4}$$

where $\mathbf{x}_{e_v}$ and $\mathbf{w}_v$ are the initial poses and uncertainty scores estimated from the input image of viewpoint $v$.

**EgoPoseFormer [17].** EgoPoseFormer is one of the most recently proposed stereo egocentric pose estimation methods. However, it was originally designed to estimate body-only keypoints. To enable comparisons with our method, we modify EgoPoseFormer to estimate whole-body keypoints during both the 2D heatmap and 3D pose estimation stages. Additionally, we incorporate the input camera poses (which are used in our method) by encoding them with an MLP-based encoder and performing feature concatenation in the 3D pose estimation network, similar to our approach.

## References


[1] Abien Fred Agarap. Deep learning using rectified linear units. *CoRR*, abs/1803.08375, 2018. 1

[2] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016. 1

[3] Jia Gong, Lin Geng Foo, Zhipeng Fan, Qiuhong Ke, Hossein Rahmani, and Jun Liu. Diffpose: Toward more reliable 3d pose estimation. In *CVPR*, 2023. 1, 2

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 1, 2, 3

[5] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, 2019. 1

[6] Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. 1

[7] Rawal Khirodkar, Timur Bagautdinov, Julieta Martinez, Su Zhaoen, Austin James, Peter Selednik, Stuart Anderson, and Shunsuke Saito. Sapiens: Foundation for human vision models. In *ECCV*, 2025. 2

[8] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 2015. 1

[9] I Loshchilov. Decoupled weight decay regularization. In *ICLR*, 2019. 2

[10] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *CoRR*, abs/1710.05941, 2017. 1

[11] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 1

[12] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 2

[13] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 1

[14] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H Bermano. Human motion diffusion model. In *ICLR*, 2023. 2

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1

[16] Jian Wang, Zhe Cao, Diogo Luvizon, Lingjie Liu, Kripasindhu Sarkar, Danhang Tang, Thabo Beeler, and Christian Theobalt. Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement. In *CVPR*, 2024. 1, 2, 3

[17] Chenhongyi Yang, Anastasia Tkach, Shreyas Hampali, Linguang Zhang, Elliot J Crowley, and Cem Keskin. Egoposeformer: A simple baseline for egocentric 3d human pose estimation. In *ECCV*, 2024. 1, 3

[18] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 1, 2