Supplementry for *Sketch2Colab*: Sketch-Conditioned Multi-Human Animation via Controllable Flow Distillation

This document accompanies the main paper and provides (i) additional preliminaries and method details referenced as *Supp. Sec. 2* in the main paper, (ii) full experimental setup, training and inference protocols, and metric definitions referenced as *Supp. Sec. 4*, and (iii) additional quantitative results, ablations, and qualitative diagnostics referenced as *Supp. Sec. S.5*.

Qualitative results. For a high-definition qualitative analysis of the generated motions across diverse HOH and HH scenarios, please refer to our project page.

S1. Overview and Notation

We summarize the notation used throughout the supplement. All details below expand the same model introduced in the main paper. They are implementation details of the same modules, not additional model components.

Time, entities, and motion. We use four notions of time. Decoded motion frames use $n \in \{1, \dots, N\}$. Storyboard keyframe locations use $k \in \mathcal{T}_{\text{key}} \subset [0, N]$, with $k = 0$ and $k = N$ denoting the first and last motion frames in the generated N-frame clip. The rectified-flow student uses continuous time $t \in [0, 1]$. The diffusion teacher uses diffusion time $\tau \in [0, 1]$. Whenever a storyboard keyframe location k must index a decoded-frame quantity, we map it to the nearest decoded frame by

$$\nu(k) = 1 + \text{round}\left(\frac{k(N-1)}{N}\right) \in \{1, \dots, N\}. \quad \text{(S1)}$$

We generate an N-frame HOH sequence with H humans and O objects. At frame n the full scene state is $\mathbf{M}_n \in \mathbb{R}^{D_{\text{full}}}$, with humans parameterized by a 22-joint SMPL-X-derived body subset (positions $\mathbf{P}_n^{(h)} \in \mathbb{R}^{J\times 3}$ and 6D rotations $\mathbf{R}_n^{(h)} \in \mathbb{R}^{J\times 6}$) and objects parameterized by rigid poses $\mathbf{O}_n^{(o)} \in \mathbb{R}^7$ (3D translation + quaternion) and K_o designated world-space anchors $\mathbf{Y}_n^{(o)} \in \mathbb{R}^{K_o\times 3}$. These anchors are used by the interaction energies and anchor metrics. We denote the stacked motion as $\mathbf{M}_{1:N} = [\mathbf{M}_1, \dots, \mathbf{M}_N] \in \mathbb{R}^{N\times D_{\text{full}}}$.

Latent space. Following COLLAGE [2], we encode motions into a hierarchical VQ-VAE latent with temporal length $T_{\text{lat}} \le N$ and $V = H + O$ entity tokens. At level $\ell \in \{1, \dots, L\}$ the encoder outputs continuous latents $\mathbf{z}^{(\ell)} \in \mathbb{R}^{T_{\text{lat}}\times V\times d}$ and codebook indices $\mathbf{c}^{(\ell)} \in \{1, \dots, K_\ell\}^{T_{\text{lat}}\times V}$. We use the aggregated continuous latent

$$\mathbf{z} = \sum_{\ell=1}^{L} \mathbf{z}^{(\ell)} \in \mathbb{R}^{T_{\text{lat}}\times V\times d}, \quad \text{(S2)}$$

which is decoded by the frozen decoder $\mathcal{D} : \mathbb{R}^{T_{\text{lat}}\times V\times d} \to \mathbb{R}^{N\times D_{\text{full}}}$.

Storyboard conditions. Storyboard controls are denoted by $\mathcal{C} = (\mathbf{K}_{\text{2D}}, \mathbf{T}_{\text{2D}}, \mathbf{S}_{\text{2D}}, \mathbf{a})$ as in the main paper. For each human h, keyposes are specified at storyboard locations $\mathbf{K}_{\text{2D}}^{(h)}[k] \in \mathbb{R}^{J\times 2}$ for $k \in \mathcal{T}_{\text{key}}$. Joint trajectories are represented by 2D polylines $\mathbf{T}_{\text{2D}}^{(h,j)} \in \mathbb{R}^{L_{h,j}\times 2}$. Object masks are denoted by $\mathbf{S}_{\text{2D}}^{(o)}[k]$ at storyboard locations. We use the same paired 2D and 3D encoders $\{\mathcal{E}_y^{\text{2D}}, \mathcal{E}_y^{\text{3D}}\}_{y\in\{\text{kf},\text{tr},o\}}$ and the same alignment loss $\mathcal{L}_{\text{align}}$ from the main paper. Unless otherwise stated, all notation introduced here is consistent with and extends that of the main paper.

S2. Preliminaries and Method Details (Supp. Sec. 2)

This section provides the additional preliminaries and method details referred to as *Supp. Sec. 2* in the main paper.

S2.1. Rectified Flows and Time Parameterization

We follow flow matching and rectified flow [6, 7] between a noise source and data:

$$\mathbf{z}_1 \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \mathbf{z}_0 \sim p_{\text{data}}. \quad \text{(S3)}$$

We define a straight-line interpolation

$$\mathbf{z}_t = (1-t)\,\mathbf{z}_0 + t\,\mathbf{z}_1, \qquad t \in [0, 1], \quad \text{(S4)}$$

with target velocity

$$v_t = \frac{d\mathbf{z}_t}{dt} = \mathbf{z}_1 - \mathbf{z}_0. \quad \text{(S5)}$$

The rectified-flow student $v_\phi(\mathbf{z}, t \mid \mathcal{C})$ is trained to match this target velocity via the flow-matching objective:

$$\mathcal{L}_{\text{RF}}(\phi) = \mathbb{E}_{t,\mathbf{z}_0,\mathbf{z}_1} \left\| v_\phi(\mathbf{z}_t, t \mid \mathcal{C}) - (\mathbf{z}_1 - \mathbf{z}_0) \right\|_2^2. \quad \text{(S6)}$$

S2.2. Teacher Probability-Flow ODE and RF→Diffusion Time Map

The frozen diffusion teacher follows a VP forward process [4, 9] with noise schedule $\beta(\tau)$ and $\bar{\alpha}_\tau = \exp(-\int_0^\tau \beta(s)\,ds)$. Let $\mathbf{z}_\tau$ be the noised latent at diffusion time τ and $\hat{\epsilon}_\theta(\mathbf{z}_\tau, \tau \mid \mathcal{C})$ the predicted noise. The corresponding probability-flow ODE velocity is

$$v_\theta^{\mathrm{PF}}(\mathbf{z}_\tau, \tau \mid \mathcal{C}) = -\frac{1}{2}\beta(\tau)\left(\mathbf{z}_\tau - \frac{\hat{\epsilon}_\theta(\mathbf{z}_\tau, \tau \mid \mathcal{C})}{\sqrt{1-\bar{\alpha}_\tau}}\right). \quad \text{(S7)}$$

Because the rectified-flow time $t \in [0,1]$ in Eq. (S4) does not coincide with the diffusion time τ in Eq. (S7), we introduce a monotone reparameterization $\tau = \mathcal{T}(t)$ during distillation. We use a quantile-matching map

$$\tau = \mathcal{T}(t) = F_\tau^{-1}(t), \quad \text{(S8)}$$

where F_τ is the CDF of

$$p(\tau) \propto \beta(\tau)\sqrt{1-\bar{\alpha}_\tau}. \quad \text{(S9)}$$

This map is used as a time-alignment device rather than as an exact statewise equivalence between the RF interpolation path and the VP diffusion path.

Because the teacher and student operate in the same latent space, distillation evaluates the teacher PF field at the current student state $\mathbf{z}_t$ and remaps only the time coordinate. We therefore minimize

$$\mathcal{L}_{\mathrm{distill}}(\phi) = \mathbb{E}_{t,\mathbf{z}_t}\left\| v_\phi(\mathbf{z}_t, t \mid \mathcal{C}) - v_\theta^{\mathrm{PF}}(\mathbf{z}_t, \mathcal{T}(t) \mid \mathcal{C})\right\|_2^2. \quad \text{(S10)}$$

S2.3. Latent Space, Decoder, and Distance-Biased Entity Attention

Latent aggregation. As in COLLAGE [2], the per-level latents $\mathbf{z}^{(\ell)} \in \mathbb{R}^{T_{\mathrm{lat}}\times V\times d}$ are combined into a single continuous latent by summing across levels:

$$\mathbf{z} = \sum_{\ell=1}^{L} \mathbf{z}^{(\ell)}. \quad \text{(S11)}$$

The frozen decoder $\mathcal{D} : \mathbb{R}^{T_{\mathrm{lat}}\times V\times d} \to \mathbb{R}^{N\times D_{\mathrm{full}}}$ maps this latent sequence to the full motion $\mathbf{M}_{1:N}$.

Distance-biased entity attention. Both the diffusion teacher and rectified-flow student share the same temporal U-Net backbone with temporal convolutions and local temporal self-attention. We also adopt entity-graph attention with a distance bias. Let $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{T_{\mathrm{lat}}\times V\times d_h}$ denote head-wise projections and let $\mathbf{D}_q \in \mathbb{R}^{V\times V}$ denote the pairwise entity-distance matrix (zero diagonal) at latent step q, computed from cached low-resolution decodes. We apply attention independently at each latent step:

$$\mathrm{Attn}_q = \mathrm{Softmax}\left(\frac{\mathbf{Q}_q\mathbf{K}_q^\top}{\sqrt{d_h}} - \lambda\,\mathbf{D}_q\right)\mathbf{V}_q, \quad \text{(S12)}$$

and stack over the latent temporal axis to obtain the full attention output.

S2.4. Sketch Conditioning Path

The supplement uses the same lift-then-fuse conditioning path as the main paper. The aligned encoders map 2D human keyposes and joint trajectories to 3D proxy controls $(\hat{\mathbf{K}}_{3D}, \hat{\mathbf{T}}_{3D})$. Object masks $\mathbf{S}_{2D}$ are embedded by $\mathcal{E}_o^{2D}$ and are used for object association, scale, and the latent anchor branch. We do not use a separate object-trajectory ControlNet unless otherwise stated.

Conditioning enters the latent U-Net through two routes. First, a trajectory route injects per-level residual sequences $\mathbf{r}_{\mathrm{tr}}^{(\ell)}[q]$ from $\hat{\mathbf{T}}_{3D}$ together with a temporal bias $\mathbf{b}_{\mathrm{tr}}^{(\ell)}[q]$ derived from the path phase $\boldsymbol{\varphi}_q = [s_q, \dot{s}_q, \kappa_q]$. Second, a time-gated keyframe adapter injects localized residuals along the latent temporal axis:

$$\mathbf{r}_{\mathrm{kf}}^{(\ell)}[q] = \sum_{k^\star\in\mathcal{T}_{\mathrm{key}}} \delta_\sigma\left(\frac{q-1}{T_{\mathrm{lat}}-1} - \frac{k^\star}{N}\right)\mathcal{F}_{\mathrm{kf}}^{3D}(\hat{\mathbf{K}}_{3D}[k^\star]), \qquad q \in \{1,\dots,T_{\mathrm{lat}}\}. \quad \text{(S13)}$$

This is the canonical conditioning design used throughout the paper. The energies below refine this conditioned model rather than define a separate control pathway.

S2.5. Dual-Space Guidance, Energies, and Gradient Routing

We now expand the dual-space guidance mechanism described in Secs. 3.2 and 3.3 of the main paper, which combines raw-motion energies and latent anchors to enforce storyboard constraints while remaining on the learned motion manifold.

S2.5.1. Storyboard-Conditioned Motion and Normalization

Let $\Pi(\mathbf{z}) = \mathcal{D}(\mathbf{z})$ denote the decode to motion space, yielding $\mathbf{M}_{1:N} = \{\mathbf{P}_n^{(h)}, \mathbf{R}_n^{(h)}, \mathbf{Y}_n^{(o)}, \mathbf{O}_n^{(o)}\}_{n=1}^N$. We write the 3D world-space position of joint j of human h at frame n as $\mathbf{p}_{n,j}^{(h)} \in \mathbb{R}^3$, and its 2D projection into the sketch camera as $\tilde{\mathbf{p}}_{n,j}^{(h)} = \Pi_{\mathrm{cam}}(\mathbf{p}_{n,j}^{(h)})$. The binary mask $\chi_{n,h,j} \in \{0,1\}$ indicates whether joint (h,j) is constrained at frame n.

Storyboard inputs provide (i) 2D keyposes $\mathbf{K}_{2D}^{(h)}[k,j]$ at annotated keyframe locations $k \in \mathcal{T}_{\mathrm{key}}$, (ii) 2D polyline trajectories $\mathbf{T}_{2D}^{(h,j)}$ with an active interval $[n_{\mathrm{beg}}^{(h,j)}, n_{\mathrm{end}}^{(h,j)}]$, and (iii) 3D proxies $\hat{\mathbf{K}}_{3D}^{(h)}$ and $\hat{\mathbf{T}}_{3D}^{(h,j)}$ obtained from the aligned encoders and the 2D sketches.

To make energy magnitudes stable across clips with different scales, we normalize 3D and 2D distances by per-sequence reference scales:

- s_{3D}: mean bone length over a fixed subset of joints
- s_{2D}: diagonal length of the tight bounding box covering all annotated 2D joints and stroke vertices in the storyboard

All 3D distances are divided by s_{3D} and all 2D distances by s_{2D}. We also normalize energy sums by the number of active elements, so energies are comparable across varying constraint counts.

S2.5.2. Gating Functions and Polyline Distance

Storyboard supervision is inherently sparse and noisy. We modulate raw-space penalties with time- and quality-dependent gates.

Time gate. A keyframe time gate $g_{\text{time}}(n) \in [0,1]$ concentrates 2D penalties near annotated keyframes:

$$g_{\text{time}}(n) = \max_{k \in \mathcal{T}_{\text{key}}} \exp\left(-\frac{\left(\frac{n-1}{N-1} - \frac{k}{N}\right)^2}{2\sigma_k^2} \right), \qquad \sigma_k = \frac{3}{N}. \tag{S14}$$

Path gate. For each joint (h,j) with a trajectory active between $n_{\text{beg}}^{(h,j)}$ and $n_{\text{end}}^{(h,j)}$, we define a smooth path gate

$$g_{\text{path}}(n,h,j) = \sigma\left(\frac{n - n_{\text{beg}}^{(h,j)}}{\sigma_n}\right) \sigma\left(\frac{n_{\text{end}}^{(h,j)} - n}{\sigma_n}\right), \quad \sigma_n = 2, \tag{S15}$$

where $\sigma(\cdot)$ is the logistic function.

Quality gate. To downweight unreliable strokes and lift artifacts, we define a keyframe-quality signal from 2D to 3D to 2D cycles and a trajectory-quality signal from 3D smoothness. Let $\hat{\mathbf{p}}_{n,j}^{(h)}$ denote the lifted 3D joint from the encoders, and let Π_{cam} be the sketch camera projection. For keyframes we define

$$e_{k,h,j}^{\text{cyc}} = \frac{\left\| \Pi_{\text{cam}}(\hat{\mathbf{p}}_{\nu(k),j}^{(h)}) - \mathbf{K}_{\text{2D}}^{(h)}[k,j] \right\|_2}{s_{\text{2D}}},$$
$$g_{\text{qual,key}}(k,h,j) = \exp\left(-\frac{(e_{k,h,j}^{\text{cyc}})^2}{2\sigma_{\text{cyc}}^2} \right). \tag{S16}$$

For trajectories we define

$$e_{n,h,j}^{\text{lift}} = \frac{\left\| \hat{\mathbf{p}}_{n,j}^{(h)} - \text{SG}_7(\hat{\mathbf{p}}_{\cdot,j}^{(h)})[n] \right\|_2}{s_{\text{3D}}},$$
$$g_{\text{qual,tr}}(n,h,j) = \exp\left(-\frac{(e_{n,h,j}^{\text{lift}})^2}{2\sigma_{\text{lift3D}}^2} \right). \tag{S17}$$

where SG_7 is a 3D Savitzky–Golay filter (window 7, order 2) applied along time. We use $\sigma_{\text{cyc}} = \sigma_{\text{lift3D}} = 0.02$. If lift quality is unavailable we set the corresponding quality gate to 1.

Effective 2D weights. We modulate 2D keyframe and trajectory penalties as

$$\lambda_{\text{key}}^{\text{2D}}(k,h,j) = \bar{\lambda}_{\text{key}}^{\text{2D}}\, g_{\text{time}}(\nu(k))\, g_{\text{qual,key}}(k,h,j), \tag{S18}$$
$$\lambda_{\text{tr}}^{\text{2D}}(n,h,j) = \bar{\lambda}_{\text{tr}}^{\text{2D}}\, g_{\text{path}}(n,h,j)\, g_{\text{qual,tr}}(n,h,j), \tag{S19}$$

with $\bar{\lambda}_{\text{key}}^{\text{2D}} = \bar{\lambda}_{\text{tr}}^{\text{2D}} = 1$ unless stated otherwise.

Polyline distance. For a polyline $\mathcal{P} = \{(\mathbf{a}_s, \mathbf{b}_s)\}_{s=1}^{S}$, we define a soft closest-point distance with a log-sum-exp over segments. We normalize by s_{2D} and the polyline arc length $L_{h,j} = \sum_s \|\bar{\mathbf{b}}_s - \bar{\mathbf{a}}_s\|_2$ with $\bar{\mathbf{x}} = \mathbf{x}/s_{\text{2D}}$:

$$d_s(\mathbf{x}) = \frac{1}{L_{h,j}} \left\| \bar{\mathbf{x}} - \text{Proj}_{[\bar{\mathbf{a}}_s, \bar{\mathbf{b}}_s]}(\bar{\mathbf{x}}) \right\|_2, \tag{S20}$$

$$\text{dist}(\mathbf{x}, \mathcal{P}) = -\tau_{\text{seg}} \log \sum_{s=1}^{S} \exp\Big(-\frac{d_s(\mathbf{x})}{\tau_{\text{seg}}} \Big), \tag{S21}$$

with $\tau_{\text{seg}} = 0.02$ and $\text{Proj}_{[\mathbf{a},\mathbf{b}]}(\mathbf{x})$ the projection of $\mathbf{x}$ onto the segment $[\mathbf{a},\mathbf{b}]$. This yields smooth gradients with respect to both $\mathbf{x}$ and the polyline vertices.

S2.5.3. Keyframe and Trajectory Energies

We combine 2D storyboard fidelity and 3D proxy matching for keyframes and trajectories. All energies are normalized by the number of active elements (denominator $Z_\cdot$), so that magnitudes remain stable across clips.

Keyframes. The 2D keyframe energy is

$$E_{\text{key}}^{\text{2D}} = \frac{1}{Z_{\text{key}}^{\text{2D}}} \sum_{k \in \mathcal{T}_{\text{key}}} \sum_{h,j} \lambda_{\text{key}}^{\text{2D}}(k,h,j)\, \chi_{\nu(k),h,j} \times \left\| \frac{\tilde{\mathbf{p}}_{\nu(k),j}^{(h)} - \mathbf{K}_{\text{2D}}^{(h)}[k,j]}{s_{\text{2D}}} \right\|_2^2. \tag{S22}$$

The 3D keyframe energy aligns to the lifted proxies:

$$E_{\text{key}}^{\text{3D}} = \frac{\lambda_{\text{key}}^{\text{3D}}}{Z_{\text{key}}^{\text{3D}}} \sum_{k \in \mathcal{T}_{\text{key}}} \sum_{h,j} \chi_{\nu(k),h,j} \times \left\| \frac{\mathbf{p}_{\nu(k),j}^{(h)} - \hat{\mathbf{K}}_{\text{3D}}^{(h)}[k,j]}{s_{\text{3D}}} \right\|_2^2, \tag{S23}$$

with $\lambda_{\text{key}}^{\text{3D}} = 1$ unless otherwise stated.

Trajectories. The 2D trajectory energy uses the polyline distance in the sketch plane:

$$E_{\text{tr}}^{\text{2D}} = \frac{1}{Z_{\text{tr}}^{\text{2D}}} \sum_{n,h,j} \lambda_{\text{tr}}^{\text{2D}}(n,h,j)\, \chi_{n,h,j}\, \text{dist}\big(\tilde{\mathbf{p}}_{n,j}^{(h)}, \mathbf{T}_{\text{2D}}^{(h,j)}\big)^2. \tag{S24}$$

The 3D trajectory energy aligns joints to the lifted proxy paths:

$$E_{\text{tr}}^{\text{3D}} = \frac{\lambda_{\text{tr}}^{\text{3D}}}{Z_{\text{tr}}^{\text{3D}}} \sum_{n,h,j} \chi_{n,h,j}\, \text{dist}\Big(\frac{\mathbf{p}_{n,j}^{(h)}}{s_{\text{3D}}}, \frac{\hat{\mathbf{T}}_{\text{3D}}^{(h,j)}}{s_{\text{3D}}}\Big)^2, \tag{S25}$$

with $\lambda_{\text{tr}}^{\text{3D}} = 1$ by default. We set $E_{\text{key}} = E_{\text{key}}^{\text{2D}} + E_{\text{key}}^{\text{3D}}$ and $E_{\text{tr}} = E_{\text{tr}}^{\text{2D}} + E_{\text{tr}}^{\text{3D}}$.

S2.5.4. Interaction, Penetration, and Physics Energies

We describe interaction energies for contact and spacing, penetration penalties via an SDF, and simple physics energies.

Interaction, contact, and spacing. Let $\mathcal{U}_{\text{cnt}}$ denote the set of designated contact pairs, such as hand-anchor, hand-hand, or hand-object points. Let $\mathcal{U}_{\text{sp}}$ denote designated spacing pairs used to discourage premature crowding before intended contact. Let $d_{n,uv}$ denote the signed distance (in meters) between points u and v in 3D, optionally measured along an SDF for object surfaces. We normalize $d_{n,uv}$ by s_{3D}. For contact pairs we penalize deviations from a contact margin m_{cnt}, while for spacing pairs we penalize distances that fall below a spacing margin m_{sp}:

$$\psi_\delta(r) = \begin{cases} r^2/(2\delta), & |r| \le \delta, \\ |r| - \delta/2, & \text{otherwise}, \end{cases} \tag{S26}$$

$$E_{\text{cnt}} = \frac{1}{Z_{\text{cnt}}} \sum_n \sum_{(u,v)\in\mathcal{U}_{\text{cnt}}} \psi_\delta\left(\frac{d_{n,uv}}{s_{\text{3D}}} - m_{\text{cnt}}\right), \tag{S27}$$

$$E_{\text{sp}} = \frac{1}{Z_{\text{sp}}} \sum_n \sum_{(u,v)\in\mathcal{U}_{\text{sp}}} \psi_\delta\left(\left[m_{\text{sp}} - \frac{d_{n,uv}}{s_{\text{3D}}}\right]_+\right), \tag{S28}$$

$$E_{\text{int}} = E_{\text{cnt}} + \lambda_{\text{sp}} E_{\text{sp}}, \tag{S29}$$

with $m_{\text{cnt}} = 0.01$, $m_{\text{sp}} = 0.04$, $\delta = 0.02$, and $\lambda_{\text{sp}} = 0.5$ unless otherwise stated.

Penetration. We reuse the SDF-based metric as a differentiable energy and normalize penetration by the characteristic 3D scale s_{3D}. For each time step n we sample surface points for both humans and objects and transform them into world space, yielding $\mathcal{S}_{\text{all}}^n$ and $\mathcal{S}_{\text{body}}^{(1),n}, \mathcal{S}_{\text{body}}^{(2),n}$. The energy is

$$E_{\text{pen}} = \frac{1}{Z_{\text{pen}}} \sum_n \Bigg[\frac{1}{s_{\text{3D}}^2} \sum_{p\in\mathcal{S}_{\text{all}}^n} \left[-D_{\text{scene}}(\mathbf{M}_n[p]) \right]_+^2 + \frac{\lambda_{\text{hh}}}{s_{\text{3D}}^2} \Bigg(\sum_{p\in\mathcal{S}_{\text{body}}^{(1),n}} \left[-D_{\text{body}}^{(2)}(\mathbf{M}_n[p]) \right]_+^2 + \sum_{p\in\mathcal{S}_{\text{body}}^{(2),n}} \left[-D_{\text{body}}^{(1)}(\mathbf{M}_n[p]) \right]_+^2 \Bigg) \Bigg]. \tag{S30}$$

where $[x]_+ = \max(x, 0)$, D_{scene} is the scene SDF (ground and static objects), and $D_{\text{body}}^{(h)}$ is the SDF of human h. The normalization Z_{pen} is the total number of samples over time and surfaces, so E_{pen} corresponds to the average squared relative penetration depth.

Foot skating. We detect stance phases based on normalized height and vertical velocity, and penalize horizontal velocity only during stance. Let $\tilde{z}_{n,f} = z_{n,f}/s_{\text{3D}}$ and $\tilde{v}_{n,f}^z = v_{n,f}^z/(s_{\text{3D}}/\text{frame})$, $\tilde{v}_{n,f}^{xy} = v_{n,f}^{xy}/(s_{\text{3D}}/\text{frame})$ denote normalized height and velocities. We define

$$E_{\text{foot}} = \frac{1}{Z_{\text{foot}}} \sum_n \sum_{f\in\{\text{L/R heel,toe}\}} \mathbb{1}\left[\tilde{z}_{n,f} < \epsilon_z \wedge |\tilde{v}_{n,f}^z| < \epsilon_v\right] \left\|\tilde{v}_{n,f}^{xy}\right\|_2^2, \tag{S31}$$

where $\epsilon_z = 0.015$ and $\epsilon_v = 0.05$, and Z_{foot} normalizes by the total number of foot-time samples.

Ground-plane consistency. We softly penalize body joints and designated object support anchors that move below the ground plane $z = 0$. Let $z_{n,a}$ denote the world-space height of support point a at frame n, where a ranges over feet and designated object support anchors. We define

$$E_{\text{gnd}} = \frac{1}{Z_{\text{gnd}}} \sum_n \sum_{a\in\mathcal{A}_{\text{gnd}}} \left[\frac{-z_{n,a}}{s_{\text{3D}}}\right]_+^2, \tag{S32}$$

where $[x]_+ = \max(x, 0)$ and Z_{gnd} normalizes by the total number of support-point and time samples.

Smoothness. We apply a temporal second-difference regularizer on joint positions and rotations:

$$E_{\text{sm}} = \frac{1}{Z_{\text{sm}}} \sum_n \left(\left\| \frac{\Delta^2 \mathbf{P}_n}{s_{\text{3D}}} \right\|_2^2 + \|\Delta^2 \mathbf{R}_n\|_2^2 \right), \tag{S33}$$

where $\Delta^2 \mathbf{X}_n = \mathbf{X}_{n+1} - 2\mathbf{X}_n + \mathbf{X}_{n-1}$.

S2.5.5. Latent Anchors

We define condition-specific 3D storyboard embeddings $\mathbf{s}_y^{\text{3D}} = \mathcal{E}_y^{\text{3D}}(\hat{\mathbf{M}}_{1:N}^{(y)})$ for $y \in \{\text{kf}, \text{tr}, o\}$. For $y \in \{\text{kf}, \text{tr}\}$, $\hat{\mathbf{M}}_{1:N}^{(y)}$ denotes the constrained human joints and frames selected by $\boldsymbol{\chi}$. For $y = o$, $\hat{\mathbf{M}}_{1:N}^{(o)}$ denotes the object anchors and rigid-pose channels associated with the storyboard masks. The main paper introduces latent anchors to keep the flow near the sketch-conditioned motion manifold. There are two related quantities.

Training loss $\mathcal{L}_{\text{lat}}$ (3D targets). We define 3D storyboard embeddings $\mathbf{s}_y^{\text{3D}} = \mathcal{E}_y^{\text{3D}}(\hat{\mathbf{M}}_{1:N})$ for $y \in \{\text{kf}, \text{tr}, o\}$, where $\hat{\mathbf{M}}_{1:N}$ denotes the constrained subset of joints and frames. A lightweight projection head $f_\gamma : \mathbb{R}^{T_{\text{lat}}\times V\times d} \to \mathbb{R}^{d_e}$ with $d_e = 256$ maps the latent to this embedding space. The latent alignment loss in training is

$$\mathcal{L}_{\text{lat}} = \sum_{y\in\{\text{kf},\text{tr},o\}} \Big(\left\| f_\gamma(\mathbf{z}) - \text{sg}[\mathbf{s}_y^{\text{3D}}] \right\|_2^2 + \lambda_{\text{NCE}} \, \text{InfoNCE}(f_\gamma(\mathbf{z}), \mathbf{s}_y^{\text{3D}}) \Big), \tag{S34}$$

where $\text{sg}[\cdot]$ denotes stop-gradient and $\lambda_{\text{NCE}} = 0.1$. InfoNCE uses in-batch negatives with temperature 0.07.

Latent-space energy E_{lat} (2D storyboard targets). At inference we treat latent agreement with 2D storyboard embeddings as an energy term. Let $\mathbf{s}_y^{\text{2D}} = \mathcal{E}_y^{\text{2D}}(\mathcal{C})$ be the 2D storyboard embeddings produced by the 2D encoders. We define

$$E_{\text{lat}}(\mathbf{z}) = \sum_{y\in\{\text{kf},\text{tr},o\}} \left\| f_\gamma(\mathbf{z}) - \mathbf{s}_y^{\text{2D}} \right\|_2^2. \tag{S35}$$

Thus, during training, $\mathcal{L}_{\text{lat}}$ uses 3D embeddings $\mathbf{s}_y^{\text{3D}}$ as targets, while during inference the latent-space potential uses the 2D storyboard embeddings $\mathbf{s}_y^{\text{2D}}$.

S2.5.6. Gradients, Jacobian Surrogate B_ρ, and Guidance

All raw-space energies E_r act on decoded motion, whereas the rectified flow operates in latent space. We therefore approximate the decoder Jacobian transpose $(\partial\mathcal{D}/\partial\mathbf{z})^\top$ by a learned low-rank, block-Toeplitz surrogate B_ρ (see Sec. S3 for training details).

Given a total raw-space energy over decoded motion $\sum_r \lambda_r(t) E_r(\mathbf{M}_{1:N})$, we define

$$u_{\text{raw}}(\mathbf{z},t) = -B_\rho \, \nabla_{\mathbf{M}_{1:N}} \Big(\sum_r \lambda_r(t) E_r \Big), \tag{S36}$$

$$u_{\text{lat}}(\mathbf{z},t) = -\nabla_{\mathbf{z}} E_{\text{lat}}(\mathbf{z}), \tag{S37}$$

and update the latent ODE as

$$\dot{\mathbf{z}} = v_\phi(\mathbf{z}, t \mid \mathcal{C}) + \eta_{\text{raw}}\, u_{\text{raw}}(\mathbf{z},t) + \eta_{\text{lat}}\, u_{\text{lat}}(\mathbf{z},t). \tag{S38}$$

with η_{raw} and η_{lat} small step-size weights. We clip $\|u_{\text{raw}}\|_2 \le 5$ to avoid large jumps when energies spike. Only raw-space energies require routing through B_ρ. The latent anchor term is differentiated directly. At inference we use only these explicit raw-space and latent-space guidance fields. The learned residual potential V_ψ is not included in the sampling drift.

S2.5.7. Total Potential and Lyapunov Loss

Following the main paper, we define a total potential over latent and decoded motion:

$$\mathcal{V}(\mathbf{z},t) = V_\psi(\mathbf{z}) + \sum_r \lambda_r(t) E_r(\mathbf{z}) + \lambda_{\text{lat}} E_{\text{lat}}(\mathbf{z}), \tag{S39}$$

where V_ψ is a learned residual potential trained by energy-equilibrium matching [1, 3, 8], E_r range over keyframe, trajectory, interaction, penetration, foot, ground-plane, and smoothness energies, and $\lambda_r(t)$ are time-dependent weights (described in Sec. S4.3).

The Lyapunov-inspired loss used in training encourages the flow to descend this potential:

$$\mathcal{L}_{\text{Lyap}}(\phi,\psi) = \mathbb{E}_t\Big[\big(\max\{0, \nabla_{\mathbf{z}}\mathcal{V}(\mathbf{z},t)\cdot v_\phi(\mathbf{z},t) + \kappa \|\nabla_{\mathbf{z}}\mathcal{V}(\mathbf{z},t)\|_2^2\} \big)^2 \Big], \tag{S40}$$

with small margin $\kappa > 0$. Gradients of raw-space energies $\nabla_{\mathbf{z}} E_r$ are routed through B_ρ, whereas $\nabla_{\mathbf{z}} E_{\text{lat}}$ is computed directly in latent space. This potential is used only in training through $\mathcal{L}_{\text{Lyap}}$. The inference sampler uses the explicit fields u_{raw} and u_{lat} from the previous subsection.

S2.6. CTMC Phase Scheduling for Discrete Interaction States

States and notation. The deployed model uses a default $S{=}3$ CTMC with composite interaction states {approach or withdraw, contact or release, hold or carry}. For the state-granularity ablation in Sec. S5.6, we additionally evaluate coarser and finer variants with $S \in \{2, 4, 5\}$. Let $\boldsymbol{\pi}_t \in \Delta^{S-1}$ denote the state occupancy at continuous time $t \in [0,1]$, and let $Q_\eta(\mathbf{h}_t) \in \mathbb{R}^{S\times S}$ be a rate matrix parameterized by a small MLP with parameters η. We enforce

$$Q_{ij} \ge 0 \text{ for } i \ne j, \quad Q_{ii} = -\sum_{j\ne i} Q_{ij}, \tag{S41}$$

so each row sums to zero.

Features. At each solver time t we summarize latent and interaction features into $\mathbf{h}_t$:

- Global latent statistics (mean and std over entity tokens)
- Temporal differences of these statistics
- Normalized distances from hands to designated anchors and from feet to ground
- Indicators for whether storyboard constraints suggest likely contact windows

These features are normalized per-dimension across the dataset.

Forward Kolmogorov equation. The CTMC occupancy evolves according to

$$\frac{d\boldsymbol{\pi}_t}{dt} = \boldsymbol{\pi}_t Q_\eta(\mathbf{h}_t). \tag{S42}$$

We discretize this on the same temporal grid as the rectified-flow ODE solver. Let Q be the number of solver steps, with step size $\Delta = 1/Q$. At time $t_q = q/Q$ we use an explicit reverse-time Euler step:

$$\boldsymbol{\pi}_{q-1} \leftarrow \text{Proj}_{\Delta^{S-1}}\big(\boldsymbol{\pi}_q - \Delta\, \boldsymbol{\pi}_q Q_\eta(\mathbf{h}_{t_q})\big), \tag{S43}$$

followed by a projection back to the simplex. Here we index $\boldsymbol{\pi}_q := \boldsymbol{\pi}_{t_q}$ for brevity.

Phase-mixed velocity and energy gating. As in the main paper, we represent the student velocity as a convex combination of phase-specific sub-fields:

$$v_\phi(\mathbf{z}, t \mid \mathcal{C}) = \sum_{s=1}^{S} \pi_t(s)\, v_\phi^{(s)}(\mathbf{z}, t \mid \mathcal{C}), \tag{S44}$$

where the sub-fields share most weights and differ through small adapters.

We also gate contact-sensitive energies based on state occupancy. Let E_{cnt} collect contact and grasp terms, a subset of E_{int}. We rescale it as

$$E_{\text{cnt}} \leftarrow \big(w_{\text{cnt}}\pi_t(\text{contact}) + w_{\text{hold}}\pi_t(\text{hold})\big)\, E_{\text{cnt}}, \tag{S45}$$

with learned nonnegative scalars $w_{\text{cnt}}, w_{\text{hold}} \geq 0$. We stop-gradient through the π_t gates for the first 10k student iterations, then allow full backpropagation.

CTMC loss. The main paper shows only the core Kolmogorov-residual term for brevity. Here we give the full training loss used in practice, which augments that core term with weak pseudo-state supervision and dwell-time regularization:

$$\begin{aligned}\mathcal{L}_{\text{CTMC}} = \frac{1}{Q}\sum_{q=1}^{Q}\left\|\frac{\boldsymbol{\pi}_q - \boldsymbol{\pi}_{q-1}}{\Delta} - \boldsymbol{\pi}_{q-1}Q_\eta(\mathbf{h}_{t_{q-1}})\right\|_2^2 \\ - \lambda_{\text{sup}}\sum_q \langle \tilde{\boldsymbol{\pi}}_{t_q}, \log \boldsymbol{\pi}_{t_q}\rangle \\ + \beta\,\mathrm{Var}_q \Phi_q + \lambda_{\text{stay}} R_{\text{stay}},\end{aligned} \tag{S46}$$

where $\tilde{\boldsymbol{\pi}}_{t_q}$ are soft pseudo-labels for the composite states inferred from contact distances and relative velocities, Φ_q is an entropy-like regularizer on Q_η's spectrum, and R_{stay} encourages minimum dwell times in each state. We use $\lambda_{\text{sup}} = 0.5$, $\beta = 10^{-4}$, $\lambda_{\text{stay}} = 10^{-3}$, and a minimum dwell in ODE-time $t_{\min} = 0.08$.

Update frequency. At inference we integrate the guided ODE with CTMC updates every $s = 3$ steps, so we update $\boldsymbol{\pi}_q$ via Eq. (S43) only every s Heun steps, holding $\boldsymbol{\pi}_q$ fixed between updates. This sub-sampling reduces overhead without noticeably affecting phase calibration.

S3. Jacobian Surrogate Training Details

For completeness, we summarize practical details about B_ρ beyond Sec. S2.5.

Structure. We parameterize B_ρ as a low-rank, block-Toeplitz operator acting from motion space $(N \times D_{\text{full}})$ to latent space $(T_{\text{lat}} \times V \times d)$. Time-wise, B_ρ uses a small support (kernel size 7) with shared weights across entity channels. Channel-wise, we factorize B_ρ into rank-r factors U, V and apply GroupNorm across groups of latent channels.

Training. We train B_ρ to approximate the cotangent action of the true Jacobian using random probes $u \sim \mathcal{N}(0, \sigma_u^2\mathbf{I})$ in motion space:

$$g_{\text{exact}} = \nabla_{\mathbf{z}}\langle u, \mathcal{D}(\mathbf{z})\rangle, \tag{S47}$$

$$\begin{aligned}\mathcal{L}_B = \mathbb{E}_{u,\mathbf{z}}\|B_\rho u - g_{\text{exact}}\|_2^2 + \lambda_{\text{tv}}\mathrm{TV}(B_\rho) \\ + \lambda_{\text{rank}}\|B_\rho - UV^\top\|_F^2,\end{aligned} \tag{S48}$$

where B_ρ has a low-rank factorization $UV^\top$ with rank $r = 16$, block-Toeplitz temporal structure (kernel size 7), and GroupNorm regularization over blocks. We sample 4 cotangent probes per iteration and train B_ρ with AdamW (lr 5×10^{-4}) for 100k iterations. This pretraining is independent of the student and allows fast energy-guided gradients in latent space.

S4. Experimental Setup, Training, and Metrics (Supp. Sec. 4)

This section corresponds to the additional details referenced in Sec. 4 of the main paper: datasets and storyboard synthesis, training and inference setup, loss composition including $\mathcal{L}_{\text{consist}}$, and metric definitions.

S4.1. Datasets and Storyboard Synthesis

Datasets. We consider two interaction datasets:

- **CORE4D** [12], using the real interaction benchmark subset rather than the full synthetic retargeting branch. We follow the 80/5/15 train, val, and test split used in [2]. We additionally utilize 682 hand-drawn storyboard frames, comprising 3-frame annotations for 156 motion sequences and 2-frame hand drawings for 107 motion sequences, distributed across the dataset splits. The hand-drawn sketch dataset will be made available. We use these sketches to augment the model's ability to work in real scenarios. The generations shown in Figure 3 of the paper were produced using only the hand-drawn sketches as the conditioning signal.
- **InterHuman** [5], using the 6,022-motion benchmark configuration adopted by [2]. This differs from the larger corpus statistics reported in the original InterGen paper and matches the benchmark setting used in our comparisons.

Storyboard synthesis from motion. Given a 3D motion clip, we synthesize storyboard controls following and extending SKETCH2ANIM [13]:

1. **Camera and projection.** We adopt the same camera convention as SKETCH2ANIM, a fixed virtual camera with intrinsics chosen so that the character pair roughly fills the frame. All 3D joints are projected to 2D using Π_{cam}.
2. **Keyframe locations.** We detect motion-salient events using (i) peaks in joint velocities, (ii) contact onset and offset events (hands touching or releasing objects, feet entering or exiting stance), and (iii) direction reversal for root translation. We then select K storyboard locations per clip to cover these events, typically $K \in [2, 5]$ depending on clip length. These locations are stored as $k \in \mathcal{T}_{\text{key}} \subset [0, N]$.
3. **2D keyposes.** For each selected storyboard location $k \in \mathcal{T}_{\text{key}}$ we store the projected 2D joint positions $\mathbf{K}_{\text{2D}}[k] \in \mathbb{R}^{(H \cdot J)\times 2}$.
4. **2D joint trajectories.** For end-effectors (hands, feet, head) and selected mid-body joints, we connect pro-

jected joint locations $\tilde{\mathbf{p}}_{n,j}^{(h)}$ over the clip with 2D polylines, subsampled and smoothed to yield $\mathbf{T}_{\text{2D}}^{(h,j)}$.

5. **Polyline perturbations.** To simulate hand-drawn sketches, we add jitter and contour noise: (i) small i.i.d. Gaussian jitter (1 to 3 pixels) at polyline vertices, (ii) low-amplitude Perlin-noise displacement along the normal direction of the polyline, and (iii) slight random thinning of points. These perturbations follow the protocol of [13] but are adjusted to the resolution of our frames. For joint trajectories, we additionally include an arrow denoting the direction of motion. The placement of the arrow is chosen randomly within the middle 70% of the trajectory. We further add minor breakages along the trajectory. We also add joint dislocations by randomly moving joints within a circular radius of 1% to 2% of the vertical sketch size around the original joint.
6. **Object silhouettes and anchors.** For each object, we rasterize a coarse 2D silhouette $\mathbf{S}_{\text{2D}}^{(o)}[k]$ from the object mesh and anchor positions, using the same camera at storyboard locations k. Silhouettes are downsampled to a sketch resolution $H_s \times W_s$ and optionally blurred.

For motion sequences with hand-drawn sketches available, the hand-drawn sketches replace all or a subset of the synthesized keyframes (on average the replacement ratio is 0.78). For these hand-drawn sketches we retain the same conditioning interface as for the synthetic storyboard inputs. Object shape is provided during both training and inference for CORE4D, as in COLLAGE [2].

S4.2. Teacher and Student Training Protocols

Diffusion teacher (frozen). The teacher is a storyboard-conditioned diffusion model in the COLLAGE latent space and decoder. We use AdamW with learning rate 2×10^{-4}, weight decay 0.01, batch size 64, and EMA decay 0.999. We train for 400k iterations with a VP schedule $\beta(\tau)$ linearly increasing from 10^{-4} to 2×10^{-2} and 1000 diffusion steps. After convergence, we freeze the teacher parameters θ and use its PF velocity v_θ^{PF} as the distillation target.

2D and 3D alignment encoders. We train the paired 2D and 3D encoders $\{\mathcal{E}_y^{\text{2D}}, \mathcal{E}_y^{\text{3D}}\}_{y\in\{\text{kf,tr},o\}}$ with the alignment loss $\mathcal{L}_{\text{align}}$ from the main paper using AdamW with lr 10^{-4}, batch size 64, and 100k iterations.

Rectified-flow student. The student shares the COLLAGE temporal U-Net backbone and is trained in two phases, totaling 500k steps, consistent with the main paper (lr 1×10^{-4}, batch size 64):

- **Phase 1 (200k steps).** We optimize $\mathcal{L}_{\text{RF}} + \lambda_{\text{dist}}\mathcal{L}_{\text{distill}}$ with $\lambda_{\text{dist}} = 0.5$ to match the teacher's transport field.
- **Phase 2 (300k steps).** We activate the Lyapunov loss, CTMC loss, latent loss, energy surrogates, and consistency loss, and optimize the full objective.

We use AdamW with lr 1×10^{-4}, batch size 64 in both phases, and maintain an EMA of the student parameters with decay 0.999.

CTMC pretraining. We pretrain the CTMC rate network Q_η for 60k iterations on teacher-encoded ground-truth latents, using the CTMC loss in Sec. S2.6, AdamW with lr 5×10^{-4}, and batch size 64, before jointly training with the student.

S4.3. Loss Composition and Time-Dependent Weights

The main paper defines the total loss as

$$\mathcal{L} = \mathcal{L}_{\text{RF}} + \lambda_{\text{dist}}\mathcal{L}_{\text{distill}} + \lambda_{\text{Lyap}}\mathcal{L}_{\text{Lyap}} + \textstyle\sum_r \lambda_r \mathcal{L}_{E_r} + \lambda_{\text{lat}}\mathcal{L}_{\text{lat}} + \lambda_{\text{CTMC}}\mathcal{L}_{\text{CTMC}} + \lambda_{\text{cons}}\mathcal{L}_{\text{consist}}, \tag{S49}$$

where $\mathcal{L}_{E_r}$ are supervised surrogates for the energy terms, and $\mathcal{L}_{\text{consist}}$ enforces segment-overlap consistency.

Supervised energy surrogates. For each energy E_r we define a supervised training loss $\mathcal{L}_{E_r}$ using ground-truth signals:

- For keyframes and trajectories, we supervise 3D positions and 2D projections against ground-truth motion and synthetic storyboard
- For interaction, we supervise binary contact labels and approximate distances between hand or foot and object anchors
- For penetration, we supervise collision-free behavior using the scene SDF and ground-truth motions

We scale these losses with weights λ_r in Eq. (S49).

Time-dependent energy weights. We schedule the raw-space energy weights $\lambda_r(t)$ over flow time $t \in [0, 1]$ to reduce conflicts between objectives:

$$\lambda_{\text{key}}(t) = \lambda_{\text{key}}^{\max}(1-t)^2, \tag{S50}$$

$$\lambda_{\text{tr}}(t) = \lambda_{\text{tr}}^{\max}\, 4t(1-t), \tag{S51}$$

$$\lambda_{\text{phys}}(t) = \lambda_{\text{phys}}^{\max}\, \sigma(10(t-0.6)), \tag{S52}$$

where "phys" collects physics-related terms like E_{foot}, E_{gnd}, and E_{sm}, and $\sigma(\cdot)$ is the logistic function. Unless otherwise stated we use $(\lambda_{\text{key}}^{\max}, \lambda_{\text{tr}}^{\max}, \lambda_{\text{phys}}^{\max}) = (1.0, 1.0, 0.1)$.

Segment-overlap consistency loss $\mathcal{L}_{\text{consist}}$. To match the main paper, we explicitly define $\mathcal{L}_{\text{consist}}$, inspired by the segment-overlap consistency in COLLAGE [2]. We sample overlapping temporal windows of length L with overlap L_{ov} (e.g., $L = 64$, $L_{\text{ov}} = 32$), encode each window into

latents, run the student to produce decoded windows $\hat{\mathbf{M}}_{1:L}^{(w)}$, and penalize disagreement in the overlap:

$$\mathcal{L}_{\text{consist}} = \frac{1}{Z_{\text{cons}}} \sum_{w \in \mathcal{W}} \sum_{n \in \text{overlap}(w)} \left\| \hat{\mathbf{M}}_n^{(w)} - \hat{\mathbf{M}}_n^{(w+1)} \right\|_2^2, \tag{S53}$$

where $\mathcal{W}$ indexes consecutive windows, and Z_{cons} is the number of overlapping frames. We use $\lambda_{\text{cons}} = 0.1$ in Eq. (S49).

Final hyperparameters. In terms of the scalars in Eq. (S49), we use

$$\lambda_{\text{dist}} = 0.5, \quad \lambda_{\text{Lyap}} = 0.5, \quad \lambda_{\text{CTMC}} = 0.1, \quad \lambda_{\text{lat}} = 0.1, \quad \lambda_{\text{cons}} = 0.1. \tag{S54}$$

Putting these together yields the concrete total loss used in practice:

$$\begin{aligned} \mathcal{L}_{\text{total}} = \mathcal{L}_{\text{RF}} &+ 0.5\,\mathcal{L}_{\text{distill}} + 0.5\,\mathcal{L}_{\text{Lyap}} \\ &+ 0.1\,\mathcal{L}_{\text{CTMC}} + 0.1\,\mathcal{L}_{\text{lat}} \\ &+ 0.1\,\mathcal{L}_{\text{consist}} + \sum_r \lambda_r\,\mathcal{L}_{E_r}. \end{aligned} \tag{S55}$$

S4.4. Inference and Sampling Algorithm

At inference, we sample from the distilled rectified-flow model with dual-space energy guidance and CTMC phase scheduling. We closely follow the description in Sec. 4 of the main paper.

Initialization. We sample an initial latent from the standard Gaussian prior $\mathbf{z}^{(Q)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and initialize the CTMC occupancy as $\boldsymbol{\pi}^{(Q)} = \text{Uniform}(S)$. We fix the number of ODE steps to $Q = 30$ on CORE4D and $Q = 60$ on InterHuman (for longer clips).

Numerical integration. We integrate the latent ODE backward in time from $t = 1$ to $t = 0$ using a second-order Heun solver. Let $t_q = q/Q$ and $\Delta = 1/Q$. At each solver step q we:

1. **CTMC update (every s steps).** Every $s = 3$ solver steps, we update $\boldsymbol{\pi}_q$ with Eq. (S43), holding it constant between updates.
2. **Phase-mixed velocity.** We compute v_ϕ^{mix} via Eq. (S44) using $\boldsymbol{\pi}_q$.
3. **Decode and energies.** We decode $\mathbf{M} = \mathcal{D}(\mathbf{z}^{(q)})$, compute energies $E_r(\mathbf{z}^{(q)})$ and $E_{\text{lat}}(\mathbf{z}^{(q)})$, apply CTMC gating to contact-related energies via Eq. (S45), and compute guidance vectors u_{raw} and u_{lat}.
4. **Heun predictor-corrector.** We perform a predictor step with the combined drift $v_\phi^{\text{mix}} + \eta_{\text{raw}} u_{\text{raw}} + \eta_{\text{lat}} u_{\text{lat}}$, then recompute the drift at the predicted point and average them to update $\mathbf{z}^{(q-1)}$.

An explicit pseudocode summary is given in Algorithm S1.

Algorithm S1 Sampling with RF + CTMC + Dual-Space Guidance (Heun, CTMC every s steps)

Input: storyboard $\mathcal{C}$, steps Q, student v_ϕ, decoder $\mathcal{D}$, Jacobian surrogate B_ρ, CTMC rates Q_η, CTMC stride s
Sample $\mathbf{z}^{(Q)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, set $\boldsymbol{\pi}^{(Q)} \leftarrow \text{Uniform}(S)$
for $q = Q, Q-1, \ldots, 1$ **do**
  $t_q \leftarrow \frac{q}{Q}$, $\Delta \leftarrow \frac{1}{Q}$
  if $q \bmod s = 0$ **then**
    $\boldsymbol{\pi}^{(q-1)} \leftarrow \text{Proj}_{\Delta^{S-1}}\left(\boldsymbol{\pi}^{(q)} - \Delta\, \boldsymbol{\pi}^{(q)} Q_\eta(\mathbf{h}_{t_q})\right)$
  else
    $\boldsymbol{\pi}^{(q-1)} \leftarrow \boldsymbol{\pi}^{(q)}$
  end if
  $v^{\text{mix}} \leftarrow \sum_{s=1}^{S} \pi^{(q)}(s)\, v_\phi^{(s)}(\mathbf{z}^{(q)}, t_q \mid \mathcal{C})$
  $\mathbf{M}^{(q)} \leftarrow \mathcal{D}(\mathbf{z}^{(q)})$
  compute $\{E_r\}$ and E_{lat}, gate contact energies as in Eq. (S45)
  $u_{\text{raw}} \leftarrow -B_\rho \nabla_{\mathbf{M}} \sum_r \lambda_r(t_q) E_r$, $u_{\text{lat}} \leftarrow -\nabla_{\mathbf{z}} E_{\text{lat}}(\mathbf{z}^{(q)})$
  $\tilde{\mathbf{z}} \leftarrow \mathbf{z}^{(q)} - \Delta\left(v^{\text{mix}} + \eta_{\text{raw}} u_{\text{raw}} + \eta_{\text{lat}} u_{\text{lat}}\right)$
  recompute all terms at $\tilde{\mathbf{z}}$, average drifts, update $\mathbf{z}^{(q-1)}$
end for
Return: $\mathbf{M}_{1:N} = \mathcal{D}(\mathbf{z}^{(0)})$

Classifier-free guidance and micro-Langevin. As described in the main paper, we use classifier-free guidance with 10% conditional dropout during training and guidance weight $\omega \in [1.4, 1.8]$ at inference (default $\omega = 1.6$). Optionally, we can apply a short latent-space micro-Langevin refinement (3 steps, step size 10^{-3}) using the teacher score to sharpen details. This refinement is off by default for all reported numbers. The optional trajectory-only 2D projection correction mentioned in the main paper is also disabled for all reported quantitative results unless explicitly noted otherwise.

S4.5. Metrics and Evaluation Protocols

This subsection provides the details of the metrics and protocols referenced in Secs. 4 and 5 of the main paper. Frame-wise metrics such as foot-skate, keypose errors, trajectory errors, object error, anchor error, and penetration are computed per clip and averaged over the test set. Distribution-level metrics such as FID, MM Dist, and R-Precision are computed following the standard dataset-level protocols used in prior work. For FID ↓, foot-skate ↓, and the 2D and 3D keypose and trajectory metrics, we follow the definitions in Sketch2Anim [13].

ObjPos-3D and Anchor-Err ↓ (CORE4D only). ObjPos-3D measures the L2 distance between the generated object root position and the ground-truth object root

over time. Anchor-Err is the L2 distance between generated and ground-truth object anchor positions, averaged over designated anchors and time. These metrics are reported only on CORE4D. They are not applicable to InterHuman, which is a human–human dataset.

Penetration ↓. We approximate penetration by the sum of squared negative SDF values, accounting for collisions with the static scene as well as inter-penetration between the two interacting characters:

$$\text{Penetration} = \frac{1}{Z_{\text{pen}}} \sum_n \Big(\sum_{p \in \mathcal{S}^n_{\text{all}}} [-D_{\text{scene}}(p)]^2_+ + \sum_{p \in \mathcal{S}^{(1),n}_{\text{body}}} [-D^{(2)}_{\text{body}}(p)]^2_+ + \sum_{p \in \mathcal{S}^{(2),n}_{\text{body}}} [-D^{(1)}_{\text{body}}(p)]^2_+ \Big). \tag{S56}$$

Here, the first term penalizes intersections of both bodies and the object against the environment (D_{scene}), while the second term measures intersection between the two human meshes. For CORE4D, D_{scene} includes ground and static object SDFs. For InterHuman, we use a ground-only SDF.

Text-motion alignment: MM Dist ↓ and R-Prec (Top-k) ↑. We follow the exact evaluation protocol used by [2, 10, 13].

Protocols. We follow Sketch2Anim [13] and OmniControl [11] and report results under two protocols:

- **Average protocol.** Errors are averaged over all constrained joints, with conditioning applied to the full set of storyboard constraints
- **Cross protocol.** For each test example we randomly sample one cross-combination of constrained joints for conditioning and then evaluate errors over all joints, following the Cross setting in [11, 13]

CORE4D Cross results and InterHuman metrics are summarized in Supp. Sec. S5. All reported metrics are averaged over three independent training and evaluation runs.

Diagnostics for Fig. 4. FMD denotes Fréchet Motion Distance. Contact F1 is computed from binary contact labels. The reliability diagram uses predicted contact probabilities and reports Expected Calibration Error and Brier score. In Fig. 4(a), error bars denote mean ± std over three runs. In Fig. 4(c,d), solid curves denote the median over test clips and shaded bands denote the interquartile range. The gray band marks the annotated contact window used for the contact-transition analysis.

S4.6. Baseline Architectures and Training Regimes

In the main paper, we compare *Sketch2Colab* against three baselines: RetrievAdapt, Sketch2Anim-INT (adapted), and the **COLLAGE Teacher**. Below we describe their architectures and training setups in more detail.

RetrievAdapt. RetrievAdapt is a hybrid retrieval-plus-diffusion baseline. It first retrieves motion segments from a database and then refines and stitches them with a small conditional diffusion tuner. We encode each HOH sequence with a MotionCLIP-style encoder augmented with object-aware features (relative human-object distances and contact flags), pooling into a 1280-d descriptor. Given a storyboard, we build a proxy motion from the sketches (interpolated keyframes and trajectories), encode it with the same network, retrieve nearest neighbors from a CORE4D+InterHuman index via cosine similarity, and temporally align them using DTW and re-timing. Around each storyboard keyframe, a short retrieved window is passed through a lightweight 1D temporal UNet diffusion tuner operating in motion space, conditioned on the local lifted keypose, sketch trajectories, and contact flags. We run only 5 to 10 DDIM steps to nudge the retrieved motion toward ground-truth supervision. Refined windows are then blended with smooth overlap weights to yield a single, coherent clip, and a 2-layer MLP (hidden size 512) predicts a global velocity reweighting to further reduce foot-skate and minor penetrations. We jointly train the encoder, retrieval head, and diffusion tuner on the union of CORE4D and InterHuman with a triplet loss for retrieval and a standard noise-reconstruction loss plus contact and penetration penalties for refinement, using AdamW (lr 1×10^{-4}, batch 256 and 64) for 300k steps on a single NVIDIA A100 80GB (overall cost ≈ 0.5 GPU-days).

Sketch2Anim-INT (adapted). Sketch2Anim-INT extends the single-character multi-conditional diffusion model of SKETCH2ANIM [13] to multi-human, object-centric HOH. The base generator is a motion diffusion model in a latent space, with a UNet denoiser ϵ_θ that operates on sequences of joint features. We replace the single-human representation with a concatenated multi-entity representation (two humans plus object anchors), and we replicate the trajectory ControlNet and keypose adapter across entities, sharing weights but conditioning on per-entity IDs. Entity interactions are modeled by adding an entity-attention block at each UNet resolution, similar to our teacher, so that trajectories of one human can influence the other's denoising path. Like the original SKETCH2ANIM, 3D keyposes and trajectories are derived from motion during training and used as surrogates for 2D sketches, with a trajectory ControlNet $\mathcal{F}_{\text{tr}}$ and keypose adapter $\mathcal{F}_{\text{kf}}$ injecting conditions into the denoiser. We use the original latent diffusion backbone (frozen) and train only the ControlNet and keypose adapter with the reconstruction, trajectory, and keyframe losses from [13], plus additional soft contact and penetration losses for HOH interactions. For the main table, we train Sketch2Anim-INT on the union of CORE4D and InterHuman so that its human-centric diffusion backbone

can leverage InterHuman's larger pool of multi-person motions. "CORE4D-only" results are obtained by re-training on CORE4D alone. All runs use AdamW (lr 5×10^{-5}, batch 64, EMA 0.999) for 400k steps on a single A100, requiring ≈ 2.5 GPU-days.

COLLAGE Teacher (sketch-conditioned diffusion teacher). The diffusion teacher used throughout this paper is built on the hierarchical VQ-VAE latent space and decoder of COLLAGE [2], while replacing COLLAGE's original text-planning interface with the storyboard conditioning path described in Sec. 3.1 of the main paper. It therefore retains the COLLAGE latent backbone, entity attention, and decoder, but receives storyboard keyposes, joint trajectories, object masks, and optional text through our lift-then-fuse conditioning modules. This is the model reported as *COLLAGE Teacher* in the main tables and the model from which the rectified-flow student is distilled.

***Sketch2Colab* (ours).** *Sketch2Colab* shares the COLLAGE latent backbone but replaces the diffusion sampler with a rectified-flow student with dual-space (latent + raw motion) guidance and CTMC-based phase scheduling, as detailed in Sec. 3 of the main paper and Supp. Sec. S2. For the main comparisons, the rectified-flow student is distilled solely on CORE4D. The joint-training ablation later reports a setting where both the teacher and *Sketch2Colab* are re-trained on the combined CORE4D+InterHuman pool.

S5. Additional Results, Ablations, and Diagnostics (Supp. Sec. S.5)

In this section we unpack the comparison setup used in the main paper and provide additional tables under different training regimes.

For the main CORE4D quantitative table (Table 2) in the main paper, the two non-ours baselines that require substantial learning (*Sketch2Anim-INT* and *RetrievAdapt*) struggle to reach reasonable performance when trained on the relatively small CORE4D training split alone. To give them a more favorable setting, we therefore train their human-specific generation and retrieval components on the *union* of CORE4D and InterHuman, denoted "CORE4D+InterHuman". In contrast, both the **COLLAGE Teacher** and ***Sketch2Colab*** are trained *only on CORE4D* for Table 2, making it an intentionally challenging comparison in which our models see less data yet still outperform the joint-trained baselines.

Below we additionally: (i) report "raw" CORE4D-only training for all baselines in Table S1, showing that their performance further drops without InterHuman augmentation, (ii) present InterHuman results for the same models in Table S2, where we similarly observe a minor drop when baselines are trained on InterHuman alone rather than jointly, and (iii) provide a joint CORE4D+InterHuman training ablation for the teacher and *Sketch2Colab* in Table S4, which yields consistent gains for our method while the teacher still underperforms the CORE4D-only *Sketch2Colab* from Table 2 of the main paper. All models are trained on a single NVIDIA A100 80GB GPU.

S5.1. CORE4D Results: CORE4D-Only Training

Table S1 reports CORE4D test performance when *all* baselines are trained strictly on CORE4D (no InterHuman augmentation). In this setting the numbers for *Sketch2Colab* and the COLLAGE Teacher are identical to those in the main paper Table 2, since they were already CORE4D-only in the main paper, while the two non-ours baselines (RetrievAdapt and Sketch2Anim-INT) degrade noticeably compared to their joint-training setting.

As expected, removing InterHuman from the training data hurts RetrievAdapt and Sketch2Anim-INT most on Traj-3D, anchor error, and penetration, reflecting undercoverage of the HOH interaction manifold when they only see the smaller CORE4D training split. *Sketch2Colab* and the teacher are unaffected, since they are already CORE4D-only in the main paper.

S5.2. InterHuman Results

Table S2 evaluates all methods on the InterHuman test split under a "raw" setting that mirrors the CORE4D-only analysis. RetrievAdapt and Sketch2Anim-INT are trained solely on *InterHuman*, while the COLLAGE Teacher is first evaluated zero-shot after being trained only on CORE4D. For both the teacher and *Sketch2Colab*, we then also report variants trained directly on InterHuman (*InterHuman-only*).

Because InterHuman is a human–human benchmark, we report realism, control, penetration, and text–motion alignment, but not object-root or anchor metrics. As shown in Table S2, the zero-shot COLLAGE Teacher suffers a pronounced domain gap when transferred from CORE4D to InterHuman. Training the teacher directly on InterHuman recovers a substantial portion of this gap but still trails our rectified-flow student. The CORE4D-only *Sketch2Colab* also exhibits a noticeable domain gap on InterHuman, and the zero-shot performance remains below InterHuman-trained Sketch2Anim-INT on all reported metrics. Training *Sketch2Colab* directly on InterHuman yields a clear and consistent improvement across all reported metrics, giving the strongest overall result on this HH benchmark.

S5.3. Joint CORE4D+InterHuman Training Ablation for Teacher and *Sketch2Colab*

Finally, Table S4 reports a joint-training ablation where we re-train both the COLLAGE Teacher and *Sketch2Colab* on the combined CORE4D+InterHuman corpus and evaluate

Table S1. **CORE4D test (Average protocol), "raw" CORE4D-only training.** All methods are trained only on CORE4D and evaluated on the CORE4D test split. In the main paper (Table 2), RetrievAdapt and Sketch2Anim-INT were trained on CORE4D+InterHuman. Here we show that training them on CORE4D alone leads to a further drop, especially on control and interaction metrics. *Sketch2Colab* and the COLLAGE Teacher are CORE4D-only in both tables.

Method (Training = CORE4D)	FID ↓	Foot-skate ↓	Key-2D ↓	Key-3D ↓	Traj-2D ↓	Traj-3D ↓	ObjPos-3D ↓	Anchor-Err ↓	Penetration ↓	MM Dist ↓	R-Prec (Top-3) ↑
RetrievAdapt (CORE4D-only)	0.525	**0.079**	0.059	0.079	0.298	0.412	0.095	0.128	**0.021**	6.50	0.456
Sketch2Anim-INT (CORE4D-only)	0.879	0.137	0.058	0.078	0.160	0.270	0.073	0.105	0.049	6.50	0.459
COLLAGE Teacher (CORE4D-only, main)	0.511	0.111	0.051	0.063	0.140	0.194	0.062	0.077	0.028	6.08	0.500
***Sketch2Colab* (CORE4D-only, main)**	**0.399**	0.084	**0.032**	**0.043**	**0.078**	**0.116**	**0.034**	**0.038**	0.022	**5.50**	**0.522**

Table S2. **InterHuman test (Average protocol), "raw" training.** RetrievAdapt and Sketch2Anim-INT are trained on InterHuman only. The COLLAGE Teacher is trained either on CORE4D (zero-shot) or on InterHuman only. For *Sketch2Colab* we report both a CORE4D-only zero-shot model and an InterHuman-only model. Because InterHuman is human–human only, object-root and anchor metrics are omitted.

Method	Train set	FID ↓	Foot-skate ↓	Key-2D ↓	Key-3D ↓	Traj-2D ↓	Traj-3D ↓	Penetration ↓	MM Dist ↓	R-Prec (Top-3) ↑
RetrievAdapt	InterHuman	0.611	**0.087**	0.066	0.084	0.341	0.468	**0.024**	6.783	0.451
Sketch2Anim-INT	InterHuman	0.931	0.140	0.062	0.083	0.168	0.283	0.049	6.656	0.456
COLLAGE Teacher (zero-shot)	CORE4D	1.183	0.167	0.072	0.094	0.204	0.336	0.061	7.503	0.392
COLLAGE Teacher (InterHuman-only)	InterHuman	0.786	0.129	0.059	0.079	0.149	0.249	0.045	6.971	0.448
Sketch2Colab (CORE4D-only, zero-shot)	CORE4D	0.968	0.155	0.065	0.088	0.179	0.297	0.051	7.041	0.428
***Sketch2Colab* (InterHuman-only)**	InterHuman	**0.563**	0.109	**0.044**	**0.058**	**0.103**	**0.164**	0.031	**6.187**	**0.472**

Table S3. **CORE4D test (Cross protocol).** All methods are trained as in the main table (RetrievAdapt and Sketch2Anim-INT on CORE4D+InterHuman, teacher and *Sketch2Colab* on CORE4D-only) and evaluated under the Cross protocol (one randomly sampled cross-combination of joints per test example).

Method	FID ↓	Foot-skate ↓	Key-2D ↓	Key-3D ↓	Traj-2D ↓	Traj-3D ↓	ObjPos-3D ↓	Anchor-Err ↓	Penetration ↓
RetrievAdapt	0.549	**0.085**	0.065	0.085	0.325	0.458	0.100	0.150	**0.023**
Sketch2Anim-INT	0.835	0.129	0.062	0.082	0.163	0.265	0.075	0.105	0.049
COLLAGE Teacher	0.658	0.134	0.059	0.078	0.142	0.254	0.067	0.094	0.036
***Sketch2Colab* (full)**	**0.512**	0.102	**0.041**	**0.054**	**0.098**	**0.152**	**0.046**	**0.051**	0.028

Table S4. **Joint training ablation on CORE4D (Average protocol).** Teacher and *Sketch2Colab* are trained either on CORE4D alone (as in the main paper) or jointly on CORE4D+InterHuman ("Joint"). We report CORE4D test metrics. Joint training improves both models, but the joint-trained teacher remains below the CORE4D-only *Sketch2Colab* from Table 2 of the main paper.

Method	Train set	FID ↓	Foot-skate ↓	Key-2D ↓	Key-3D ↓	Traj-2D ↓	Traj-3D ↓	ObjPos-3D ↓	Anchor-Err ↓	Penetration ↓	MM Dist ↓	R-Prec (Top-3) ↑
COLLAGE Teacher (main)	CORE4D	0.511	0.111	0.051	0.063	0.140	0.194	0.062	0.077	0.028	6.080	0.500
COLLAGE Teacher (Joint)	CORE4D+InterH.	0.489	0.106	0.040	0.054	0.095	0.171	0.060	0.074	0.025	6.059	0.509
Sketch2Colab (main)	CORE4D	0.399	0.084	0.032	0.043	0.078	0.116	0.034	0.038	0.022	5.500	0.522
***Sketch2Colab* (Joint)**	CORE4D+InterH.	**0.368**	**0.076**	**0.028**	**0.036**	**0.065**	**0.102**	**0.033**	**0.037**	**0.019**	**5.083**	**0.549**

Table S5. **Ablations of *Sketch2Colab* on CORE4D (Average protocol).** Removing energy guidance, CTMC scheduling, COLLAGE grounding, temporal bias, or using parallel ControlNets or trajectory-only conditioning degrades realism, control, and interaction to varying degrees.

Variant	Realism		Control Accuracy				Interaction			Text–Motion	
	FID ↓	Foot-skate ↓	Key-2D ↓	Key-3D ↓	Traj-2D ↓	Traj-3D ↓	ObjPos-3D ↓	Anchor-Err ↓	Penetration ↓	MM Dist ↓	R-Prec (Top-3) ↑
Sketch2Colab w/o Energy	0.471	0.098	0.035	0.048	0.099	0.149	0.042	0.075	0.026	5.59	0.514
Sketch2Colab w/o CTMC	0.432	0.099	0.034	0.046	0.086	0.132	0.039	0.050	0.027	5.54	0.518
Sketch2Colab w/o COLLAGE grounding	0.420	0.088	0.033	0.044	0.080	0.125	0.042	0.047	0.025	5.53	0.521
Sketch2Colab w/o Temporal bias	0.415	0.088	0.033	0.044	0.082	0.122	0.036	0.043	0.022	5.51	0.520
Sketch2Colab Parallel ControlNets	0.440	0.092	0.034	0.045	0.090	0.128	0.040	0.048	0.023	5.56	0.519
Sketch2Colab Trajectory-only route	0.484	0.097	0.037	0.049	0.104	0.149	0.044	0.061	0.024	5.61	0.514
***Sketch2Colab* (full)**	**0.399**	**0.084**	**0.032**	**0.043**	**0.078**	**0.116**	**0.034**	**0.038**	**0.022**	**5.50**	**0.522**

on the CORE4D test split. We observe consistent gains from joint CORE4D+InterHuman training for both models, with the largest improvements concentrated in control metrics rather than realism. For the COLLAGE Teacher, the clearest gains are in Key-2D, Key-3D, and Traj-2D, while realism gains remain modest. For *Sketch2Colab*, joint training yields stable improvements across realism, control, interaction, and text-motion alignment. Even after joint training, the teacher remains below the CORE4D-only *Sketch2Colab* from Table 2 of the main paper.

The joint-trained *Sketch2Colab* achieves lower FID and trajectory errors than its CORE4D-only counterpart, with

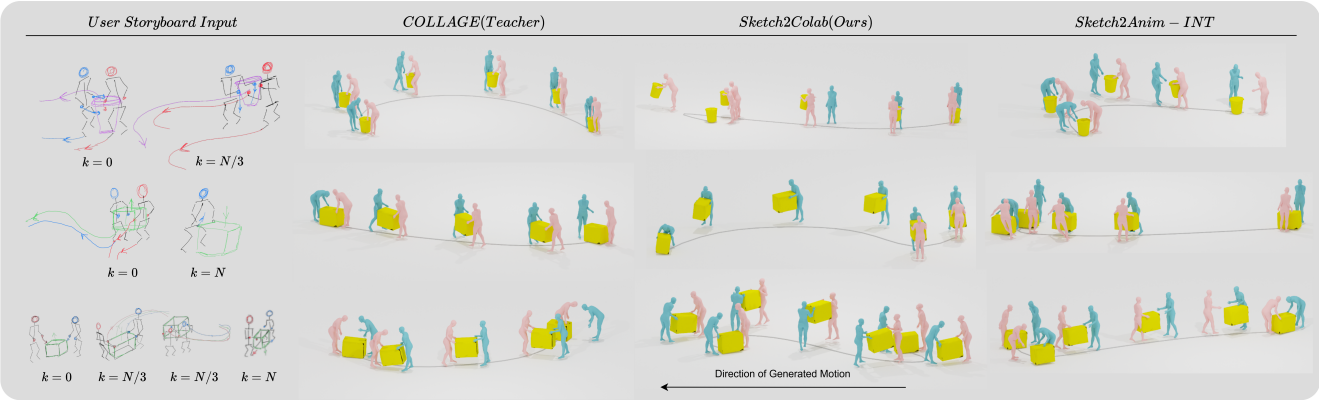

Figure S1. **Sketch→interaction motion: comparison of *Sketch2Colab*, the sketch-conditioned COLLAGE Teacher [2], and Sketch2Anim-INT [13].** From left to right: user storyboard input, COLLAGE Teacher, *Sketch2Colab*, and Sketch2Anim-INT. Given sparse storyboard keyframes and joint trajectories (top), *Sketch2Colab* (third column) generates HOH motions that closely follow the sketches, execute interaction phases at the intended times, and maintain accurate contacts and handoffs. The COLLAGE Teacher (second column) produces globally coherent motion but weakly respects sketched trajectories, keyframe timing, and HOH→OH handoffs where coordination must switch from two-handed to one-handed carrying. Sketch2Anim-INT (right) follows the sketches no better than the teacher in terms of trajectories, and its HOH motion is visibly artifact-prone: the carried object is often not firmly grasped or transported with accurate contacts (see the third row), and interaction quality is poor across all three tasks. Under each sketch, $k \in \mathcal{T}_{\text{key}}$ denotes the storyboard location of the keyframe within the N-frame sequence (e.g., $k = N/3$ specifies that the carry phase should occur around one-third of the motion and guides the behavior until the next keyframe, or until frame N if no later keyframe is provided). Figures were chosen from the best of three generations for the given sketches.

especially clear gains on Key-3D, Traj-2D, Penetration, Foot-skate, and text-motion alignment, while still requiring only 30 to 60 ODE steps at test time.

S5.4. CORE4D Cross-Protocol Results

For completeness, Table S3 reports CORE4D results under the *Cross* protocol, where we follow [11, 13] and, for each test clip, sample a single cross-combination of joints to condition on and evaluate the error over all joints.

Under the Cross protocol, all methods degrade relative to the Average protocol. *Sketch2Colab* preserves the strongest overall ranking, but the drops are not negligible, especially on control and anchor metrics. The harder conditioning regime mainly affects Key-2D, Key-3D, Traj-2D, Traj-3D, ObjPos-3D, and Anchor-Err, while Foot-skate and Penetration remain comparatively more stable.

S5.5. Efficiency, Training Time, and Memory

We summarize inference-time efficiency and training cost in Table S6. All models are trained and evaluated on a single NVIDIA A100 80GB GPU. Training times are approximate wall-clock GPU-days for the CORE4D-only runs used in the main paper. Joint-training runs are $\approx 1.2\times$ longer but follow the same scaling.

Compared to the 1000-step DDPM samplers for Sketch2Anim-INT and the COLLAGE Teacher, *Sketch2Colab* provides a substantial inference speedup. With K=30 Heun steps it is about 10 to 12 times faster than the two DDPM baselines in Table S6, and with K=60 Heun steps it remains about 5 to 6.5 times faster,

Table S6. **Efficiency on CORE4D (120-frame clips).** Average inference time per clip on a single A100 80GB, together with approximate CORE4D-only training cost. For *Sketch2Colab*, the reported training time is the incremental student-side distillation cost only and excludes teacher pretraining, encoder alignment, CTMC pretraining, and Jacobian-surrogate pretraining.

Method	Sampler steps	Inf. time (s)	Train time (GPU-days)
RetrievAdapt (kNN + DDIM tuner)	5–10 (DDIM)	1.8	≈ 0.5
Sketch2Anim-INT (DDPM)	1000	12.3	≈ 2.5
COLLAGE Teacher (DDPM)	1000	14.9	≈ 3.5
Sketch2Colab (Heun, K=30)	30	**1.2**	≈ 2.0 (student only)
Sketch2Colab (Heun, K=60)	60	2.3	same as above

while simultaneously improving control and interaction metrics. RetrievAdapt still records the lowest foot-skate and penetration, benefiting from real captured motion as its retrieval seed and a dedicated velocity-reweighting MLP that further suppresses artifacts, but it lags substantially on control and specified interaction: although its lightweight diffusion tuner can nudge retrieved segments toward storyboard constraints within a 5–10 DDIM-step budget, the refinement is anchored to the nearest-neighbor retrieval and cannot deviate far enough to realise novel compositions or precisely track complex sketch trajectories, especially in the sketch-only setting (Table 1).

S5.6. Ablations and Design Choices

We extend the ablation study in Table 2 of the main paper with a standalone summary in Table S5, using the same CORE4D-only training for the teacher and *Sketch2Colab* as in the main paper. The values mirror the ablation block of

Table S7. **Effect of CTMC state granularity on CORE4D (Average protocol).** We vary the number of CTMC states S and report realism, control, interaction, text–motion, and calibration diagnostics. The default 3-state composite configuration *approach or withdraw, contact or release, hold or carry* corresponds to the full *Sketch2Colab* row in Table 2 of the main paper. Best values are marked using the underlying unrounded means.

	Realism		**Control Accuracy**				**Interaction**			**Text–Mot.**		**Calibration**	
S	FID↓	Foot↓	K-2D↓	K-3D↓	T-2D↓	T-3D↓	Obj↓	Anch↓	Pen↓	MM Dist↓	RP3↑	ECE%↓	Brier↓
No CTMC	0.432	0.099	0.034	0.046	0.086	0.132	0.039	0.050	0.027	5.54	0.518	7.6	0.072
2 (A/C)	0.415	0.092	0.033	0.044	0.082	0.124	0.036	0.044	0.025	5.52	0.520	5.2	0.064
3 (A/C/H)	**0.399**	**0.084**	**0.032**	**0.043**	**0.078**	**0.116**	**0.034**	**0.038**	**0.022**	**5.50**	**0.522**	**4.3**	**0.062**
3 (A/C/H) w/o π-gate	0.420	0.096	0.034	0.045	0.084	0.129	0.038	0.048	0.030	5.53	0.519	6.8	0.069
4 (A/C/H/R)	0.404	0.086	0.032	0.043	0.079	0.118	0.035	0.039	0.022	5.50	0.522	4.5	0.063
5 (finer)	0.411	0.089	0.033	0.044	0.080	0.121	0.035	0.041	0.023	5.51	0.521	4.9	0.064

Table 2 in the main paper.

Removing energy guidance produces the largest degradation in interaction quality (anchor error and penetration roughly double), while disabling CTMC scheduling mainly hurts temporal phasing and contact timing. Replacing the unified dual-space adapters with parallel ControlNets or using trajectory-only conditioning increases trajectory and keyframe errors across the board, confirming that tight coupling between latent anchors, raw-space energies, and discrete phase scheduling is essential for high-fidelity, sketch-conditioned HOH generation.

How does CTMC state granularity affect performance? Table S7 summarizes the sweep over $S \in \{2, 3, 4, 5\}$ CTMC states. The best overall setting is the default $S{=}3$ composite model with *approach or withdraw*, *contact or release*, and *hold or carry*. Using only $S{=}2$ states merges transient contact change with sustained coupled manipulation, which increases slip and flicker during longer carries. Expanding to $S{=}4$ by splitting release from the middle composite state remains competitive but gives little benefit because release is brief in most sequences. Increasing the granularity further to $S{=}5$ over-partitions the interaction timeline and makes state assignments less stable. The row without π-gating shows that the gain is not only from phase-specific sub-fields, but also from using the CTMC occupancy to modulate contact-sensitive energies.

Storyboard→motion examples. Additional panels show diverse HOH interactions (co-manipulation, handoffs, height adjustments) where *Sketch2Colab* tightly follows: (i) sketched hand and body trajectories, (ii) keyframe poses at annotated times, (iii) contact schedules (approach, grasp, handoff, release), and (iv) object placement and alignment. Compared to the COLLAGE Teacher and the Sketch2Anim-INT baseline, *Sketch2Colab* shows reduced jitter, fewer penetrations, and cleaner phase transitions, while also maintaining accurate object contacts and successfully executing HOH→OH handoffs that the baselines consistently miss. In particular, Sketch2Anim-INT exhibits low-quality HOH motion with many artifacts: it roughly understands that the object should be picked up and moved, but often fails to establish stable grasps or to carry the object coherently throughout the sequence (e.g., the block is not properly picked up or transported in the third row), despite a coarse alignment with the sketched trajectories.

Failure modes. Typical failure cases include: (i) long-horizon drift in clips with very sparse mid-sequence keyframes, (ii) degraded performance on unseen object categories with atypical geometry, and (iii) occasional over-smoothing of subtle stylistic nuances when energy weights are set too high late in the trajectory.

S6. Limitations

Limitations. As discussed in the main paper and illustrated in the qualitative examples:

- Object categories are limited by those present in CORE4D and InterHuman. Extending to arbitrary object meshes will require improved SDF construction and anchor definition.
- CTMC scheduling uses relatively simple features. Richer state representations (e.g., learned interaction states) may further improve phase modeling.
- The student relies on a calibrated diffusion teacher. Training a purely flow-based model from scratch without a teacher remains an interesting direction that we plan to explore.

References

[1] Michal Balcerak, Tamaz Amiranashvili, Antonio Terpin, Suprosanna Shit, Lea Bogensperger, Sebastian Kaltenbach, Petros Koumoutsakos, and Bjoern Menze. Energy matching: Unifying flow matching and energy-based models for generative modeling. *arXiv preprint arXiv:2504.10612*, 2025. 5

[2] Divyanshu Daiya, Damon Conover, and Aniket Bera. Collage: Collaborative human-agent interaction generation using hierarchical latent diffusion and language models. In

2025 IEEE International Conference on Robotics and Automation (ICRA), pages 8203–8210. IEEE, 2025. 1, 2, 6, 7, 9, 10, 12

[3] Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 5

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[5] Han Liang, Wenqian Zhang, Wenxuan Li, Jingyi Yu, and Lan Xu. InterGen: Diffusion-based multi-human motion generation under complex interactions. *arXiv*, 2023. 6

[6] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022. 1

[7] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022. 1

[8] Yang Song and Diederik P Kingma. How to train your energy-based models. *arXiv preprint arXiv:2101.03288*, 2021. 5

[9] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020. 2

[10] Guy Tevet, Brian Gordon, Amir Hertz, Amit H Bermano, and Daniel Cohen-Or. Motionclip: Exposing human motion generation to clip space. In *European Conference on Computer Vision*, pages 358–374. Springer, 2022. 9

[11] Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. OmniControl: Control any joint at any time for human motion generation. *arXiv*, 2023. 9, 12

[12] Chengwen Zhang, Yun Liu, Ruofan Xing, Bingda Tang, and Li Yi. Core4d: A 4d human-object-human interaction dataset for collaborative object rearrangement. *arXiv*, 2024. 6

[13] Lei Zhong, Chuan Guo, Yiming Xie, Jiawei Wang, and Changjian Li. Sketch2anim: Towards transferring sketch storyboards into 3d animation. *ACM Transactions on Graphics (TOG)*, 44(4):1–15, 2025. 6, 7, 8, 9, 12