

# FaceCam: Portrait Video Camera Control via Scale-Aware Conditioning

## Supplementary Material

### 6. Overview

In this supplementary material, we first present a video (contains audio) overview of *FaceCam* and its visual results. In Sec. 7, we provide ablation studies on training data generation and proxy head selection. Additional qualitative results are shown in Sec. 8. Implementation details are given in Sec. 9. We include relevant preliminaries in Sec. 10, and discuss limitations and future work in Sec. 11.

### 7. Ablation Study

#### 7.1. Training Data Generation

We provide an ablation study on training data generation in Tab. 3 and Fig. 9 to analyze the impact of each strategy on the final results. We compare three ablated variants and our full model:

- *FaceCam* (w/o Synthetic Camera Motion): applies only Multi-shot Stitching to NeRSemble [30] videos.
- *FaceCam* (w/o Multi-shot Stitching): applies only Synthetic Camera Motion to NeRSemble videos.
- *FaceCam* (w/o In-the-wild Videos): applies both Synthetic Camera Motion and Multi-shot Stitching to NeRSemble videos, without using any in-the-wild videos.
- *FaceCam*: our full model, which adds in-the-wild videos with Synthetic Camera Motion (since multi-view videos are not available) on top of the third baseline.

Through these experiments, we observe that Synthetic Camera Motion enables the model to learn zoom and pan motions and to produce smooth trajectories without sudden camera pose changes. Multi-shot Stitching further teaches the model to follow camera angle changes along the target trajectory, and together these two strategies yield accurate camera control. Incorporating in-the-wild video data improves generalization to diverse real-world lighting conditions and objects, leading to better appearance consistency with the input video.

#### 7.2. Proxy 3D Head Selection

For all experiments reported in the main paper, we use a single generic 3D Gaussian head as a proxy during inference to render videos and extract facial landmarks. This proxy head can be generated by any 3D head generation methods [31, 38]. To verify that the specific choice of proxy head does not influence performance, we select two additional proxy heads (Fig. 8) and evaluate *FaceCam* on in-the-wild videos. The results in Tab. 3 show only minor differences across all three proxies, indicating that our approach is largely insensitive to the particular proxy head. This sup-

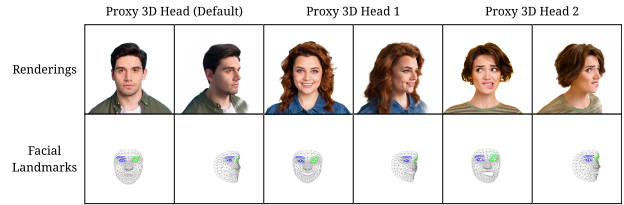


Figure 8. **Different choices of proxy 3D head.** We select two additional proxy 3D heads with different identities and corresponding facial landmark detections, and conduct an ablation study showing that the proxy’s identity and expression do not affect the final generation results.

ports our design choice that the landmarks serve purely as a camera-conditioning signal, rather than conveying identity or expression information. The identity and expression in the generated videos come solely from the source video.

### 8. Experimental Results

We conduct extensive experiments on in-the-wild videos with diverse camera trajectories to assess *FaceCam* under both challenging synthetic settings and realistic use cases. The results are shown in Fig. 10 and Fig. 11. Across a wide range of inputs, the model tracks intricate head and hair dynamics, responds smoothly to varied facial expressions, and respects motion-dependent artifacts such as blur from rapid body movement. When the target trajectory places the virtual camera farther from the subject, *FaceCam* plausibly completes missing regions by synthesizing coherent clothing and background content. On real footage, it recreates studio and streaming scenes with stable identity and layout, while reliably retaining fine-grained accessories and props (e.g., cosmetics, jewelry, headbands, microphones, and glasses). Notably, the same pipeline extends to stylized inputs such as cartoon characters, indicating strong generalization beyond the distribution of the training data.

### 9. Implementation Details

#### 9.1. Training Data Generation

We provide pseudo-code for three training data generation procedures. *Scale and Color Augmentation* (Algorithm 1) is applied to both source and target videos to increase data diversity, including variations in head size and background appearance. *Synthetic Camera Motion* (Algorithm 2) is applied to the target video to create continuous camera trajectories with zoom and pan effects, which is essential for achieving smooth, temporally coherent generated videos.

Table 3. **Ablation study.** We conduct ablation studies to quantify the impact of different training data components on the final performance of our model. We also vary the choice of proxy head and show that this selection has negligible effect on the generated results.

| Method                                       | Camera Correctness | ArcFace Similarity | Imaging Quality | Aesthetic Quality | Subject Consistency | Background Consistency | Motion Smoothness | Dynamic Degree |
|--|--------------------|--------------------|-----------------|-------------------|---------------------|------------------------|-------------------|----------------|
| <i>FaceCam</i> (w/o Synthetic Camera Motion) | 96.00              | 81.19              | 72.03           | 58.02             | 94.27               | 94.91                  | 99.29             | 83.00          |
| <i>FaceCam</i> (w/o Multi-shot Stitching)    | 86.00              | 76.38              | 70.73           | 55.10             | 94.56               | 95.12                  | <b>99.30</b>      | 80.00          |
| <i>FaceCam</i> (w/o In-the-wild Videos)      | <b>100.00</b>      | 77.73              | 70.71           | 55.73             | 94.52               | <b>95.16</b>           | 99.23             | 89.00          |
| <i>FaceCam</i>                               | 97.00              | <b>83.94</b>       | <b>73.49</b>    | <b>59.91</b>      | <b>94.77</b>        | 94.98                  | 99.05             | <b>96.00</b>   |
| <i>FaceCam</i> (Proxy 3D Head 1)             | 97.00              | 84.45              | 73.48           | 59.85             | 94.80               | 94.89                  | 99.02             | 95.00          |
| <i>FaceCam</i> (Proxy 3D Head 2)             | 97.00              | 84.74              | 73.47           | 59.89             | 94.74               | 94.89                  | 99.03             | 94.00          |

#### Algorithm 1 Scale and Color Augmentation

**Require:** Source clip  $V^s = \{I_i^s\}_{i=1}^{T_s}$ , target clip  $V^t = \{I_i^t\}_{i=1}^{T_t}$   
**Ensure:** Augmented clips  $\tilde{V}^s, \tilde{V}^t$

- 1: Sample scale factors  $s^s, s^t \sim \mathcal{U}(0.75, 1.25)$
- 2: Sample a background color  $c \sim \text{UniformColor}()$   $\triangleright$  shared between source and target
- 3: **for**  $i = 1$  to  $T_s$  **do**  $\triangleright$  augment source clip
- 4:  $J_i^s \leftarrow \text{Resize}(I_i^s, s^s)$
- 5:  $M_i^s \leftarrow \text{FaceSeg}(J_i^s)$   $\triangleright M_i^s \in \{0, 1\}^{H \times W}$
- 6:  $B_i^s \leftarrow c \cdot (\mathbf{1} - M_i^s)$
- 7:  $\tilde{I}_i^s \leftarrow M_i^s \odot J_i^s + B_i^s$
- 8: **end for**
- 9:  $\tilde{V}^s \leftarrow \{\tilde{I}_i^s\}_{i=1}^{T_s}$
- 10: **for**  $i = 1$  to  $T_t$  **do**  $\triangleright$  augment target clip with same background color
- 11:  $J_i^t \leftarrow \text{Resize}(I_i^t, s^t)$
- 12:  $M_i^t \leftarrow \text{FaceSeg}(J_i^t)$
- 13:  $B_i^t \leftarrow c \cdot (\mathbf{1} - M_i^t)$
- 14:  $\tilde{I}_i^t \leftarrow M_i^t \odot J_i^t + B_i^t$
- 15: **end for**
- 16:  $\tilde{V}^t \leftarrow \{\tilde{I}_i^t\}_{i=1}^{T_t}$

Finally, *Multi-shot Stitching* (Algorithm 3) is applied to the target video to introduce discrete camera pose changes, enabling the model to handle viewpoint transitions in the generated outputs.

## 9.2. Experimental Details

All experiments are conducted at a resolution of  $704 \times 480$ , with generated videos of length 81 frames. We use the same text prompt for all experiments: “A portrait of a person.” TrajectoryCrafter [57] can generate at most 49 frames in the general setting, and only 29 frames when the first frame of the generated video does not coincide with the first frame of the source video. To ensure a fair comparison, we therefore evaluate on the first 29 frames in the static-camera setting and on the first 49 frames in the dynamic-camera setting for all baselines. ReCamMaster [3] produces camera-controlled results only when the target camera pose for the

#### Algorithm 2 Synthetic Camera Motion (Zoom and Pan)

**Require:** Input clip  $V = \{I_i\}_{i=1}^T$ , motion type  $m \in \{\text{zoom}, \text{pan}\}$ , image resolution  $(H, W)$   
**Ensure:** Motion-augmented clip  $\tilde{V}$

- 1: **if**  $m = \text{zoom}$  **then**
- 2: Sample  $s_{\text{start}}, s_{\text{end}} \sim \mathcal{U}(1.0, 1.25)$
- 3: **for**  $i = 1$  to  $T$  **do**
- 4:  $\alpha \leftarrow \frac{i-1}{\max(T-1, 1)}$
- 5:  $s_i \leftarrow (1 - \alpha) \cdot s_{\text{start}} + \alpha \cdot s_{\text{end}}$
- 6:  $J_i \leftarrow \text{Resize}(I_i, s_i)$
- 7:  $\tilde{I}_i \leftarrow \text{CenterCropOrPad}(J_i, H, W)$
- 8: **end for**
- 9: **else if**  $m = \text{pan}$  **then**
- 10: Choose maximum offset  $\delta_x, \delta_y$  relative to  $(H, W)$
- 11: Sample offsets  $\mathbf{o}_{\text{start}}, \mathbf{o}_{\text{end}} \sim [-\delta_x, \delta_x] \times [-\delta_y, \delta_y]$
- 12: **for**  $i = 1$  to  $T$  **do**
- 13:  $\alpha \leftarrow \frac{i-1}{\max(T-1, 1)}$
- 14:  $\mathbf{o}_i \leftarrow (1 - \alpha) \mathbf{o}_{\text{start}} + \alpha \mathbf{o}_{\text{end}}$
- 15:  $\tilde{I}_i \leftarrow \text{CropOrPadWithOffset}(I_i, \mathbf{o}_i, H, W)$
- 16: **end for**
- 17: **end if**
- 18:  $\tilde{V} \leftarrow \{\tilde{I}_i\}_{i=1}^T$

Table 4. Source and target camera pairs used in experiment on Ava-256 [40].

| ID | Source Camera | Target Camera |
|----|---------------|---------------|
| 1  | cam_400944    | cam_401031    |
| 2  | cam_400944    | cam_401410    |
| 3  | cam_400981    | cam_401045    |
| 4  | cam_400981    | cam_401292    |
| 5  | cam_401163    | cam_401031    |
| 6  | cam_401163    | cam_401458    |
| 7  | cam_401168    | cam_401045    |
| 8  | cam_401168    | cam_401292    |
| 9  | cam_401316    | cam_401410    |
| 10 | cam_401316    | cam_401458    |

first frame has an identity rotation; otherwise, the generated video degenerates to the source video. In the static-



---

**Algorithm 3** Multi-shot Stitching

---

**Require:** Set of clips for a target video  $\mathcal{C} = \{V^{(k)}\}_{k=1}^N$ ,  
 $V^{(k)} = \{I_i^{(k)}\}_{i=1}^{T_k}$ , maximum shots  $K_{\max} = 4$   
**Ensure:** Stitched target clip  $\tilde{V}$

- 1: Sample number of shots  $K \sim \text{Uniform}\{1, 2, \dots, K_{\max}\}$
- 2: Sample  $K$  distinct indices  $\{i_1, \dots, i_K\}$  from  $\{1, \dots, N\}$   $\triangleright$  different camera poses
- 3: Initialize stitched sequence  $\tilde{V} \leftarrow \emptyset$
- 4: **for**  $j = 1$  to  $K$  **do**
- 5:    $V^{(j)} \leftarrow V^{(i_j)}$ , length  $T^{(j)}$
- 6:   Sample start index  $a_j \sim \{1, \dots, T^{(j)} - 1\}$
- 7:   Sample end index  $b_j \sim \{a_j + 1, \dots, T^{(j)}\}$
- 8:   Define segment  $S_j \leftarrow \{I_i^{(j)} \mid i = a_j, \dots, b_j\}$
- 9:    $\tilde{V} \leftarrow \text{Concat}(\tilde{V}, S_j)$
- 10: **end for**
- 11: **return**  $\tilde{V}$

---

camera experiments, we thus enforce an identity rotation as the first-frame camera condition for this baseline to obtain valid results. All baselines are run with their official configurations and released pre-trained weights.

For the static-camera setting on the Ava-256 dataset, we select 10 identities, each with 10 source–target camera pairs, yielding a total of 100 videos. The selected identities are KDA058, XJT672, LAS440, IFG774, EID363, NRE683, PAK800, MCR809, SKB942, KJJ701. The source and target cameras are summarized in Tab. 4.

## 10. Preliminary

### 10.1. Conditional Video Generation

We build our system on the open-source video foundation model Wan [48] for conditional video generation. Wan is a latent video diffusion model comprising a 3D Variational Autoencoder (VAE) [29], a text prompt encoder [9], and two transformer-based diffusion models (DiT) [41] specialized for the high and low noise stages. The model adopts Rectified Flow framework [12] for the noise schedule and denoising process. Detailed architecture and training settings are provided in the supplementary material.

During training, a pretrained 3D VAE encodes a video  $V \in \mathbb{R}^{f \times h \times w \times c}$  into latent space:  $z_0 = \mathcal{E}(V)$ . Then in the forward diffusion process, the DiT injects Gaussian noise  $\epsilon$  into  $z_0$  to create a noisy latent. The forward process is defined as straight paths between data distribution and a standard normal distribution.

$$z_t = (1 - t)z_0 + t\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \quad (9)$$

where  $t$  denotes the iterative timestep.  $z_t$  is then patchified, concatenated with text tokens encoded by the text

prompt encoder and other additional conditioning signals, and fed into DiT blocks. To solve the reverse denoising process, Conditional Flow Matching (CFM) [35] learns a time-dependent velocity field  $v_\theta(z, t, \mathbf{c})$  that defines an ordinary differential equation (ODE):

$$\frac{dz_t}{dt} = v_\theta(z_t, t, \mathbf{c}), \quad t \in [0, 1], \quad (10)$$

transporting samples from the base (standard Gaussian) to the data distribution under conditioning  $\mathbf{c}$ . With the rectified interpolant, the target velocity along the path is constant:

$$u^*(z_t, t | z_0, \epsilon) = \frac{dz_t}{dt} = \epsilon - z_0. \quad (11)$$

CFM trains  $v_\theta$  by regressing to this target with MSE loss:

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}_{t, z_0, \epsilon} \left[ \|v_\theta(z_t, t, \mathbf{c}) - (\epsilon - z_0)\|_2^2 \right]. \quad (12)$$

At inference, we integrate the learned ODE deterministically from noise to data by marching from  $t = 1$  to  $t = 0$ :

$$z_{t-\Delta t} = z_t - \Delta t \, v_\theta(z_t, t, \mathbf{c}), \quad (13)$$

yielding the final latent  $z_0$  consistent with the conditioning  $\mathbf{c}$ .  $z_0$  is then decoded by the pre-trained VAE decoder and outputs the generated video:  $V = \mathcal{D}(z_0)$ .

### 10.2. Wan2.2 and MoE Video Diffusion

Wan2.2 [48] is a family of large-scale latent video diffusion models. It supports multi-modal conditioning (text-to-video, image-to-video, text–image-to-video, and specialized speech/animation variants) and generates high-fidelity videos up to 720p at 24 fps using a high-compression video VAE [29] and a DiT-style [41] diffusion backbone.

To scale model capacity without increasing inference cost, Wan2.2 replaces a single denoising network with a Mixture-of-Experts (MoE) [49] architecture. A set of expert denoisers is specialized for different noise regimes (*e.g.*, high-noise early steps *vs.* low-noise late steps), and a routing scheme based on the diffusion timestep selects which expert to apply at each step. This design enlarges the total parameter count and improves motion, semantic, and aesthetic fidelity, while keeping per-step FLOPs comparable to a dense model.

More generally, an MoE layer consists of a collection of experts  $\{E_k\}_{k=1}^K$  and a gating function  $g(x)$  that selects a sparse subset of experts for each input  $x$ , often via top- $k$  routing. Only the selected experts are evaluated and their outputs are combined, for example

$$y = \sum_{k \in \mathcal{S}(x)} g_k(x) E_k(x), \quad (14)$$

where  $\mathcal{S}(x)$  is a small set of active experts and  $g_k(x)$  are normalized routing weights. By activating only a few experts per input, MoE architectures enable models with billions of parameters to operate at roughly the same compute cost as much smaller dense networks, a property that Wan2.2 leverages to scale video generation quality and controllability.

### 10.3. MediaPipe Facial Landmark Detection.

We use Google’s MediaPipe Face Mesh [37] as an off-the-shelf module to obtain dense 2D/3D facial keypoints from monocular RGB inputs. MediaPipe Face Mesh predicts a set of  $K = 468$  landmarks in real time, even on mobile devices, by applying a lightweight neural network to a cropped face region and regressing per-vertex coordinates that approximate the full facial surface. The model operates on a single RGB camera without requiring depth sensors and is optimized for GPU acceleration, making it suitable for large-scale video processing and interactive applications.

Concretely, given an input frame  $I_i$ , the detector returns a landmark set

$$\mathbf{U}_i = \{\mathbf{u}_{i,k}\}_{k=1}^K, \quad K = 468, \quad (15)$$

where each

$$\mathbf{u}_{i,k} = (x_{i,k}, y_{i,k}, z_{i,k}) \quad (16)$$

encodes normalized image coordinates  $(x_{i,k}, y_{i,k})$  and a relative depth value  $z_{i,k}$ . In practice, MediaPipe adopts a two-stage pipeline: a BlazeFace-style face detector first produces a tight region of interest, and a dedicated mesh regressor then predicts the dense landmark configuration within that region. These landmarks are widely used in AR and avatar applications to recover facial geometry and pose from video streams; in our work, we reuse them as a compact, robust representation for conditioning and camera control.

## 11. Limitations and Future Work

Despite the accurate camera control and high-quality results achieved, *FaceCam* still has several limitations. First, because facial landmarks can only be detected when facial features are visible in the input video, *FaceCam* cannot handle views where the camera rotates to the back of the head. For the same reason, although *FaceCam* can generalize to cartoon characters, it does not extend to general scenes in which a facial landmark detector is inapplicable. Building on the same idea of using image-space correspondences as a camera representation, but redefining how these correspondences are encoded, could help address this limitation. Second, background generation is not the focus of this work, partly due to data limitations. Incorporating synthetic data with multi-view-consistent backgrounds could further improve the model’s ability to synthesize background content

behind the subject. Third, due to the limitations of the underlying video generation model, *FaceCam* remains relatively slow at inference and is not yet suitable for real-time applications. Distilling the model or adopting a more efficient video generation backbone are promising directions.

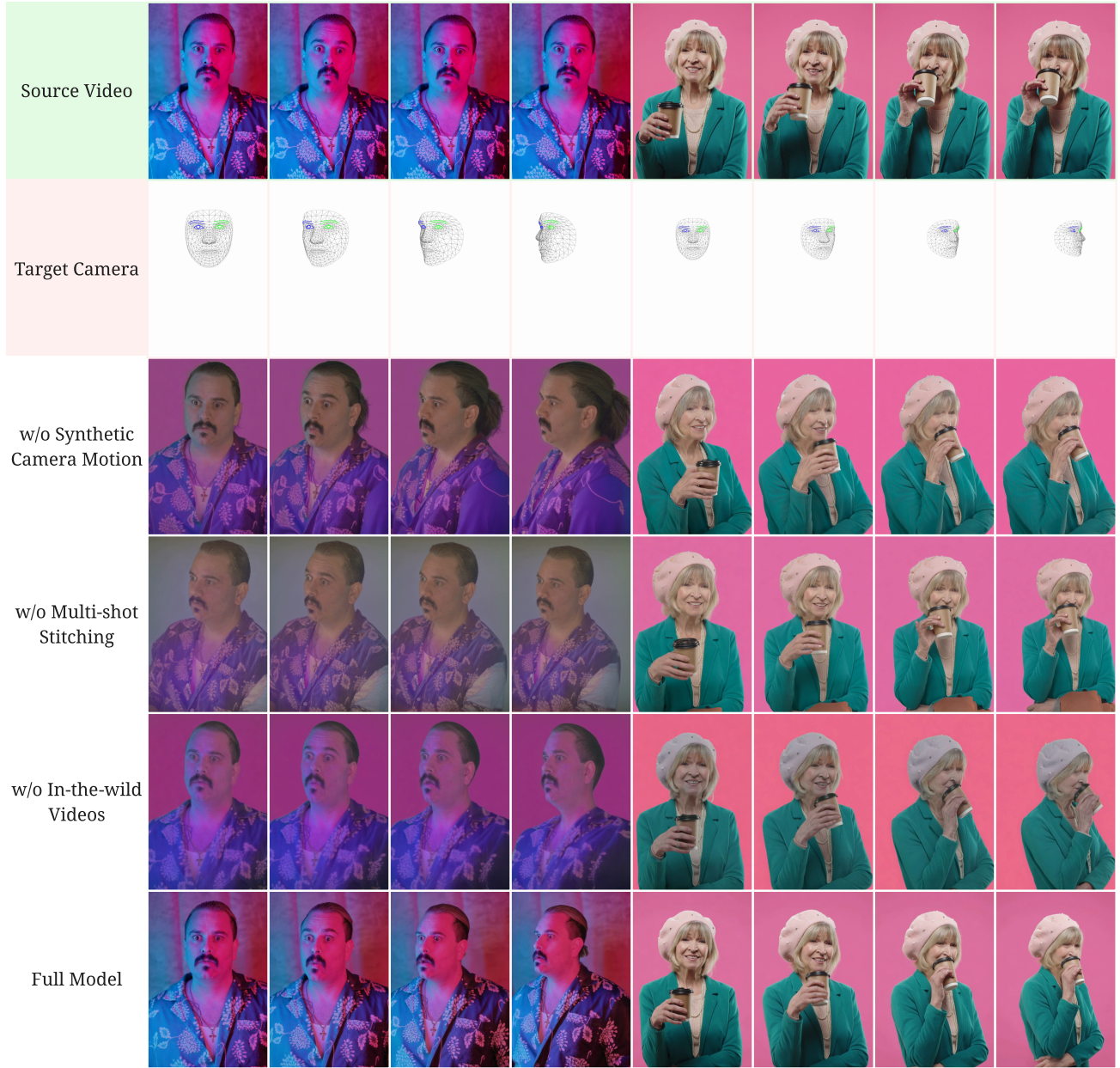


Figure 9. **Ablation study on training data generation.** Without Synthetic Camera Motion, the model often produces inaccurate camera trajectories with discontinuous or abrupt changes. Without Multi-shot Stitching, the model cannot learn to change camera angles along a trajectory. With both strategies applied but without in-the-wild videos (*w/o In-the-wild Videos*), the model generates correct camera motion and angle changes, but the lighting remains tied to the training distribution and fails to generalize to real-world illumination, leading to inconsistencies with the source video. Our full model provides accurate camera control and high image quality with lighting and appearance consistent with the source video.



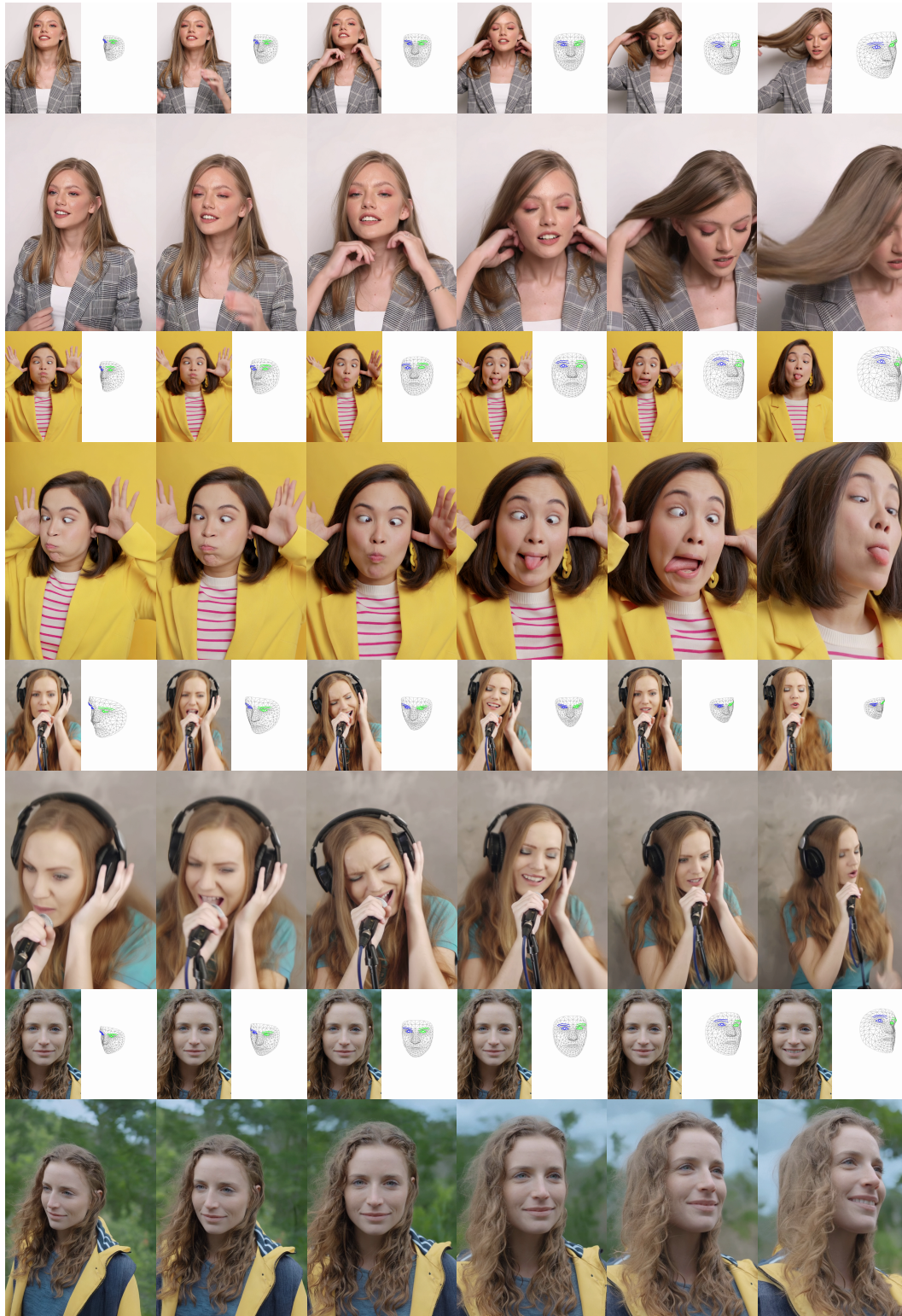


Figure 10. *FaceCam* performs robustly across diverse, challenging scenarios: it retains head and hair motion from the input video (example 1), captures a wide range of facial expressions (example 2), maintains motion-induced blur from fast body movements (example 3), and plausibly outputts clothing and background when the generated video contains a smaller face region than the input (example 4).



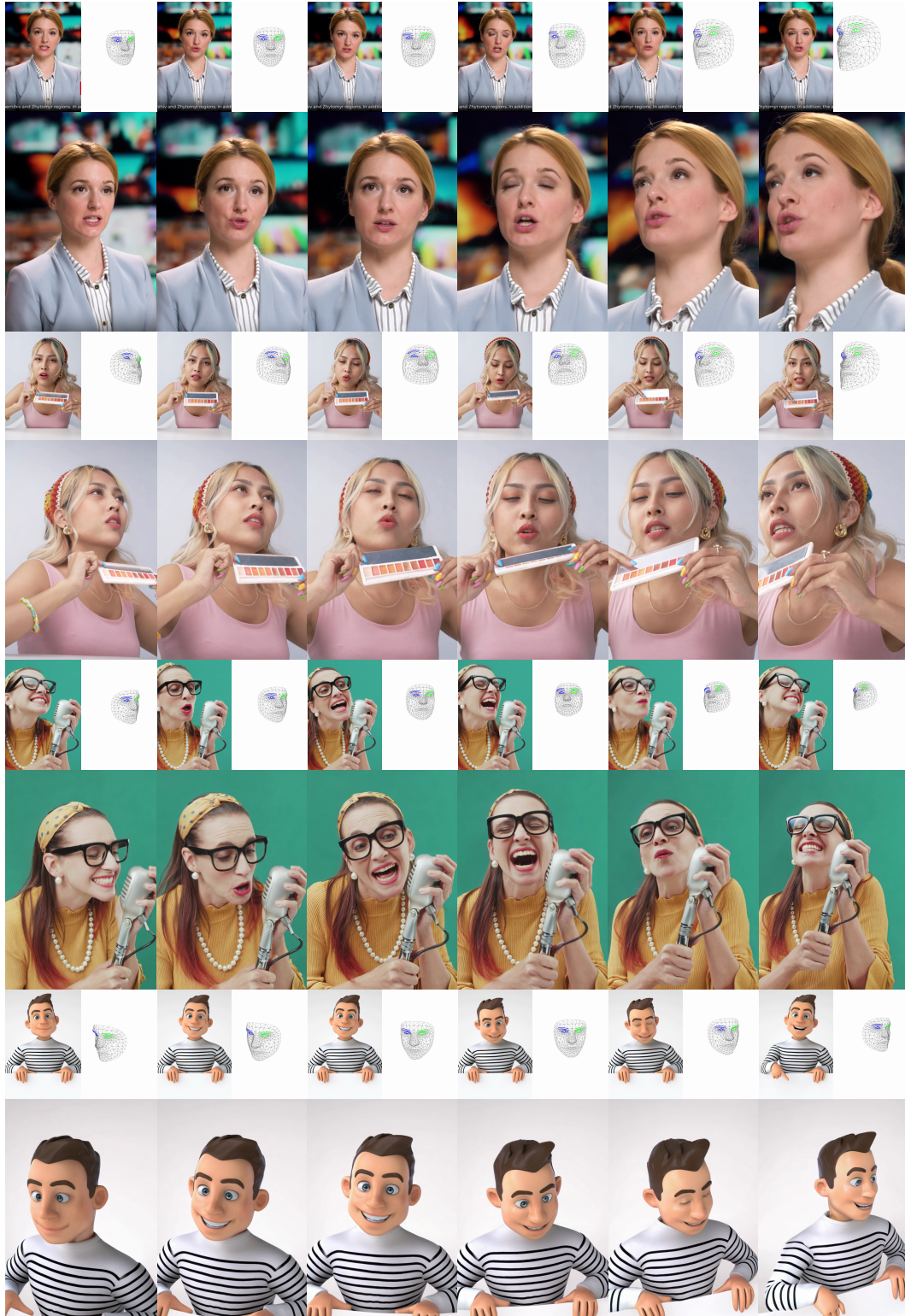


Figure 11. *FaceCam* in real-world scenarios. It recaptures a newscaster with detailed facial texture while keeping the studio background consistent (example 1). It further re-synthesizes an e-commerce streamer (example 2) and a singer (example 3) under novel camera angles, accurately maintaining co-occurring objects such as an eyeshadow palette, earrings, headband, necklace, microphone, and glasses, *etc.* The model even generalizes to cartoon characters (example 4), despite never having seen such content during training.