Generating 3D-Consistent Videos from Unposed Internet Photos

Supplementary Material

6. Video Playback

Please see our project page for video playback: https://genechou.com/kfcw/

7. Training Details

7.1. Dimensions

We resize and center crop all internet photos and video frames to $3 \times 512 \times 512$. The VAE downsamples them to $C \times 64 \times 64$, and each image is then patchified with a patch size of 2, to $1024 \times 4C$. We pass the patches through a linear layer to get $1024 \times D$. Our sinusoidal positional embeddings, frame indices, and CLIP embeddings are all resized through a linear layer and repeated to the same dimensions for addition. For instance, each CLIP embedding of an image has shape 1×1024. We pass it through a linear layer to match the shape $1 \times D$, and repeat it to $1024 \times D$.

Classifier-free guidance. To perform classifier-free guidance, we set all the clean patches (blue patches in Fig. 3) to have diffusion timestep $= 999$ with a probability of 10%, and diffusion timestep $= 0$ otherwise. We remove text-conditioning completely, so CFG is only applied to the condition image patches. During testing, we set the CFG scale to 1.5. We observed unnatural saturation at higher CFG scales.

7.2. Multiview Inpainting Details

Data loading. MegaScenes contains categories of scenes that are labeled by Wikimedia Commons. We sample images from the same category. One sampled batch of images may contain no overlap or noisy images that do not belong to the category, but on the whole this did not affect results. Additional filtering and annotations could make training more stable, if needed.

Our method can take an arbitrary number of condition images, but during experiments, we found two conditions to be a reasonable balance between quality and compute. Training with two conditions led to noticeable improvements from only one condition, but adding more images did not seem to have a significant effect. Thus, we fixed our number of conditions to be 2, for a total of three input images per iteration.

Segmentation. Our video datasets (DL3DV and Re10k) do not contain dynamic objects, so we also removed dynamic objects from internet photos since the model cannot handle motion, such as people walking. We apply DINOv2 [48] for semantic segmentation on the RGB images and mask out all people and vehicles. Since diffusion is performed in latent space, we downsample the masked RGB regions to latent space, and dropout these pixels (i.e. we do not input them to the transformer). This is done for the conditions as well as the target image. When calculating the loss between the denoised and ground truth patches, we also skip the patches that are masked.

We believe that with more video data, we can model dynamic objects realistically, and remove the requirement for segmentation.

Illumination. We also condition each image patch on its CLIP embedding, following the process described in the main paper for view interpolation. Even though illumination information can be acquired from the 20% clean pixels in the target, we add the CLIP embedding for consistency across the two objectives.

7.3. View Interpolation Details

Training augmentation. As mentioned in the main paper, we perform color jittering on the condition images such that illumination information can only be inferred from the CLIP embedding. Specifically we set

```
T.ColorJitter(brightness=0.75,
 ↪  contrast=0.5, saturation=0.6,
 ↪  hue=0.5)
```

Additionally, to simulate segmentation, we randomly drop patches from the condition images (i.e. we do not input them to the transformer).

Data loading. Since we only sample 15 frames between viewpoints, taking neighboring frames from the raw video resulted in excessively small viewpoint changes. We opted for a simpler approach: using the pre-selected frames provided by the datasets. Specifically, we used the frames registered in COLMAP for DL3DV and the frames with pose information for Re10k. We did not use the poses, but the images were easy to download and process. Although the selected frames have different sampling rates, our model generalized to both wide and narrow baselines without any issues.

8. Experiment and Evaluation Details

8.1. User Study

For our study, we randomly sampled 25 scenes: 15 from the Phototourism dataset, and 10 from Re10k. Our method is compared to each baseline via pairwise comparisons. Users are shown two videos generated from the same input frames and are asked to select which method they prefer according to each evaluation criterion (or the user can select "Cannot

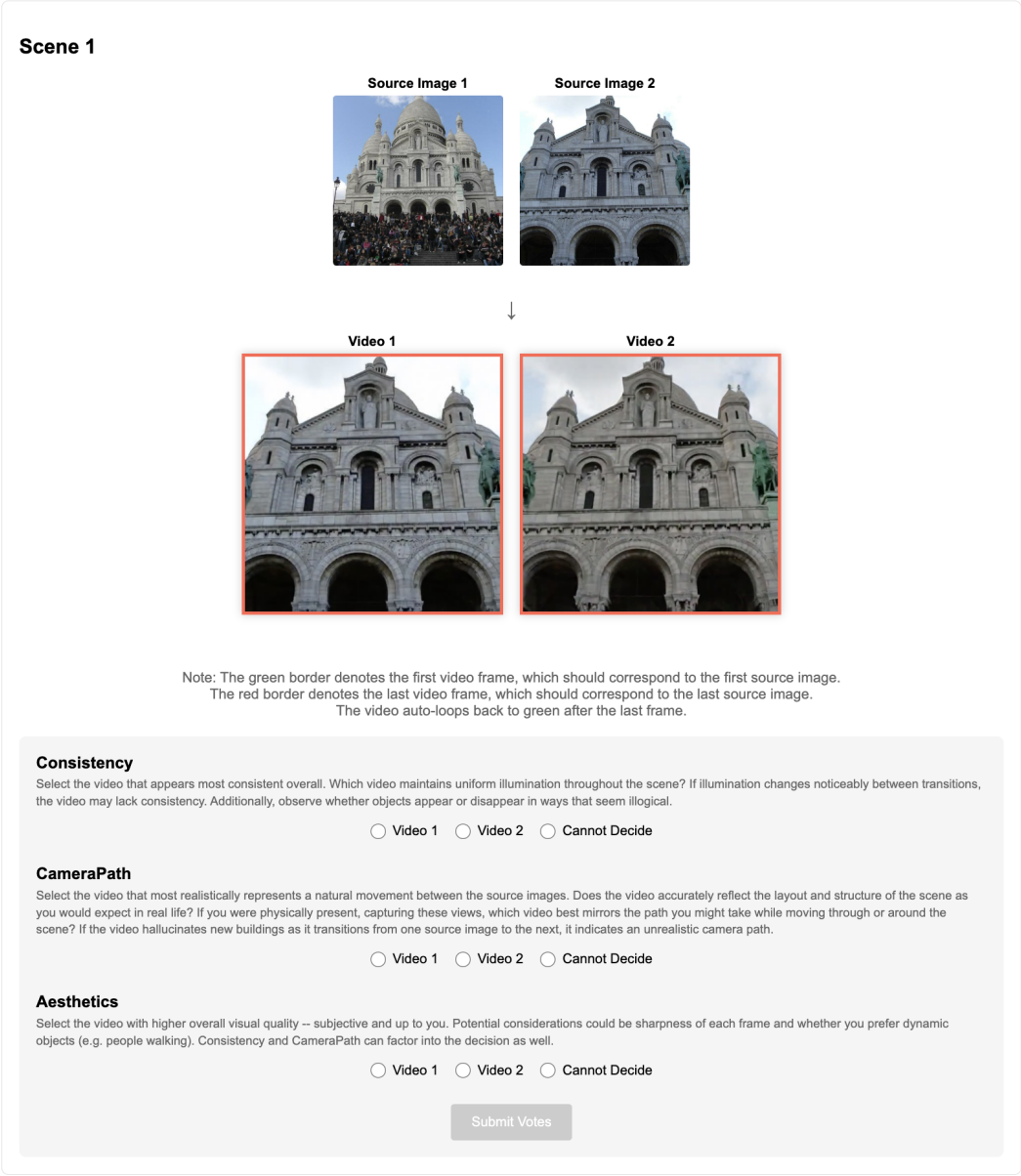

Figure 9. Example scene from our user study interface. We provided detailed descriptions for three criteria: Consistency, CameraPath, and Aesthetics. For each scene, users are asked to express a preference between our results and those of a random baseline.

Decide"). We show the user study interface in Fig. 9, which also includes detailed descriptions of each criterion. When tallying the results, a direct vote counts as 1 point, and a "Cannot Decide" option counts as 0.5 votes each.

8.2. Ablation Setup: Video-only and Long-video

The video-only ablation follows the same procedure as described in Sec. 7.3. For long-video, we randomly sample $j \in [0, 1, 2, 3, 4, 5]$ and skip j frames when sampling from the video sequence. This means the video length can be extended up to 5 times, but each frame in a video sequence is still spaced out uniformly. As shown in the main paper, we find that this still does not generalize as well as the multiview inpainting objective, likely because internet photos contain more diverse viewpoints, such as extreme rotations and zooming levels, that teach the model to find correspondences between images even when there is minimal overlap.

8.3. InstantSplat Training and Testing

We follow the official repository of InstantSplat for both training and rendering images. For training, we simply provide either the original input photos from our Phototourism

and Re10k test sets, or our generated frames from the same input photos. However, InstantSplat does not directly take camera poses as input, but rather uses DUSt3R to initialize poses. Thus, we must use the same coordinate system during testing in order to render the images. We follow the process in the InstantSplat repository. First, we store the point cloud initialized by the training images. Then, during testing, we run DUSt3R again on both the training and testing images. We align only the points from the training images to the stored point cloud, to get a transformation. Then, we can render the testing images by transforming the estimated poses by DUSt3R to the 3DGS model's coordinate frame.

9. Limitations and Future Work

Input keyframes with no overlap. Our method is very robust to extreme viewpoint changes, but it fails when there is no overlap between input keyframes. Sometimes the model performs morphing, similar to Luma's artifacts; other times it attempts to rotate or zoom to link images but produces blurry transitions. We believe solving this task might require different training schemes as it relies heavily on extrapolation rather than finding correspondences.

Fine-grained illumination. To control illumination we use CLIP embeddings, which likely only contain coarse information, since its training captions contain terms like "cloudy," "sunny," rather than physical properties such as sun angles. Thus, even though we show that this method is capable of controlling coarse illumination, such as the general color scale, there are many possibilities for future work for fine-grained control.

Removing occlusions. We remove occlusions through segmentation, but there can be artifacts when the removal is not accurate. More accurate masks, such as a semantic version of Segment Anything, or directly modeling dynamic objects, would be two possible ways to handling this issue.