A Ablation Study: Dual-Clock Denoising

We ablate the dual-clock denoising scheme, presented in 3.3, using the same evaluation protocol described in 4.1. In TTM, the *first tick* t_{weak} sets the initialization noise level for sampling, while the *second tick* t_{strong} sets when we stop overriding the masked part with the noisy warped reference; after this point, all pixels denoise together. For this section, we evaluate this procedure under different settings. In these experiments, we use different timing ticks, denoting the first as t_1 and the second as t_2

First tick (t_1)	Second tick (t_2)	CoTracker distance	Dynamic degree	Imaging quality
t_{weak}	t_{weak}	27.316	0.265	0.623
t_{strong}	t_{strong}	5.528	0.353	0.620
T	0	2.954	0.411	0.578
t_{weak}	0	2.923	0.404	0.576
t_{strong}	0	2.942	0.353	0.579
T	t_{weak}	29.399	0.254	0.622
T	t_{strong}	9.228	0.430	0.615
t_{weak}	t_{strong}	7.967	0.427	0.617

Table 3: **Dual-Clock Ablation.**

The resulting behaviors under different settings, together with their quantitative outcomes, are summarized below and in Table 3:

Single-clock baseline ($t_1 = t_2$). This implies applying SDEdit on the warped video ($t_{\text{weak}} = t_{\text{strong}}$). When $t_1 = t_2 = t_{\text{weak}}$, too little conditioning is induced: the CoTracker distance is high, reflecting poor motion adherence. When $t_1 = t_2 = t_{\text{strong}}$, non-masked regions become over-constrained to unintended motion, suppressing dynamics (e.g., the background freezes); see Fig. 3, where the boat's foam remains static although the boat moves.

RePaint-style ($t_2 = 0$). Here denoising occurs only outside the masked reference (equivalent to RePaint). As expected, for any t_1 the CoTracker distance drops sharply, since the warped masked region is injected throughout denoising. However, this comes at the cost of Imaging quality: the videos appear nearly perfect in motion adherence but unnatural overall, due to the lack of flexibility inside the mask region.

Unconstrained background ($t_1 = T$). No constraint is applied to non-masked regions. For $t_2 = t_{\text{weak}}$, motion is not enforced and the model tends to generate overly static videos. For $t_2 = t_{\text{strong}}$, performance improves, but tracking error remains unsatisfactory; in practice, this setup often produces duplicate copies of the source object, which harms adherence.

Dual clock (ours). $t_1 = t_{\text{weak}},\ t_2 = t_{\text{strong}}$. This setting achieves the best overall trade-off, combining strong motion-conditioning adherence (low CoTracker distance) with higher dynamic degree and robust visual quality.

B Ablation Study: Mask Perturbations

Our object-control evaluation on MC-Bench in Sec. 4 implicitly measures robustness to inaccurate masks: the dataset provides human-annotated brush masks that are coarse, include background regions, and often miss fine object details, rather than pixel-accurate segmentations. To further validate this, we add an experiment that explicitly perturbs the input masks and reports the resulting performance, complementing Sec. 4.1. We follow the same experimental setup, but apply morphological erosion and dilation with varying kernel sizes to the MC-Bench masks, simulating under- and over-segmented masks with different boundary characteristics. As shown in Tab. 4, these perturbations lead to only minor changes in all metrics, indicating that our method is robust to mask inaccuracies.

Morphological Operation	Kernel Size	CoTracker distance	Dynamic degree	Imaging quality
-	-	7.967	0.427	0.617
Dilate	3	8.059	0.435	0.618
Dilate	5	8.527	0.425	0.618
Dilate	7	7.898	0.438	0.617
Erode	3	8.499	0.416	0.618
Erode	5	8.276	0.430	0.617
Erode	7	9.125	0.433	0.617

Table 4: **Mask Perturbations Ablation.**

C Ablation Study: Timestep Sensitivity

Tuning a small set of inference-time parameters is a common requirement in diffusion-based methods, analogous to adjusting the guidance scale in classifier-free guidance (CFG) (Ho & Salimans, 2022) or selecting a timestep schedule in SDEdit. To characterize the sensitivity of TTM to its timestep parameters, we perform an ablation in which we vary t_{weak} and t_{strong} around their default values on MC-Bench and measure the resulting performance. As summarized in Tab. 5, smaller t_{strong} values increase motion adherence and reduce the CoTracker distance, but slightly degrade imaging quality, as the object becomes more rigid and less free to adapt. Conversely, larger t_{weak} values (i.e., more initial noise) generally lead to a higher dynamic degree. Overall, the trends are smooth, and our default settings lie in a stable operating regime that provides a favorable trade-off between motion control and visual fidelity.

t_{weak}	t_{strong}	CoTracker distance	Dynamic degree	Imaging quality
38	27	11.571	0.419	0.620
36	27	10.596	0.454	0.619
34	27	9.546	0.416	0.619
38	25	8.576	0.433	0.617
36	25	7.967	0.427	0.617
34	25	8.031	0.419	0.618
38	23	6.500	0.438	0.612
36	23	6.757	0.419	0.612
34	23	6.130	0.414	0.615

Table 5: **Timestep Sensitivity Ablation.** The original experiment used $t_{weak} = 36$ and $t_{strong} = 25$.

D Implementation Details

D.1 Object Motion Control

This subsection complements Sec. 4.1 with concise protocol and implementation details:

- **Single-Trajectory.** To avoid ambiguity stemming from masks linked to multiple objects/trajectories in the original MC-Bench dataset, we restrict evaluation to single-trajectory cases (over 91% of the dataset).
- **Input handling.** Inputs are resized to each model's native size and padded to match aspect ratio; after generation, padding is removed and outputs are resized back. Exceptions: MotionPro uses its original benchmark pipeline; DragAnything is run with its default input handling (we observed best results without external resizing).
- **Trajectory scaling**: the 2D trajectory points are affinely transformed with the *same* resize-and-pad mapping applied to the frames. After generation, we remove padding and invert the scaling when mapping tracks back for evaluation, ensuring geometric consistency.
- **Clip length.** Standardized to 16 frames for SVD-based methods (as Zhang et al. (2025b)) and 49 frames for CogVideoX-based methods. Concretely, SVD emits 14 or 25 frames—thus TTM_{SVD} generates 25 and keeps the first 16; DragAnything emits 20 and we keep the first 16; SG-I2V produces 14 and we evaluate the native 14-frame output, which may be slightly favorable to its metrics. If the output has more frames than the provided trajectory, we trim the trajectory to the target length.
- **Pre/post-processing and prompts.** SG-I2V is conditioned on bounding boxes rather than masks, unlike the other methods. Therefore, following Burgert et al. (2025), we supply the tight bounding box of the provided mask. For prompts, SVD-based methods are text-free, while CogVideoX-based methods use the MC-Bench prompts.
- **Mask resizing**: Since both SVD and CogVideoX operate in a downsampled latent space, we project the binary masks to the latent resolution with nearest-neighbor interpolation. For SVD this is spatial-only; for CogVideoX also subsample in time to match temporal compression (nearest-neighbor in time).
- **Hyperparameters.** All methods use $T{=}50$ denoising steps. For TTM_{SVD} set $(t_{\text{weak}}, t_{\text{strong}}) = (36, 25)$ and fix MotionPro's motion bucket to 17 (as in their release). For TTM_{Cog} use $(46, 41)$. Other run-time settings follow each method's defaults.

- **VBench.** For CogVideoX-based 49-frame models, we use the long benchmark variant[2].
- **Dynamic Degree.** VBench flags a clip as *dynamic* when the mean of the top 5% RAFT flow magnitudes in a frame exceeds a resolution-scaled threshold $\alpha \cdot \frac{\min(H,W)}{256}$ in at least 25% of sampled frames. The default $\alpha = 6.0$, tuned for VBench's source videos, is too strict for our MC-Bench setting—predominantly static camera with small, localized motions—so nearly all clips are marked static. We therefore set $\alpha = 3.5$ (keeping the 25% rule unchanged), which yields a more meaningful separation of dynamic vs. static.
- **CoTracker Distance (CTD).** Following MC-Bench's MD-Vid protocol, we leverage recent advances in point tracking (Sand & Teller, 2008; Tumanyan et al., 2024) and employ CoTracker (Karaev et al., 2024) to assess the alignment between the input 2D trajectory and the motion in the generated video, measured by the mean per-frame Euclidean distance in pixel space (lower is better).
- **Background–Object CoTracker Distance (BG–Obj CTD).** This metric measures whether the background unintentionally moves together with the controlled object. We run CoTracker on the generated video for both (i) the tracked object trajectory and (ii) a uniform 16×16 grid of points sampled in the first frame. Let $o_t \in \mathbb{R}^2$ denote the object's tracked position at frame t, and $p_{j,t} \in \mathbb{R}^2$ the tracked position of grid point j at frame t. We convert all tracks to displacements from frame 1: $\Delta o_t = o_t - o_1$, $\Delta p_{j,t} = p_{j,t} - p_{j,1}$. For each frame $t \geq 2$ and grid point j, we compute $d_{j,t} = \|\Delta p_{j,t} - \Delta o_t\|_2$ (pixels). The BG–Obj CTD is then the average over frames and grid points:

$$\text{BG–Obj CTD} = \frac{1}{(T-1)J} \sum_{t=2}^{T} \sum_{j=1}^{J} d_{j,t}.$$

  Higher values indicate stronger object–background disentanglement (less co-motion).

D.2 Camera Control on DL3DV

This subsection provides additional details for Sec. 4.2. For our camera control experiments, we use a subset of the DL3DV-10K dataset. The reference warped videos are created at a resolution of 960p using PyTorch3D.

We utilize the official DL3DV camera transition data and align the coordinate systems with a sign flip of the z-axis and a flip of the camera pitch due to convention differences between PyTorch3D and NerfStudio, which was used to create the DL3DV dataset. The point cloud is generated in the original camera frame, with the camera extrinsics derived from parameter estimations and the estimated transition of the first frame. To resolve the inherent depth ambiguity, we perform a binary search to find the transition scale that maximizes the MSE alignment between the warped and original videos. This aligns the transitions to a metric scale consistent with the output of DepthPro.

To select a robust subset for evaluation, we first filter out videos from the 10K subset with an estimated scale of less than 0.3, as these were found to exhibit minimal real camera movement. We then select the 150 scenes with the lowest MSE loss between the warped and ground truth videos. The masks used for the TTM process are generated by first marking pixels with no point cloud contribution as regions denoised freely from t_{weak}, while all other pixels are regions denoised freely from t_{strong}. The strong guidance masks are generated by first marking pixels with no point cloud contribution as 0 (no guidance) and all other pixels as 1 (guidance). To ensure only regions with dense point cloud data are used for guidance, we apply a morphological "open" operation to the mask using a kernel size of 5. This operation serves to remove isolated noise and expand the non-guidance areas, resulting in a cleaner, more reliable mask. For text guidance, we automatically generate a text prompt for each scene using GPT-4o (OpenAI, 2024), following CogVideoX's protocol[3].

D.3 Appearance Control

For the chameleon example demonstrating joint motion and appearance control, we use the prompt: "A realistic video of a four-legged chameleon walking slowly and naturally from left to right along

[2] https://github.com/Vchitect/VBench/tree/master/vbench2_beta_long

[3] https://github.com/zai-org/CogVideo/blob/main/inference/convert_demo.py

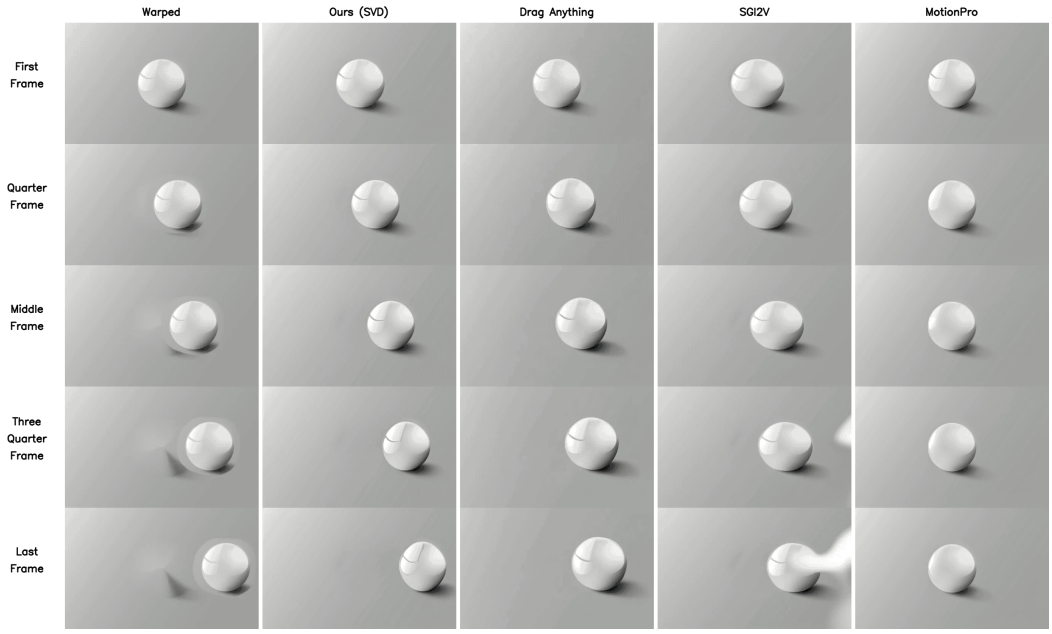
a thick, textured vine in a lush jungle. Its limbs move in a coordinated, controlled reptilian gait as it adjusts its body to the curve of the vine. The chameleon gradually changes color from green to purple."

E Extra Qualitative Comparisons

For the video versions of the comparisons in this paper, as well as additional results, please visit our project page.

E.1 Qualitative Comparisons from MC-Bench

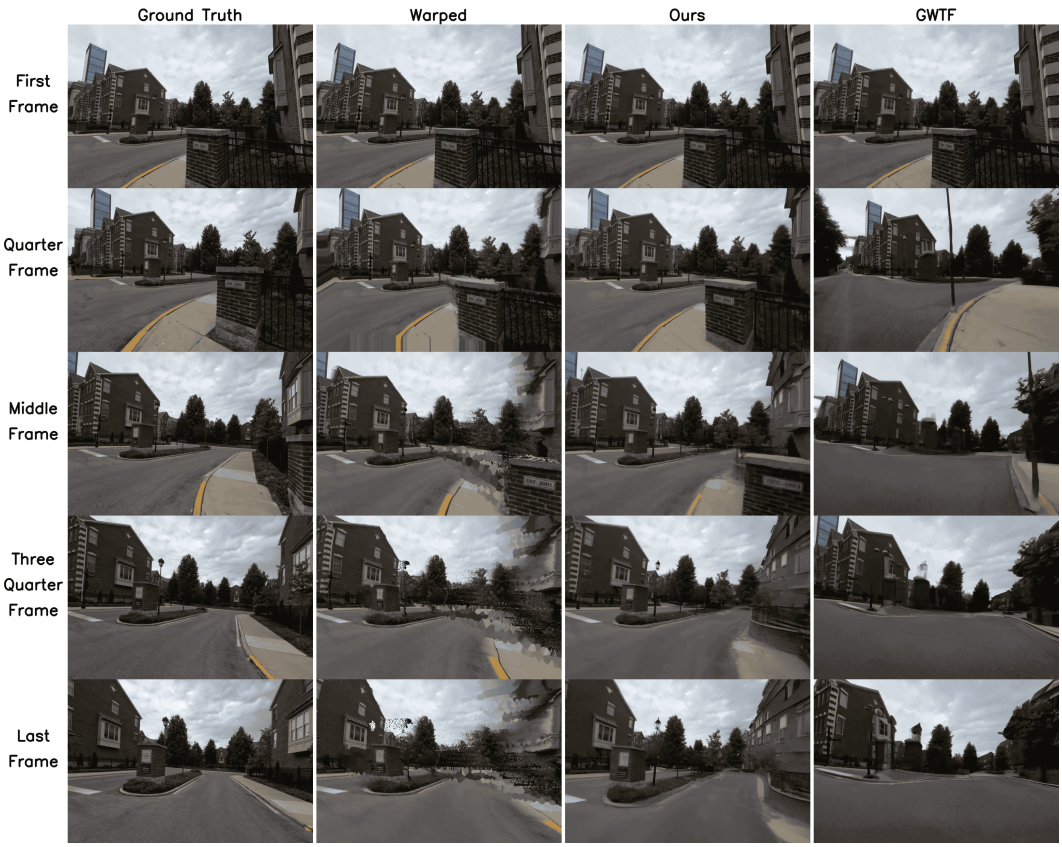
Following the experiment described in Sec. 4.1, we present additional results beyond those shown in Fig. 4, further illustrating our method's performance against leading approaches on the MC-Bench dataset using the SVD backbone.


Warped
Ours (SVD)
Drag Anything
SGI2V
MotionPro
First Frame
Quarter Frame
Middle Frame
Three Quarter Frame
Last Frame


Warped
Ours (SVD)
Drag Anything
SGI2V
MotionPro
First Frame
Quarter Frame
Middle Frame
Three Quarter Frame
Last Frame

E.2 QUALITATIVE COMPARISONS FROM DL3DV

Following the experiment described in Sec. 4.2, we present qualitative results for camera-motion control on the DL3DV benchmark, comparing our method with GWTF given an input image, its monocular depth estimate, and a depth-warped video. These examples demonstrate superior performance in maintaining the intended camera motion and overall visual fidelity.

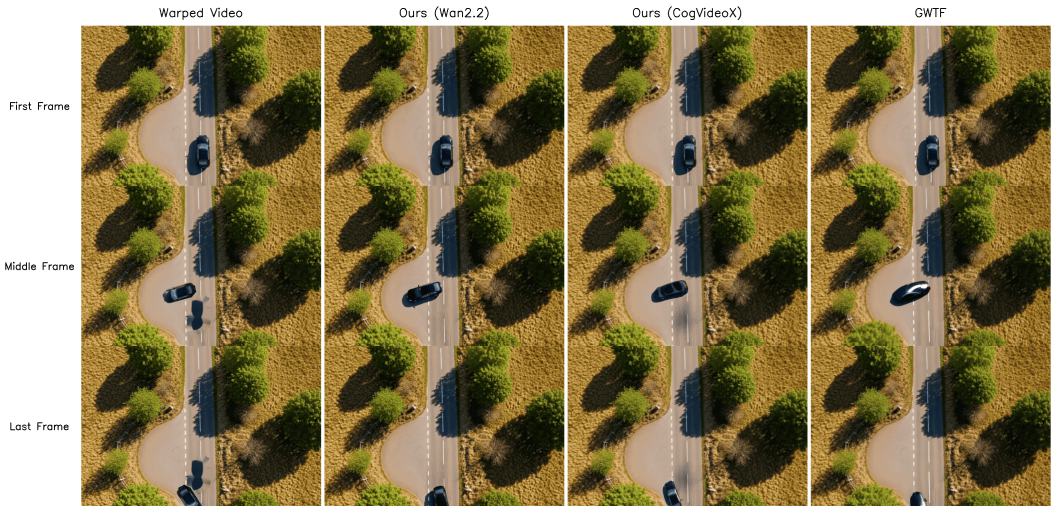

Ground Truth
Warped
Ours
GWTF
First Frame
Quarter Frame
Middle Frame
Three Quarter Frame
Last Frame

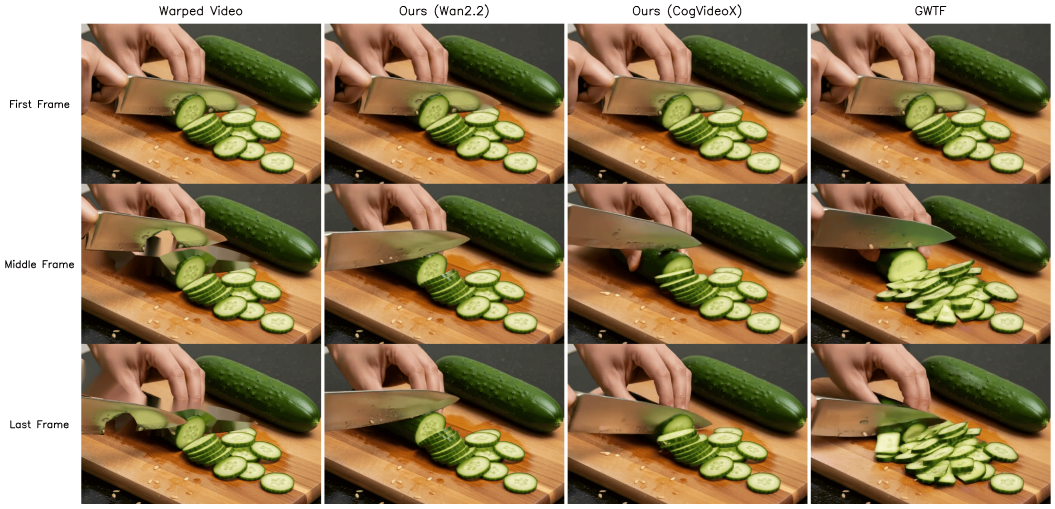

Ground Truth
Warped
Ours
GWTF
First Frame
Quarter Frame
Middle Frame
Three Quarter Frame
Last Frame

F CHALLENGING USER-CREATED EXAMPLES

Generation. To produce the examples shown in Fig. 5, Fig. 6, and on the demo page, we collected 53 test cases, hand-crafted by users, spanning both object-motion and camera-motion control. For each case, the initial reference frame was generated with Gemini (Comanici et al., 2025), and object-control inputs were specified via a GUI adapted from the interface introduced in (Burgert et al., 2025).

Additional Results As explained in Sec. 4.4, we leverage the plug-and-play nature of our method to run on the recently released WAN2.2. Below we present additional "cut-and-drag" examples that complement Fig. 5. These real-world cases illustrate scenarios in which the current state-of-the-art baseline, GWTF, often struggles to produce coherent results.




Warped Video
Ours (Wan2.2)
Ours (CogVideoX)
GWTF
First Frame
Middle Frame
Last Frame


Warped Video
Ours (Wan2.2)
Ours (CogVideoX)
GWTF
First Frame
Middle Frame
Last Frame

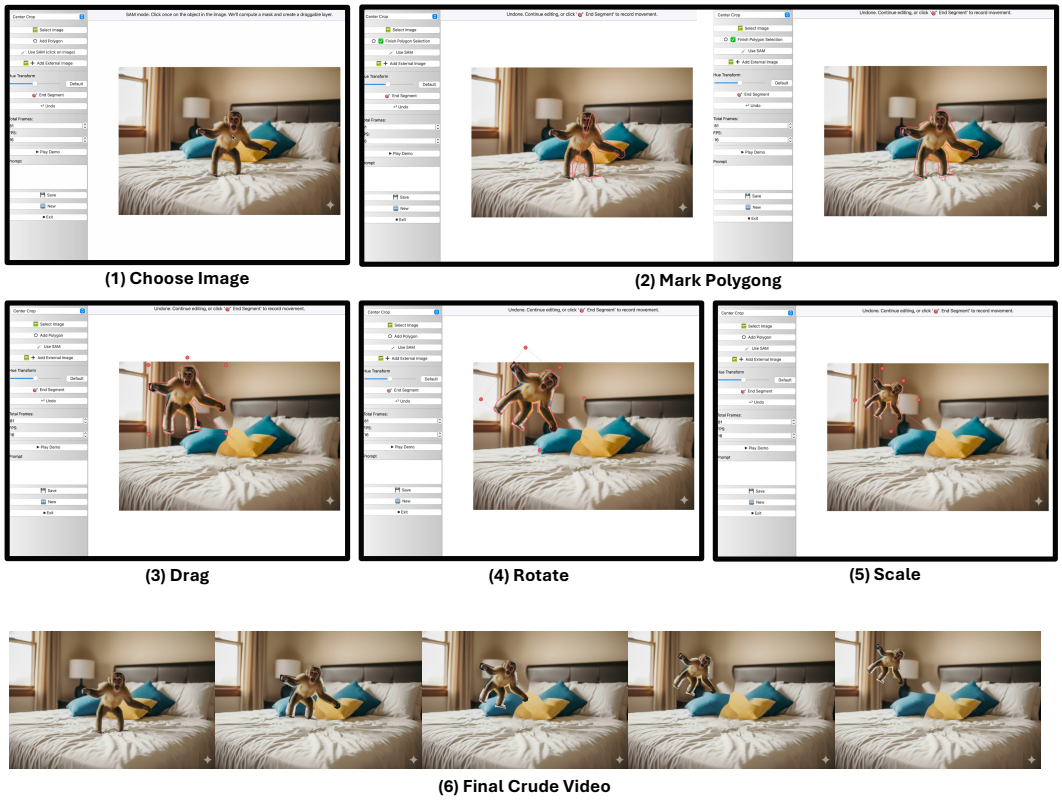


Figure 8: Example interaction with our cut-and-drag GUI. (1) The user selects an input image. (2) A region (e.g., the monkey) is defined via a polygon. (3) The object is dragged to define a motion segment. The final key pose is then refined by applying controls for (4) rotation and (5) uniform scaling. (6) The GUI interpolates these key poses, automatically generating the warped video and corresponding mask sequence, where the object gradually moves, rotates, and scales to its final position; these serve as the final motion signal for the TTM framework.

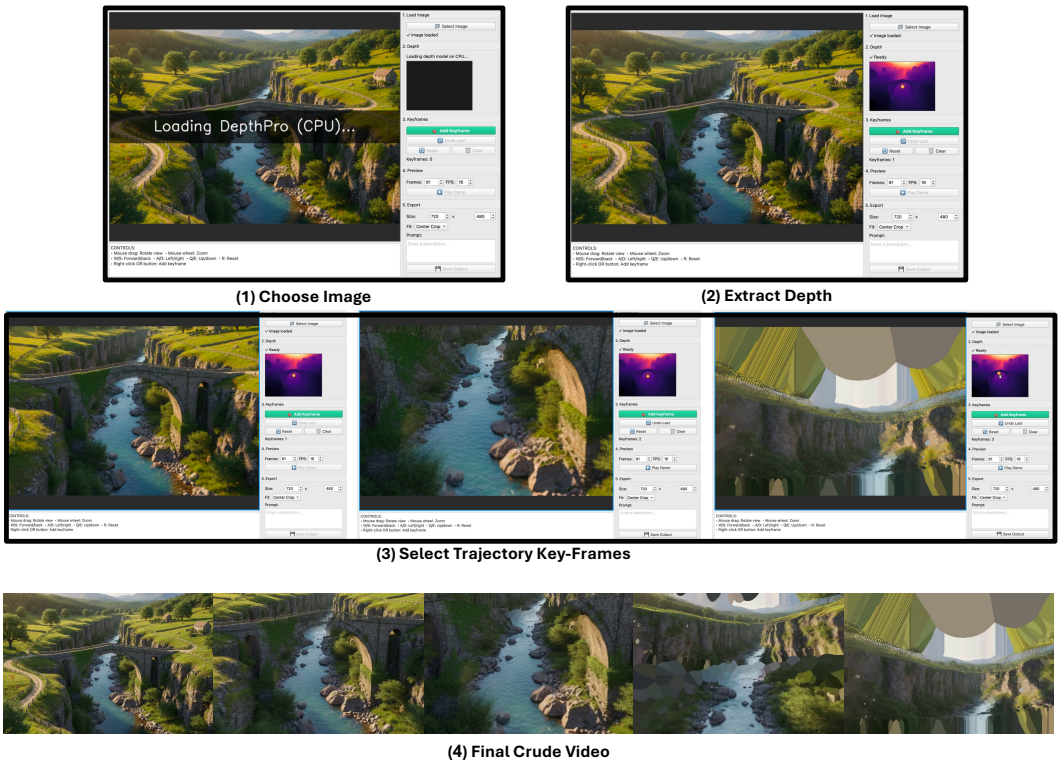


Figure 9: Example interaction with our camera-control GUI. (1) The user selects an input image. (2) The GUI automatically estimates depth and constructs a 3D reprojection for view synthesis. (3) The user navigates the 3D scene and selects camera-pose keyframes to define a trajectory, which is visualized and interpolated by the GUI. (4) The resulting depth-based reprojection yields a warped "crude" video, which is used as the motion signal for TTM.