Supplementary Material – PhysGen: Rigid-Body Physics-Grounded Image-to-Video Generation

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta*, and Shenlong Wang*

University of Illinois Urbana-Champaign
https://stevenlsw.github.io/physgen/

Abstract. The supplementary material provides implementations, additional analysis and experiments, as well as discussion of limitations in detail. In summary, we include

- Appendix A. More implementation details of each module in our pipeline.
- Appendix B. More experiment details in the main paper.
- Appendix C. More experiments on key components in our framework.
- Appendix D. Additional qualitative comparisons and controllable generation results.
- Appendix E. Generation limitation analysis of our current approach.

A Implementation Details

A.1 Perception

Segmentation. We use GPT-4V to recognize all objects in the given image and determine if they are movable. Movable objects are treated as foreground objects, while non-movable objects are treated as background objects. The query prompt is shown in Fig. 1. The outputs from GPT-4V are sent to Grounded-SAM [6] for instance segmentation. We also enable non-maximum suppression (NMS) to prevent overlapping segmentation. For foreground objects, we check if the segmentation is fully connected within its mask. If not, we rerun Grounded-SAM to further separate any poor segmentation from the first round.

Boundary extraction. To extract boundaries from non-movable background objects, we utilize depth and normal information estimated from GeoWizard [2]. We first order the background objects according to relative depth estimation and select only those whose depth range falls within that of the foreground objects. For candidate objects, we extract their segmentation boundaries within the foreground object's depth range and use corresponding normal to determine if they are planes. If so, we fit horizontal or vertical edges to the boundaries and use it as the physical boundary of the scene.

* Equal advising

User: Describe all unique object categories in the given image, ensuring all pixels are included and assigned to one of the categories, do not miss any movable or static object appeared in the image, each category name is a single word and in singular noun format, do not include '-' in the name. Different categories should not be repeated or overlapped with each other in the image. For each category, judge if the instances in the image is movable, the answer is True or False. If there are multiple instances of the same category in the image, the judgement is True only if the object category satisfies the following requirements: 1. The object category is things (objects with a well-defined shape, e.g. car, person) and not stuff (amorphous background regions, e.g. grass, sky, largest segmentation component). 2. All instances in the image of this category are movable with complete shape and fully-visible.
Format Requirement: You must provide your answer in the following JSON format, as it will be parsed by a code script later. Your answer must look like: "category-1": False, "category-2": True
Do not include any other text in your answer. Do not include unnecessary words besides the category name and True/False values.

Fig. 1: Prompt used for GPT-4V image recognition and movability judgment.

Physical properties reasoning. The query prompt for reasoning the physical properties of a foreground object is shown in Fig. 2.

Geometry primitives. Given a object segmentation mask, we automatically choose the proper primitive that best fits the object. We first use a circle to fit the corresponding segmentation mask and compute the Intersection over Union (IoU) between the fitted mask and segmentation mask. If IoU is smaller than 0.85, we switch to the generic polygons by extracting the contour of the segmentation.

Background Inpainting. Given the foreground masks of the input image, we use off-the-shelf image inpainting model [8] to recover the background scene. Considering foreground objects might have shadow beneath, we dilate the foreground segmentation mask by a kernel size of 40 pixels. To this end, we aim to get a clean background image without shadows. However, if the input image is heavy-shadowed, the inpainting model could not remove it completely. We discuss it use a detailed example in Appendix E.

A.2 Image Space Dynamics Simulation

In image space dynamics simulation, we set Δ_t as 1 second, gravity as $g = 980cm/s^2$. Considering most foreground objects are captured at a similar distance, we set 1pixel as 1cm to map from image space to world space without reasoning metric scale. We visualize the physical simulation procedure of billiard balls and blocks with two different primitives (circle and polygon) in Fig. 3.

User: You will be given an image and a binary mask specifying an object on the image, analyze and provide your final answer of the object physical property. The query object will be enclosed in white mask. The physical property includes the mass, the friction and elasticity. The mass is in grams. The friction uses the Coulomb friction model, a value of 0.0 is frictionless. The elasticity value of 0.0 gives no bounce, while a value of 1.0 will give a perfect bounce.
Format Requirement: You must provide your answer in the following JSON format, as it will be parsed by a code script later. Your answer must look like: "mass": number, "friction": number, "elasticity": number The answer should be one exact number for each property, do not include any other text in your answer, as it will be parsed by a code script later.

Fig. 2: Prompt used for GPT-4V image recognition and movability judgment.

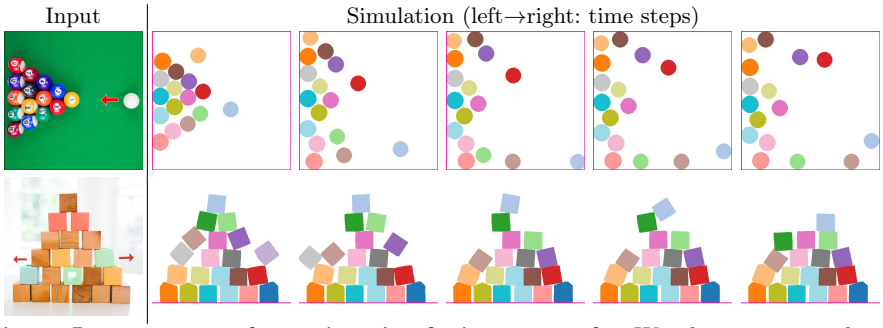

Fig. 3: Image space dynamics simulation example. We show two simulation examples with different primitives. The primitives are fitted from the segmentation mask of the input image and we perform dynamic simulation on those primitives. The edges show the physical boundary.

A.3 Generative Refinement Algorithm

We present the detailed algorithm of the proposed generative refinement with latent diffusion models in Sec. 3.4. In video diffusion model, a input video $\mathbf{V} \in \mathbb{R}^{T\times H\times W\times 3}$ is encoded to a latent vector via a encoder $\mathcal{E}$ by $\mathbf{z}_0 = \mathcal{E}(\mathbf{V}) \in \mathbb{R}^{T\times h\times w\times 3}$, where c is the dimension of the latent space. The forward diffusion process [4] is to iteratively add Gaussian noise to the signal, given by Eq. (1).

$$\begin{aligned} q\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}\right) &= \mathcal{N}\left(\mathbf{z}_t; \sqrt{1-\beta_{t-1}}\mathbf{z}_{t-1}, \beta_t\mathbf{I}\right), \quad t = 1,\ldots,T \\ q\left(\mathbf{z}_t \mid \mathbf{z}_0\right) &= \mathcal{N}\left(\mathbf{z}_t; \sqrt{\bar{\alpha}_t}\mathbf{z}_0, (1-\bar{\alpha}_t)\mathbf{I}\right), \quad t = 1,\ldots,T \end{aligned} \tag{1}$$

where $\alpha_t = 1-\beta_t$, $\bar{\alpha}_t = \prod_{i=1}^{t}\alpha_i$, $q\left(\mathbf{z}_t \mid \mathbf{z}_{t-1}\right)$ is the one-step forward diffusion process, and $q\left(\mathbf{z}_t \mid \mathbf{z}_0\right)$ is t-step forward diffusion process. T is a large integer to make the forward process completely destroys the initial signal $\mathbf{z}_0$ resulting in $\mathbf{z}_T \sim \mathcal{N}(0,\mathbf{I})$. The diffusion model learns to recover $\mathbf{z}_0$ from standard Gaussian

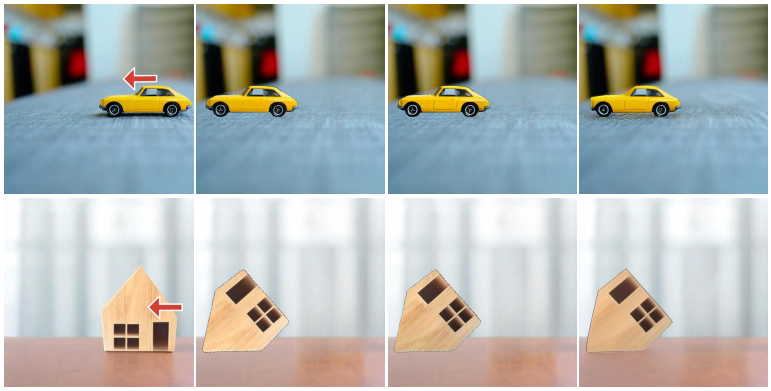

Fig. 4: Rendered Video comparison. The left shows the input frame, and the rest 3 are future frame generations. The composited frame hasn't been aware of the light change. The relighted output synthesized no shadows beneath. The rendered output from diffusion model is most photorealistic.

Table 1: Runtime analysis. We summarize the runtime of each module for each run.

Perception			Simulation	Render	Refinement
Segmentation	GPT4-V	Inpainting			
50s	10s	20s	5s	60s	35s

noise $\mathbf{z}_T$ by backward diffusion process in Eq. (2).

$$p_\theta\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t\right)=\mathcal{N}\left(\mathbf{z}_{t-1} ; \mu_\theta\left(\mathbf{z}_t, t\right), \mathbf{\Sigma}_\theta\left(\mathbf{z}_t, t\right)\right), \quad t=T, \ldots, 1 \tag{2}$$

where θ is the learned parameters of the model. The output is passed to a decoder $\mathcal{D}$ to generate the output video $\mathbf{V}=\mathcal{D}(\mathbf{z}_0)$. To this end, given the relit video $\tilde{\mathbf{V}}$, we encode to get $\tilde{\mathbf{z}_0}$, our goal is to obtain the denoised $\mathbf{z}_0$ from the pretrained latent-diffusion-based video p_θ. The algorithm is shown in Algorithm 1.

In Fig. 4, we compare the composed video $\hat{\mathbf{V}}$, the relit video $\tilde{\mathbf{V}}$, and our final output $\mathbf{V}$ by 2 additional examples.

A.4 Runtime Analysis

We summarize the runtime of each module in Tab. 1 for a single run. The perception module takes 1 minute, the simulation (120 steps) takes 5 seconds, the render module takes 1 minute, the generative refinement takes 35 seconds. In total it takes around 3 minute for a single generation, which is much faster than other controllable generation, *e.g.* Motion Guidance [3] takes 70 minutes for a single run.

Algorithm 1 Video Generative Refinement

Input: Init $\tilde{\mathbf{z}_0}$, foreground mask m, video diffusion model p_θ, noise strength s, fusion timestamp δ, total denoising timesteps T
Output: refined latent $\mathbf{z}_0$

$T \leftarrow \lfloor T * s \rceil$ ▷ Denoised time steps
$\mathbf{z}_T, \tilde{\mathbf{z}_T} \sim q\left(\mathbf{z}_T \mid \tilde{\mathbf{z}_0}\right)$ ▷ Add noise to the guidance latent code
for $t = T, T-1, ..., 1$ **do**
  if $t \leq (T - \delta)$ **then**
    $\tilde{\mathbf{z}_{t-1}} \sim q\left(\mathbf{z}_{t-1} \mid \tilde{\mathbf{z}_0}\right)$ ▷ Add noise to guidance code at $t-1$
    $\mathbf{z}_{t-1} \sim p_\theta\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t\right)$ ▷ Denoised output from network at $t-1$
    $w = (T - t)/T$ ▷ fusion weight
    $\mathbf{z}_{t-1} \leftarrow (1-m)\mathbf{z}_{t-1} + m\left[w\mathbf{z}_{t-1} + (1-w)\tilde{\mathbf{z}_{t-1}}\right]$ ▷ update latent code
  else
    $\mathbf{z}_{t-1} \sim p_\theta\left(\mathbf{z}_{t-1} \mid \mathbf{z}_t\right)$ ▷ Denoised output from network
  end if
end for
return $\mathbf{z}_0$

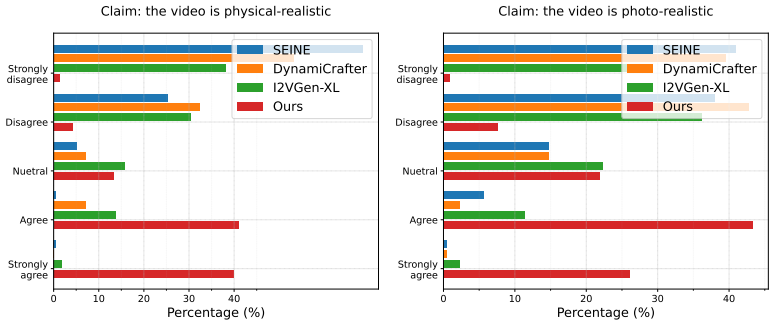

Fig. 5: Human evaluation score distribution. The distribution of scores shows our method largely outperforms other I2V generative models in both physical-realism and photo-realism. Our average rate is close to agree for both two claims.

B Experiment details

B.1 Human evaluation

Fig. 5 shows the score distribution of the conducted human evaluation. Our method has much higher percentage in agree and strongly-agree to both claims, outperform compared I2V methods by a large margin in physical-realism and photo-realism. Our average rate falls within Agree level.

B.2 Quantitative evaluation

To quantitative evaluate the generation performance and compare with other approaches, we record 50 videos for a given scene as GT by varying initial forces applied on the object. The random selected sequences are shown in Fig. 6.

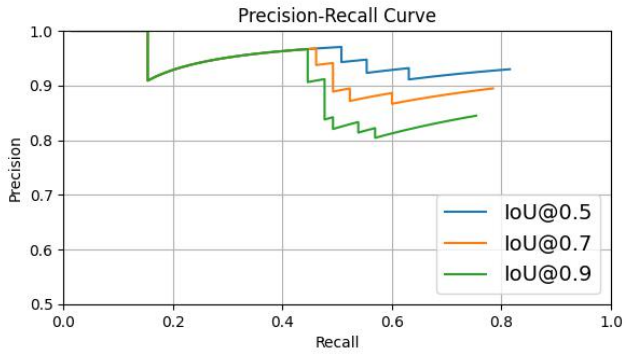

Fig. 7: Precision-recall curve of open-world movable objects segmentation. Our proposed pipeline achieves **0.93** precision, **0.82** recall at 0.5 IoU.

C More experiments

C.1 Open-world movable object segmentation

To measure the performance of the open-world movable object segmentation, we collect 10 images of very complicated open-world scenes where 118 movable instances are annotated following COCO [5] format. The perception system works well, achieves **0.93** precision **0.82** recall under 0.5 IoU. The precision-recall curve is shown in Fig. 7. Qualitative results are shown in Fig. 9.

C.2 GPT-4V physical property estimation evaluation

We evaluate the physical property estimator GPT-4V used in the paper. For *1) mass*: follow [9], we select 20 different portable objects from ABO dataset and find that GPT4-V has an average absolute error of **0.39 kg** and achieves **75%** accuracy within 30% of the GT. *2) friction and elasticity*: Since GT is unavailable, we use reference videos of toy cars sliding on various surfaces and balls of different materials bouncing to rank materials by friction and elasticity. Comparing the models' rankings to the ones in the videos, GPT-4V gives reasonable estimations, getting **12 out of 13** for friction and **6 out of 7** for elasticity across different comparisons. Two testing scenarios of friction and elasticity are shown in Fig. 8.

D More qualitative results

D.1 Qualitative comparison

We visualize more qualitative comparisons in Fig. 10 and Fig. 11. As can be seen, image-video generation methods could not generate physically plausible outputs. We notice DynamiCrafter [7] model tends to generate static scene with lighting

or viewpoint changes, without output reasonable physical motion. SEINE [1] and I2VGen-XL [10] outputs reasonably apparent motions, but the motions are not physical realistic, and the foreground objects could not guarantee consistency across different time steps. Please check the supplementary video for details.

D.2 More controllable generation

We show more controllable generation results in Fig. 12.

E Limitation Analysis

We summarize 4 different generation artifacts in Fig. 13.

Incorrect Inpainting. The first row shows the domino example. The generation results contains incorrect shadow in the middle of the frame. The reason is that the shadow of the boxes could not be fully removed in the inpainted image as shown on the right.

Gap between segmentations and primitives. The second row shows the blocks example where there is a gap between different blocks in the generated videos, whereas no such phenomenon appeared in simulation. The reason is the primitives used in the simulation is slight different from the real segments, thus could cause some gap in the simulated results and rendered outputs.

Inaccurate Segmentation. The third row shows the billiard example where the balls' shape is not perfect circle in the generated frames given the input segmentation mask is not accurate on the boundaries. Thus the synthesized ball has artifacts near its boundary. The generated video quality is affected by the segmentation mask of the input image.

Hallucinations introduced by diffusion refinement. The fourth rows shows the blocks example where the diffusion refinement brings hallucination of the input object. In our proposed generative refinement algorithm, the generated latent injects into both the foreground and background. Thus in some scenarios there could be hallucinations of the foreground object and slightly modify its appearance, *e.g.* the block on the top.

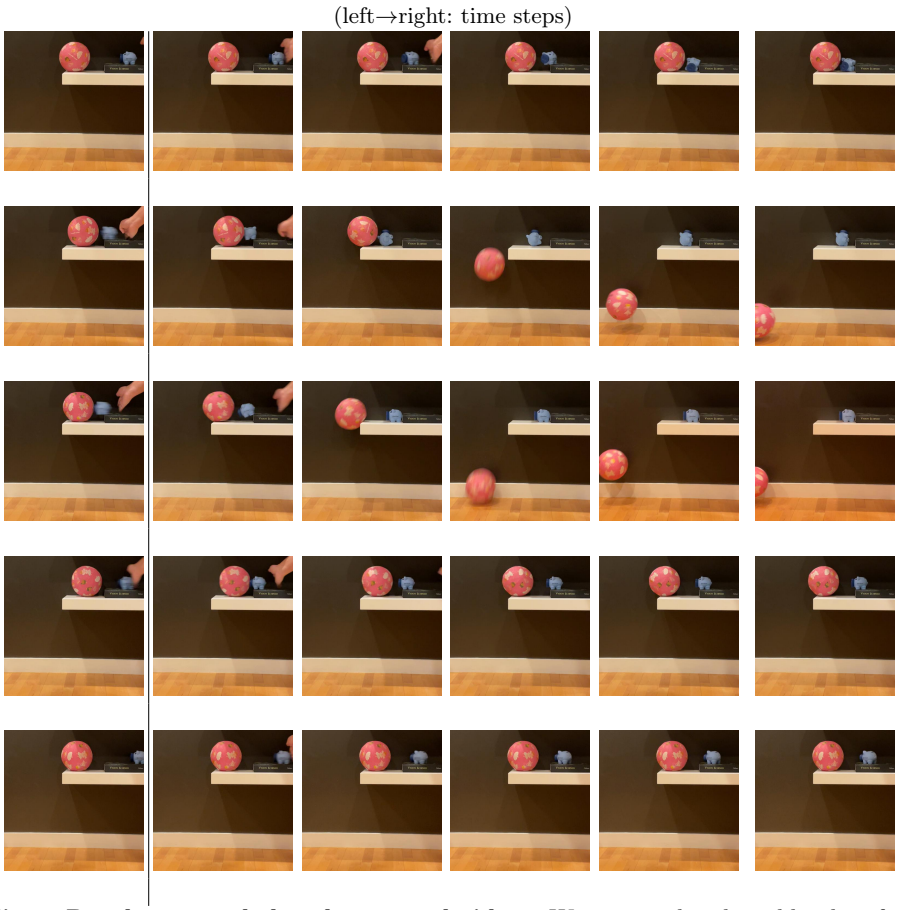

Fig. 6: Random sampled real-captured videos. We captured real-world videos for the same given scene 50 times to evaluate the generation fidelity. We vary initial force on the hand that applied to the blue piggy bank in each run, and record the videos. We **random select** 5 recored videos for visualization.

Fig. 8: GPT-4V physical property estimation evaluation testing scenes. Given directly measure friction and elasticity is hard, we use reference videos of toy cars sliding on various surfaces (left) and balls of different materials bouncing (right) to rank materials by friction and elasticity.

Fig. 9: Qualitative visualization of open-world movable objects segmentation results. From left to right, we show the selected input image, inferred segmentation from our perception module and GT movable object segmentations.

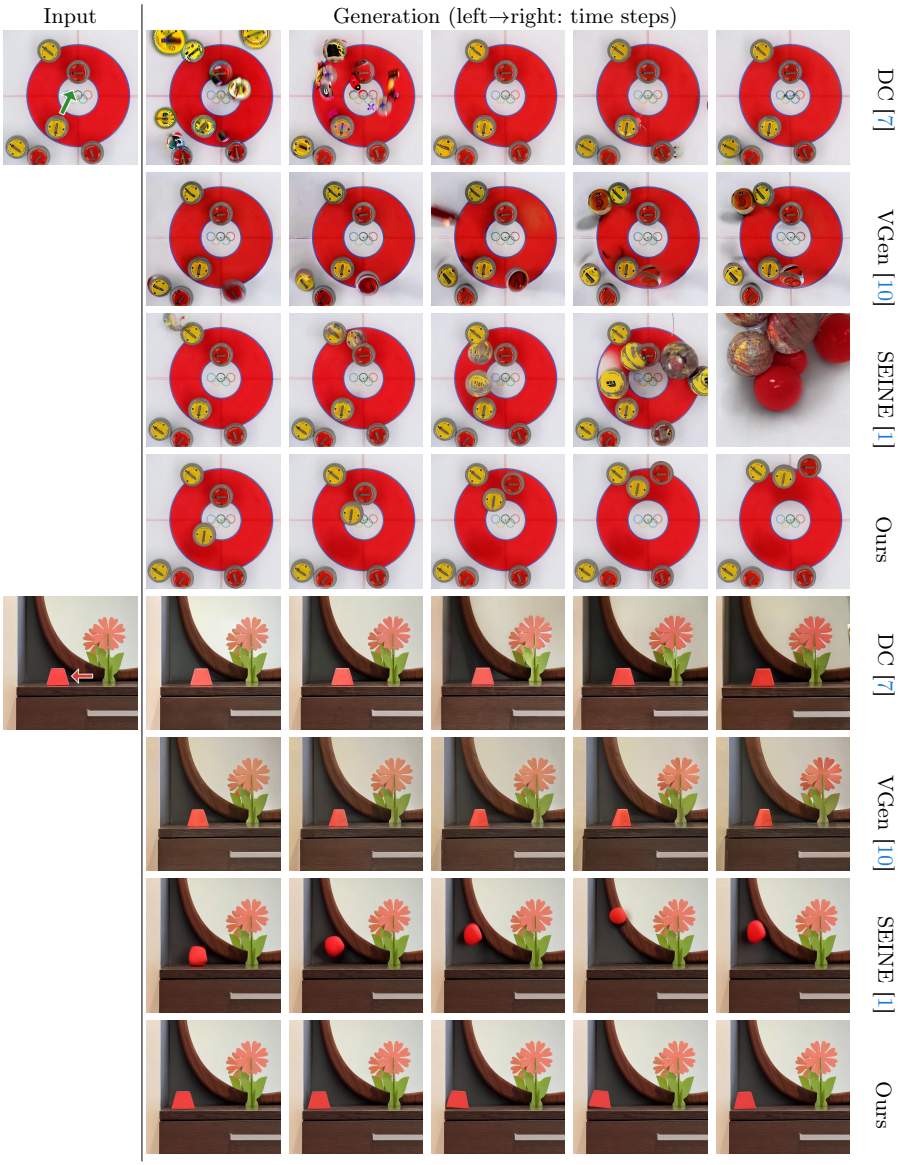

Fig. 10: Additional qualitative comparison against I2V generative models: DynamiCrafter(**DC**) [7], I2VGen-XL(**VGen**) [10], **SEINE** [1].

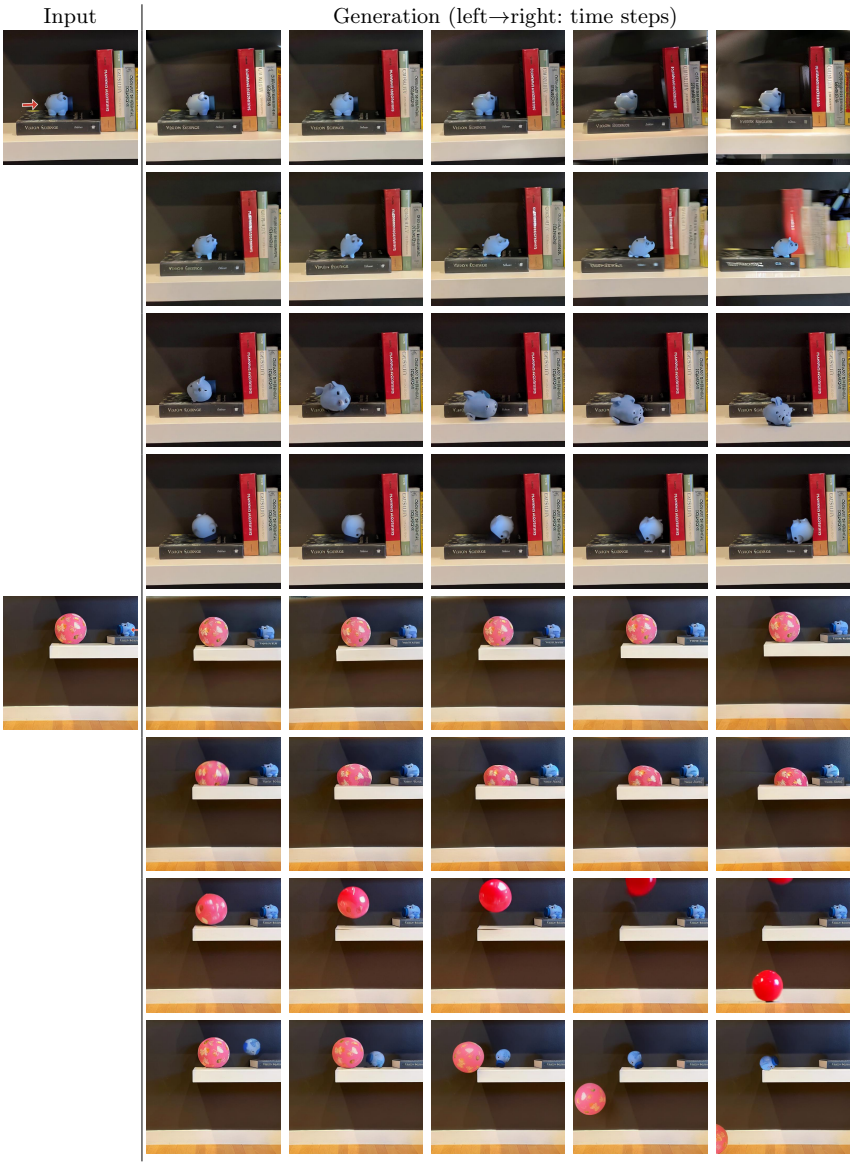

Fig. 11: Additional qualitative comparison against I2V generative models: DynamiCrafter(**DC**) [7], I2VGen-XL(**VGen**) [10], **SEINE** [1].

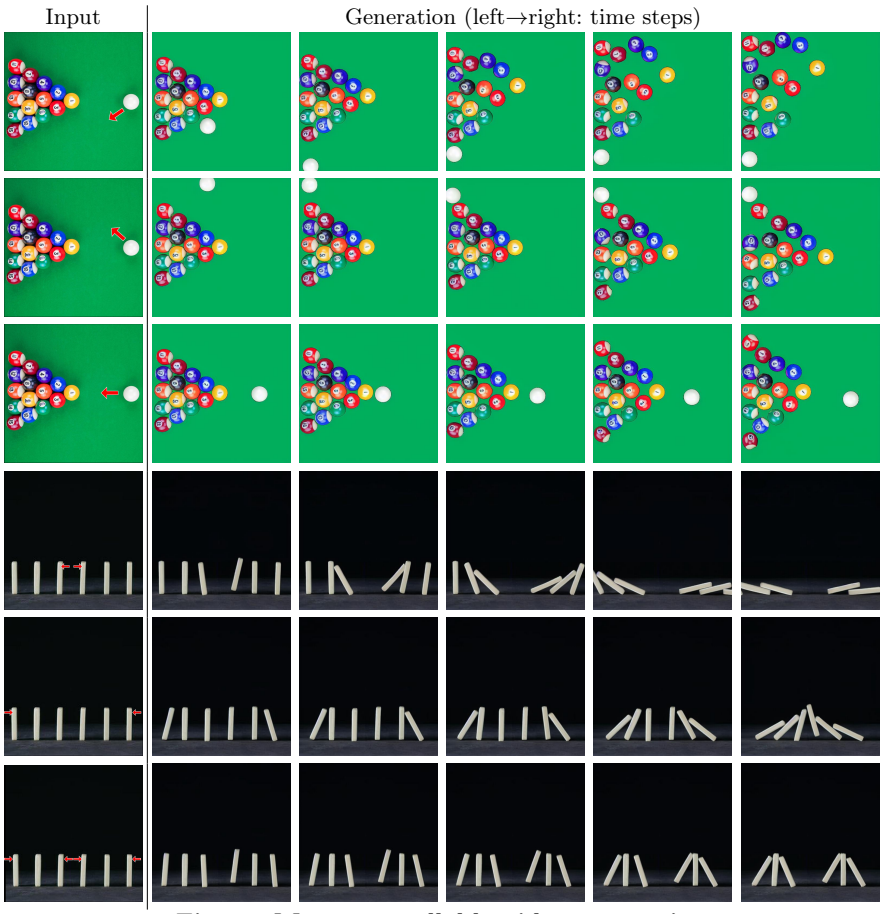

Fig. 12: More controllable video generation.

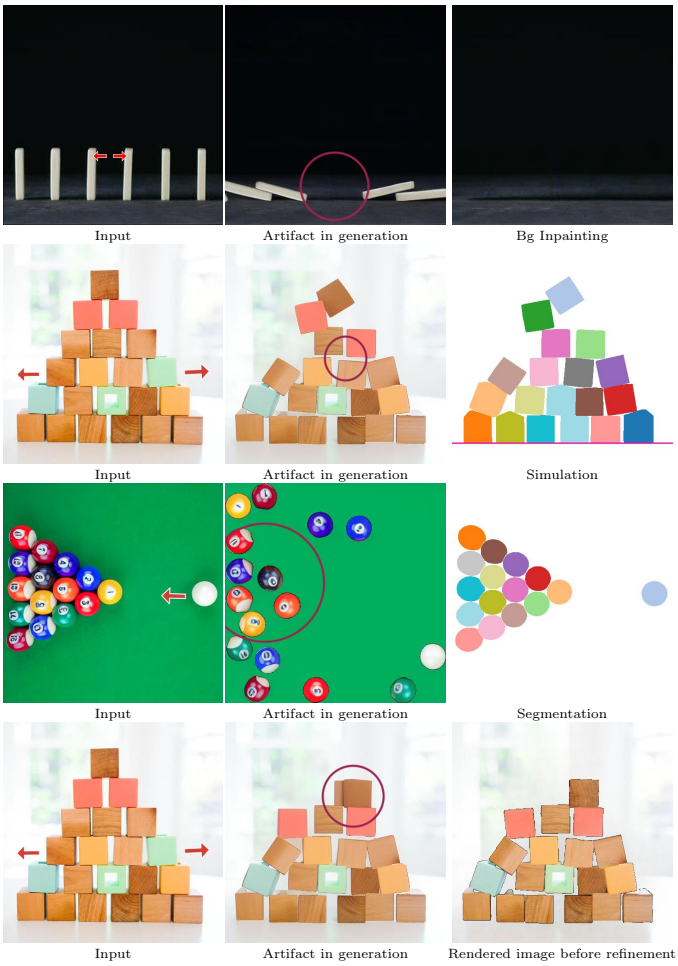

Fig. 13: Limitation analysis. We showcase 3 different examples of our current method's limitation. The left column shows the input image, the middle column shows the sampled frame from the generation video with artifacts, the right column shows the underlying reason for the artifact.

References

1. Chen, X., Wang, Y., Zhang, L., Zhuang, S., Ma, X., Yu, J., Wang, Y., Lin, D., Qiao, Y., Liu, Z.: Seine: Short-to-long video diffusion model for generative transition and prediction. arXiv preprint arXiv:2310.20700 (2023)
2. Fu, X., Yin, W., Hu, M., Wang, K., Ma, Y., Tan, P., Shen, S., Lin, D., Long, X.: Geowizard: Unleashing the diffusion priors for 3d geometry estimation from a single image. arXiv preprint arXiv:2403.12013 (2024)
3. Geng, D., Owens, A.: Motion guidance: Diffusion-based image editing with differentiable motion estimators. In: ICLR (2024)
4. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. NeurIPS (2020)
5. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
6. Ren, T., Liu, S., Zeng, A., Lin, J., Li, K., Cao, H., Chen, J., Huang, X., Chen, Y., Yan, F., Zeng, Z., Zhang, H., Li, F., Yang, J., Li, H., Jiang, Q., Zhang, L.: Grounded sam: Assembling open-world models for diverse visual tasks (2024)
7. Xing, J., Xia, M., Zhang, Y., Chen, H., Yu, W., Liu, H., Wang, X., Wong, T.T., Shan, Y.: Dynamicrafter: Animating open-domain images with video diffusion priors. arXiv preprint arXiv:2310.12190 (2023)
8. Yu, T., Feng, R., Feng, R., Liu, J., Jin, X., Zeng, W., Chen, Z.: Inpaint anything: Segment anything meets image inpainting. arXiv preprint arXiv:2304.06790 (2023)
9. Zhai, A.J., Shen, Y., Chen, E., Wang, G., Wang, X., Wang, S., Wang, X., Guan, K., Wang, S.: Physical property understanding from language-embedded feature fields. In: CVPR (2024)
10. Zhang, S., Wang, J., Zhang, Y., Zhao, K., Yuan, H., Qing, Z., Wang, X., Zhao, D., Zhou, J.: I2vgen-xl: High-quality image-to-video synthesis via cascaded diffusion models. arXiv preprint arXiv:2311.04145 (2023)