Appendix (Supplementary Material)

Please find the video visualizations on our project website: https://flymin.github.io/magicdrive3d/

A. Further Discussion on Differences from Reconstruction Works

The main focus of this work is the generation of non-existent 3D scenes, which distinguishes it from the reconstruction of existing scenes (such as [41, 50]). While 3D scene reconstruction also leverages video generation models to enhance quality [24, 42], its goal is to produce novel views of the same scene from a given set of camera angles. In contrast, our model can directly generate entirely new 3D scenes through latent variable sampling after training. The key difference is that our approach enables the creation of entirely new scenes, a capability that reconstruction methods cannot achieve (as shown in Table 1). Specifically, our model can generate novel scenes that, while adhering to control conditions, allow for the updating of objects and environmental features, such as buildings, across different samples. Figures 1 and 6 demonstrate our method's ability to construct diverse 3D scenes based on varying conditions. We hope this discussion helps readers gain a clearer understanding of the contribution of this work.

B. More Implementation Details

For video generation, we train our generator based on the pre-trained street view image generation model from Gao et al. [12]. By adding the proposed relative pose control, we train 4 epochs (77040 steps) on the nuScenes training set with a learning rate of $8e^{-5}$. We follow the settings for 7-frame videos described in Gao et al. [12], using 224×400 for each view but extending to $T = 16$ frames. Consequently, for 3DGS generation, we select $t = 8$ as the canonical space. Except we change the first 500 steps to optimize $(s_{c,t}, b_{c,t})$ for each view and $\lambda_{\text{reg}_o} = 1.0$, other settings are the same as 3DGS.

For the monocular depth model, we use ZoeDepth [4]. Although it is trained for metric depth estimation, due to domain differences, raw estimation is not usable, as shown in Section 5.5 and Figure 4. Methodologically, *MagicDrive3D* does not rely on a specific depth estimation model. Better estimations can further improve our scene generation quality.

Since GS only supports perspective rendering, to stitch the view for panorama, we use code provided by https://github.com/timy90022/Perspective-and-Equirectangular to transform from perspective to equirectangular.

All our experiments are conducted with NVIDIA V100 32GB GPUs. The generation of a single scene takes about 2 minutes for video generation and 30 minutes for GS generation. For reference, 3DGS reconstruction typically takes about 23 minutes for scenes of similar scales. Therefore, the proposed enhancement is efficient. As for rendering, there is no additional computation for our method compared with 3DGS.

As mentioned in Section 5.5, we use two testing scenarios to show the effectiveness of the enhanced GS pipeline as an ablation study, *i.e.*, 360° and vary-t. We use one testing trajectory as an example to illustrate the training/testing view splits in Table I.

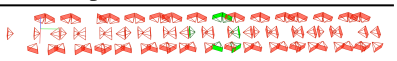
name	#test	#train	camera poses
360°	6	90	
vary-t	12	84	

Table I. Two settings for ablation study on the enhanced GS pipeline. Testing views are in green while training views are in red.

C. Optimization Flow

We demonstrate the overall optimization flow of the proposed FTGS in Algorithm 1. Line 2 is the first optimization of monocular depths. Lines 4-8 refer to the second optimization of the monocular depths. Lines 10-16 are the main loop for FTGS, where we consider temporal offsets on Gaussians, camera pose optimization for local inconsistency, and AEs for appearance discrepancies among views.

D. More Ablation Study

Ablation on Offset Choice for Fault-Tolerant GS. In addition to the overall module ablation, we observe that for Fault-Tolerant GS, beyond the Gaussians' center coordinates, their attributes (including anisotropic covariance, opacity, and har-

Methods	L1 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS	0.0733	19.7514	0.5210	0.4496
Features offset	0.0624	20.9882	0.5940	0.3463
Cov. offset	0.0632	20.8133	0.5854	0.3626
Opacity offset	0.0656	20.5332	0.5733	0.3845
Ours (xyz offset)	**0.0546**	**21.9428**	**0.6288**	**0.2759**

Table II. Ablating comparison with offsets on anisotropic covariance (Cov.), opacity, and harmonic coefficients (Features) properties in GS. We randomly sample 10 scenes from the nuScenes validation set for experiments and apply color correction (cc) to all the renderings.

monic coefficients) can also be utilized to address local inconsistencies. To verify the effects of different choices, we randomly select 10 scenes from the nuScenes validation set for experimentation. As shown in Table II, the results obtained by adding offsets to the center coordinates (xyz) are the best. This aligns with our observation that local inconsistencies in the generated views occur primarily in the shape of objects, and thus, xyz displacements can most effectively resolve these inconsistencies.

Qualitative Comparison with 3DGS baseline. Figure I shows the results of reconstructing the generated video directly using the 3DGS algorithm. As discussed in Section 4.3, 3DGS tends to amplify artifacts in the generated view, whereas our improved FTGS pipeline effectively controls the impact of these artifacts, ensuring the quality of the 3D scene generation.

E. More Reconstruction Baseline

Focusing on dynamic scenes, 4DGS [39] introduces comprehensive improvements over 3DGS and achieves notable results. Therefore, we replace Fault-Tolerant GS with 4DGS. As shown in Table III, by incorporating non-rigid dynamics, 4DGS already performs better than 3DGS. However, in our task, 4DGS underperforms compared to our Fault-Tolerant GS. Based on the results in Table II, we hypothesize that our FTGS algorithm only needs to address local inconsistency caused by content inconsistency. Allocating excessive degrees-of-freedom at the representational level may hinder model convergence, thus 4DGS does not yield better results in our scenarios.

Algorithm 1 Enhanced Fault-Tolerant Gaussian Splatting (FTGS)

Require: camera views $\{\mathcal{I}_i\}$, camera parameters $\{\mathbf{P}_i^0\}$, monocular depth $\{\mathcal{D}_i\}$, optimization steps for depth s_D, camera pose s_{C}, and GS s_{GS}
Ensure: FTGS of the scene $\{\boldsymbol{\mu}_p, \boldsymbol{\mu}_p^o, \boldsymbol{\Sigma}_p, \text{SH}_p\}$, and optimized camera pose $\{\mathbf{P}_i^0\}$
$\mathcal{P}_{SfM}$ = PCD from SfM
Optimize $(s_{c,t}, b_{c,t})$ with $\mathcal{P}_{\text{SfM}}$ for each $\{c, t\}$
Random initialize AEs $\{\mathbf{e}_i\}$
for step in $1, \ldots, s_D$ **do**
    Random pick one view $\mathcal{I}_i$
    $\mathcal{L} = \mathcal{L}_{\text{AEGS}}(\mathcal{I}_i, \mathcal{I}_i^r, \mathbf{e}_i)$
    Update $(s, b), \mathbf{e}_i, \boldsymbol{\Sigma}$, SH with $\nabla\mathcal{L}$
end for
Initialize $\boldsymbol{\mu}$ with (s, b) and $\mathcal{D}$
for step in $s_D, \ldots, s_{\text{GS}}$ **do**
    Random pick one view $\mathcal{I}_i$ and get its t
    $\mathcal{L} = \mathcal{L}_{\text{FTGS}}(\mathcal{I}_i, \mathcal{I}_i^r, \mathbf{e}_i, \boldsymbol{\mu}^o(t))$
    Update $\boldsymbol{\mu}, \boldsymbol{\mu}^o(t), \mathbf{e}_i, \boldsymbol{\Sigma}$, SH with $\nabla\mathcal{L}$
    if step $> s_{\text{C}}$ **then**
        Update $\mathbf{P}_i^0$ with $\nabla\mathcal{L}$
    end if
end for

Original 3DGS (w/ nuScenes dataset)

Ours (w/ nuScenes dataset)

Figure I. Qualitative Comparison with 3DGS baseline on the same scene shown in Figure 5. With our improved FTGS pipeline, the generated 3DGS contains fewer floaters, leading to higher-quality rendering.

Methods	L1 ↓	PSNR ↑	SSIM ↑	LPIPS ↓
3DGS	0.0733	19.7514	0.5210	0.4496
4DGS	0.0601	21.1195	0.5892	0.4475
Ours	**0.0546**	**21.9428**	**0.6288**	**0.2759**

Table III. Comparison with 4DGS [39]. We randomly sample 10 scenes from the nuScenes validation set for experiments and apply color correction (cc) to all the renderings.

F. Limitation and Future Work

As a data-centric method, *MagicDrive3D* sometimes struggles to generate complex objects like pedestrians, whose appearances are intricate. Additionally, areas with much texture detail (e.g., road fences) or small spatial features (e.g., light poles) are occasionally poorly generated due to limitations in the 3DGS method. We also observe that the exposure may not be consistent if the overlapping area between views cannot provide enough clues to infer a unified exposure in a single scene. Future work may focus on addressing these challenges and further improving the quality and robustness of generated 3D scenes.

G. Comparison with Simple Baselines

As shown in Figure II, we further compare *MagicDrive3D* with two baselines, *i.e.*, LucidDreamer [9] and WonderJourney [44]. The former method has been proposed recently and takes text description as the only condition. Thus, it is hard to generate photo-realistic street scenes. When providing multi-view video frames from nuScenes with known camera poses, their pipeline fails to reconstruct. We suppose the reason is limited overlaps and errors from depth estimation. As suggested by the released code, we changed the image generation model to `lllyasviel/control_v11p_sd15_inpaint` for inpainting by providing a nuScenes image, *i.e.*, Figure IIa. However, due to the lack of controllability, the results from LucidDream (*e.g.*, Figure IIb) are unsatisfactory. On the other hand, due to the lack of control over objects within the scene, WonderJourney struggles to generate coherent scenes. Inpainting-based methods like the two above exhibit a pronounced sense of patches and face significant challenges in achieving 360° coverage.

Figure IId further shows directly stitching real data. It is also bad due to the limited overlaps between views. On the contrary, the scene generated from *MagicDrive3D* can render a continuous panorama, as shown in Figure IIe, which is also controllable through multiple conditions.

Note that, panorama generation is only one of the applications of our generated scenes. We show them just for convenient qualitative comparison within the paper. Since our scene generation contains geometric information, it can be rendered from any camera view, as shown in Figure 5.

H. Implementation Detail of Appearance Embedding

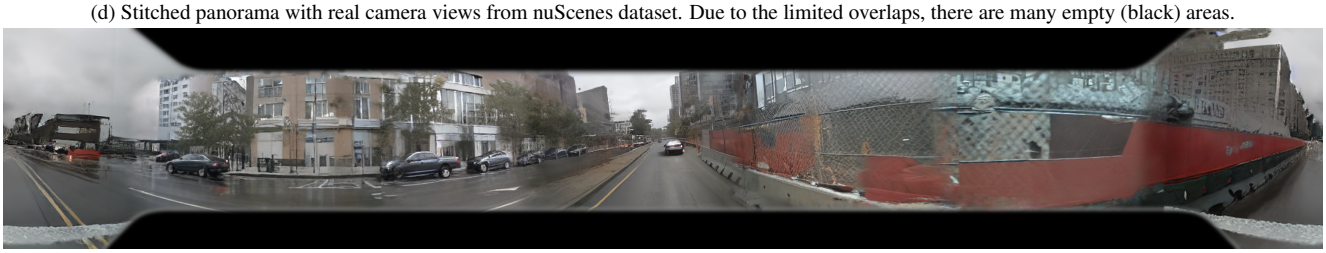
We show in Figure III the detailed architecture of the CNN used in our appearance modeling. The AE map is $32\times$ smaller than the input image to reduce the computational cost. Hence, we first downsample the input image by $32\times$. Then, we use 3×3 convolution for feature extraction and pixel shuffle for upsampling. Each convolution layer is activated by ReLU.

(a) Conditional image to LucidDreamer [9]

(b) Scene generated by LucidDreamer [9], with text "*A driving scene in the city from the front camera of the vehicle. A bus on the right side. There is a bridge overhead. There is a railing in the center of the left road. Some vehicles ahead*"

(c) Scene generated by WonderJourney [44]. We set the focal length to be the same as the conditional image. WonderJourney cannot control road semantics (many objects are physically implausible) and fails in loop closure for 360° scene generation. Conditions are the same as Figure IIb.

(d) Stitched panorama with real camera views from nuScenes dataset. Due to the limited overlaps, there are many empty (black) areas.



(e) Panorama from *MagicDrive3D*. The scene is generated with the same object boxes and BEV map as Figure IId, but turned to "*rainy day*".

Figure II. Comparison with two baselines (LucidDreamer [9] and WonderJourney [44]) and direct stitching real images.

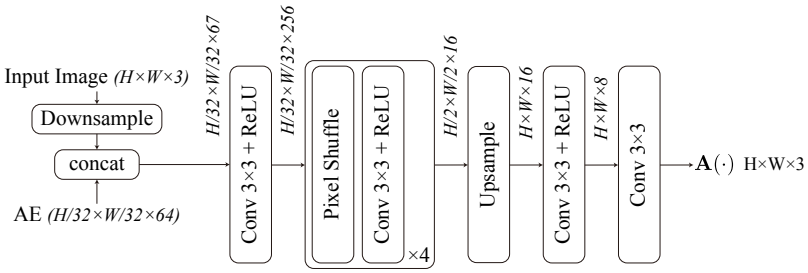


Figure III. The CNN architecture of appearance modeling, as introduced in Section 4.3.

I. Broader Impacts

The implementation of *MagicDrive3D* in controllable 3D street scene generation could potentially revolutionize the autonomous driving industry. By creating detailed 3D scenarios, self-driving vehicles can be trained more effectively and efficiently for real-world applications, thereby leading to improved safety and accuracy. Moreover, it could potentially provide realistic simulations for human-operated vehicle testing and training, thus contributing to reducing the occurrence of accidents on the roads while enhancing driver expertise. In the broader scope, *MagicDrive3D* could be of considerable value to the virtual reality industry and video gaming industry, enabling these sectors to generate more lifelike 3D scenes and intricate gaming experiences.

On the downside, the development and application of such advanced technology could lead to certain unwanted scenarios. For instance, the increased automation in industries, driven by the potential of this technology, could lead to job losses for

(a) original view (b) shift left 1.5m. (c) shift left 3m. (d) shift left 4.5m.

Figure IV. We show renderings by shifting the camera up to 4.5 meters left from the front-facing camera pose. The generated 3DGS keeps good quality in 3 meters range.

drivers and other related professionals as their roles become automated. A societal transition will be needed to avoid negative impacts on employment levels and the fairness of wealth distribution.

J. More Qualitative Results

We show more generated street scenes from *MagicDrive3D* in Figure V and Figure VI. We also show the rendering with large camera movement in Figure IV. The generated 3DGS keeps good quality in 3 meters range to the left/right of the input trajectory.

K. Quantitative Comparison on Controllability

Section 5.3 already show that the data generated from *MagicDrive3D* can well support perception model training, which indicate that the generated scenes accurately reflect the given conditions. Besides, we adopt the benchmark from [7] to further test the video quality and controllability for our method. As shown in Table IV, *MagicDrive3D* has comparable controllability as [12], but better video generation quality as indicated by FVD. This is reasonable since we mainly focus on improving camera pose control of each video generation. Thus, the generated videos are better aligned with the real ego trajectories, as presented in Section 4.2.

Method	FVD↓	mAP↑	mIoU↑
MagicDrive [12]	218.12	11.86	18.34
MagicDrive3D	210.40	12.05	18.27

Table IV. Generation quality and controllability comparison with [12], using benchmark from [7].

L. Pipeline Robustness and Overall Success Rate

Our method leverages camera pose control in video generation and monocular depth, thereby mitigating the error in pose estimation and sparse point cloud from SfM, as discussed in Section 4.3. With known camera poses, SfM will not crash at all. SfM provides only 100-200 points with high-quality depth scale information, while the pipeline fails only when the depth scale is far from correct (e.g., on negative depth). The success rate of the entire pipeline is 84.09% on nuScenes validation set. As for reference, Cosmos-Predict1-7B-Text2World has 62.6% success rate on SfM [1].

Figure V. Generated street scenes from *MagicDrive3D*. We adopt control signals from the nuScenes validation set. We crop the center part for better visualization.

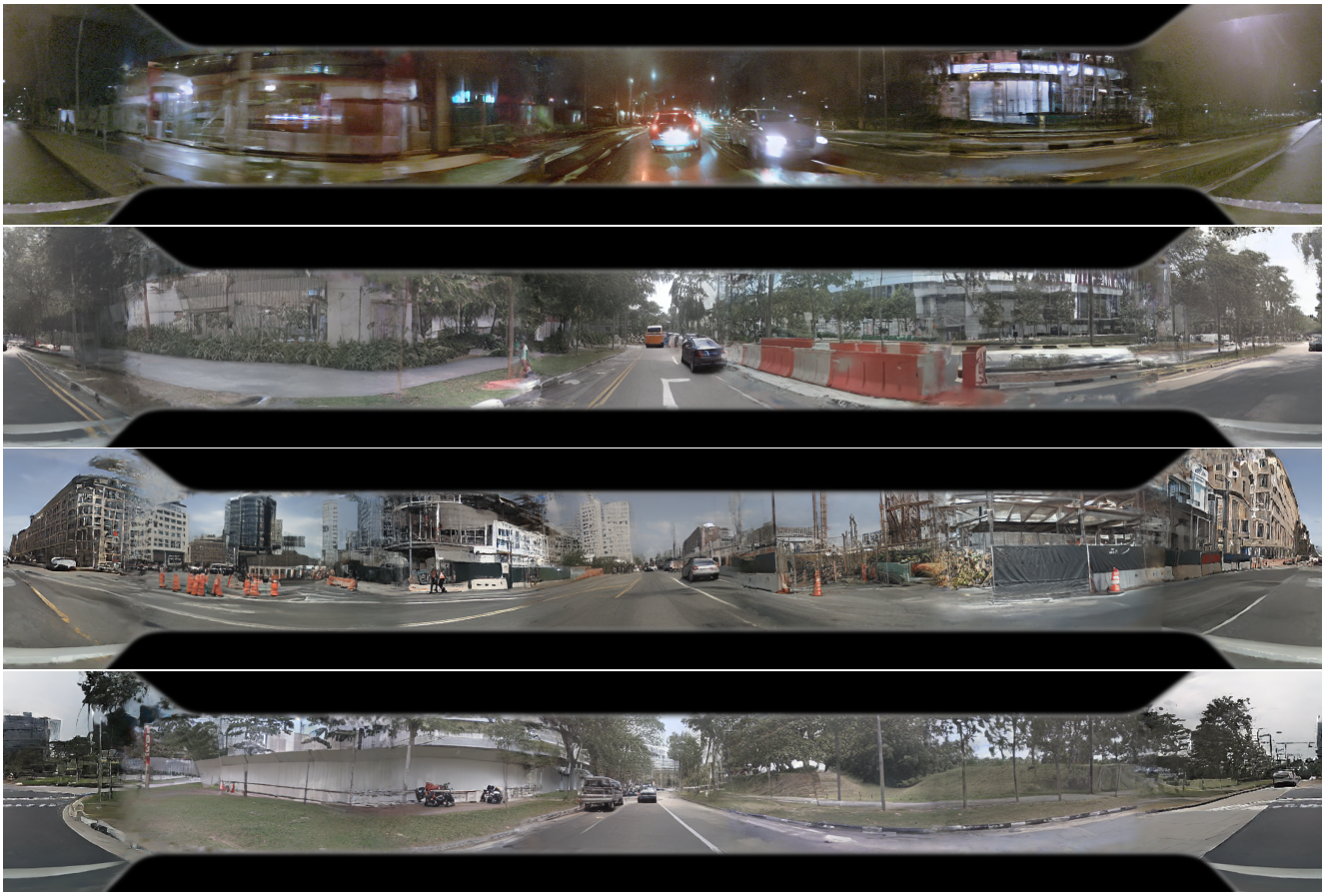


Figure VI. Generated street scenes from *MagicDrive3D*. We adopt control signals from nuScenes validation set. The black regions are not fully covered, constrained by the camera's FOV.