A Video Demonstrations

For a more extensive visual presentation of our findings, including generated videos, please follow this link: https://oooolga.github.io/ctrl-v.github.io/. Code can be found here: https://github.com/oooolga/Ctrl-V.

B Notations

B.1 SVD Scaling Function Definitions

As presented in Equation 2, λ_{skip}, λ_{out}, λ_{in} and λ_{noise} are scaling functions where λ_{in} and λ_{out} scale the input and output magnitudes of the neural network $\mathbb{U}_\theta$ being trained, λ_{skip} modulates the skip connection, and λ_{noise} maps noise level σ_t into a conditioning input for $\mathbb{U}_\theta$. More detailed information on these functions can be found in the work of Karras et al. (2022).

Unless otherwise specified, we use the SVD definitions as presented in (Blattmann et al., 2023a). The mathematical formulations for the scaling functions are thus:

$$\begin{aligned} \lambda_{\text{out}}(\sigma) &= \frac{-\sigma}{\sqrt{\sigma^2+1}} & \lambda_{\text{in}}(\sigma) &= \frac{1}{\sqrt{\sigma^2+1}} \\ \lambda_{\text{skip}}(\sigma) &= (\sigma^2+1)^{-1} & \lambda_{\text{noise}}(\sigma) &= \frac{\log\sigma}{4} \end{aligned} \tag{3}$$

B.2 Ctrl-V Notations

In this section, we provide a table of definition for all of the Symbols notations used in our work.

Symbol	Appearance	Definition
$\mathcal{E}$	Section 3.1, Figure 2	the off-the-shelf VAE Image Encoder used by SVD
$\mathcal{D}$	Section 3.1, Figure 2	the off-the-shelf VAE Image Decoder used by SVD
$\mathcal{C}_\xi$	Figure 2	modified ControlNet module in Box2Video
$\mathbb{D}_\theta$	Equation 2	denoiser network of the SVD backbone
$\mathbb{U}_\theta$	Equation 1, Equation 2	the main UNet component of $\mathbb{D}$
$\boldsymbol{f}$	Section 3.1	a sequence of frames
$\boldsymbol{f}^{(i)}$	Section 3.1	the i^{th} frame in $\boldsymbol{f}$
$\boldsymbol{z}$	Section 3.1	$\boldsymbol{z} = \mathcal{E}(\boldsymbol{f})$. a sequence of frames encoded by the VAE encoder
$\boldsymbol{z}^{(i)}$	Section 3.1	$\boldsymbol{z}^{(i)} = \mathcal{E}(\boldsymbol{f}^{(i)})$. the encoded i^{th} frame.
$\boldsymbol{z}^{(0)}$	Section 3.1, Equation 1	encoded first frame $\boldsymbol{f}^{(0)}$
$\boldsymbol{z}^{(0)}_{\text{pad}}$	Equation 1, Figure 2	padded $\boldsymbol{z}^{(0)}$ by repeating itself along the first dimension N times.
t	Section 3.1, Equation 1	level used by the EDM noise scheduler do determine the noise level
$\hat{\boldsymbol{z}}_t$	Section 3.1, Equation 1	latent representation of frames corrupted by noise level t
$\boldsymbol{c}^{(0)}$	Section 3.1, Equation 1	CLIP encoding of the first frame $\boldsymbol{f}^{(0)}$
$\boldsymbol{f}_{bbox}$	Figure 2	bounding box frames. Each frame contain pixel renderings of bounding boxes on a blank background. Used as conditioning for Box2Video
$\boldsymbol{b}$	Figure 2	latent representation of $\boldsymbol{f}_{bbox}$ (encoded by $\mathcal{E}$)
$\boldsymbol{b}^{(0)}$	Figure 2	latent representation of the first bounding box frame
$\boldsymbol{b}^{(N-1)}$	Figure 2	latent representation of the last bounding box frame
$\hat{\boldsymbol{b}}_t$	Figure 2	$\boldsymbol{b}$ corrupted by noise level t

Table 4: Glossary of terms.

C Implementation Details

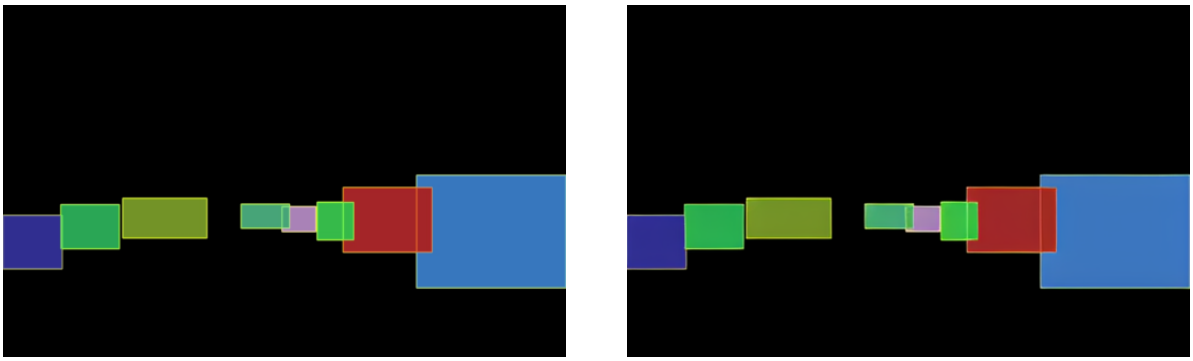
C.1 Step-by-Step Guide to Plotting bounding box Frames

We create bounding box plots for each frame according to the following steps:

1. Each unique track ID is randomly assigned a specific color, and the random color selected has values exceeding 50 across all RGB channels.
2. Each unique object class label is assigned a distinct color.
3. For each object, we fill the 2D bounding box region with its track ID's color at 75% transparency. This transparency allows the overlapping regions of the boxes to remain visible.
4. If 3D bounding box information is available for an object, we draw the 3D bounding boxes and outline them with the color corresponding to the object's class label.
5. If 3D bounding box information is unavailable, we outline the 2D bounding box with the color corresponding to the object's class label. Moreover, we mark an X at the rear face of each 3D bounding box for enhanced contextualization.
6. In certain experiments, we substitute bounding box plots with trajectory plots. To create a trajectory plot, we first compute the mid-points of its 2D bounding boxes. Subsequently, at each bounding box midpoint, we draw a circle with a diameter of 10 pixels, filled with the color corresponding to its track ID. Within this circle, we draw a smaller circle with a diameter of 5 pixels, filled with the color representing the object's class label.

C.2 bounding box Visualization and Reconstruction Analysis

In this work, we refrain from introducing new modules dedicated to encoding the bounding box frames. Instead, we leverage the encoder ($\mathcal{E}$) within the SVD's autoencoder framework. Our experiments demonstrate the efficacy of the SVD autoencoder in accurately encoding and decoding bounding box frames.



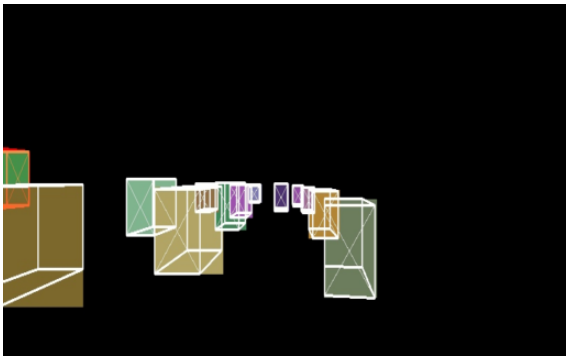
(a) original frame plot (b) reconstructed via SVD's VAE

Figure 6: Visualizing a BDD100K 2D-bbox frame created using the method described in C.1

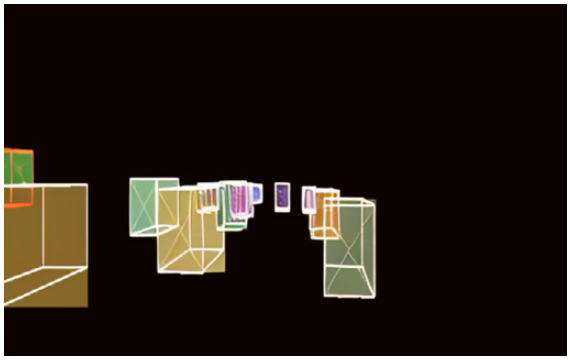
C.3 Dataset Configuration

KITTI: The downloadable content of the official KITTI dataset does not include bounding box label files for the testing set. As a result, we manually partition the dataset's training folder into two segments for our experiments. The training split encompasses samples '0000' to '0018', while the testing split consists of samples '0019' and '0020'.

Virtual-KITTI 2 (vKITTI): We've not found specification of the train/test split on the official website. During training, we exclude 'Scene20', and the models were trained on the remaining four scenes.



(a) original frame plot



(b) reconstructed via SVD's VAE

Figure 7: Visualizing a VKITTI 3D-bbox frame created using the method described in C.1

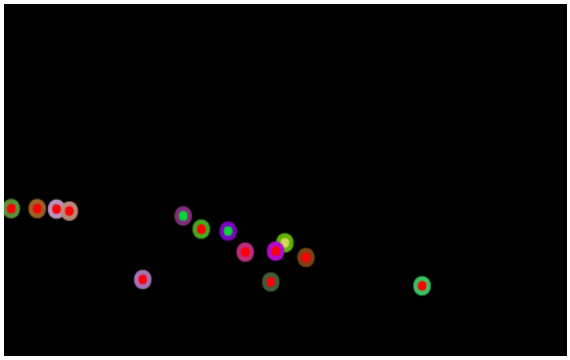


Figure 8: Visualizing a BDD100K trajectory frame created using the method described in C.1.

BDD100k: We adhere to the official train/validation split for our training and testing phases.

nuScenes: We adhere to the official train/validation split (700/150 scenes) for our training and testing phases.

In our experiments, we use a standardized frame size of 312×520 pixels and a clip length of 25 frames at a frame rate of 7 frames per second. All experiments are trained using the training set and assessed using the validation or test sets. Unless otherwise stated, the results and the visualizations provided are derived from the test sets.

C.4 Training Configuration

Our pretrained SVD models are adaptations from HuggingFace's diffuser library (von Platen et al., 2022). Specifically, we utilize the 'stable-video-diffusion-img2vid-xt' SVD variant in this project.

We have trained various types of bounding box generators in this work: 1. *3D bounding box generator (3-to-1)* 2. *2D bounding box generator (3-to-1)* 3. *2D bounding box-trajectory generator (3-to-1^{Traj})*.

The 3D bounding box generator anticipates frames using 3D bounding boxes, while the 2D generator does the same for 2D bounding boxes. A 1-to-1 model generates frames between the first and last bounding box frames, while a 3-to-1 model generates frames between the first 3 and last bounding box frames. Additionally, there's a bounding box-trajectory model where the last conditional frame is a trajectory frame instead of a bounding box frame.

For the bounding box generators trained on KITTI or vKITTI datasets, each model undergoes 5 epochs of training. The 2D bounding box generator (1-to-1) is trained on BDD100K for 33,000 iterations. Additionally, the 2D bounding box (3-to-1) and 2D bounding box-trajectory (3-to-1) models start their fine-tuning process for an additional 15,000 iterations from where the 2D bounding box generator (1-to-1)'s training ends. All of

Model	Submodule	Status	umber of Parameters
BBox Generator	VAE-Encoder	Frozen	34,163,592
	VAE-Decoder	Frozen	63,579,183
	CLIP-Image Encoder	Frozen	632,076,800
	ST Condition UNet	Trainable	1,524,623,082
Box2Video	VAE-Encoder	Frozen	34,163,592
	VAE-Decoder	Frozen	63,579,183
	CLIP-Image Encoder	Frozen	632,076,800
	ST Condition UNet	Frozen	1,524,623,082
	ControlNet	Trainable	680,946,897

Table 5: This table presents the parameter counts for the BBox Generator (2.25 billion) and Box2Video (2.94 billion) models, with a detailed submodule parameter breakdown.

these generation models are trained on a single A100-80G GPU with a batch size of 1. We also set gradient accumulation steps to 5 and employed AdamW (Loshchilov & Hutter, 2019) as our optimizer.

The fine-tuning and training procedures for the baseline Stable Diffusion and the ControlNet models are almost the same as of the bounding box generator model. The only difference is that we trained our baseline and ControlNet for 48,000 iterations on BDD100K.

C.5 Model Capacity

The number of parameters for each model, along with its submodule parameter counts, is provided in Table 5.

D Evaluation Details

D.1 Generation Quality Metrics: PSNR, SSIM, LPIPS and FVD

PSNR, or Peak signal-to-noise ratio, calculates the ratio between the maximum value of a signal and the noise that affects the fidelity of its representation. It is closely related to Mean Squared Error (MSE) loss. It can be in fact calculated by subtracting the log of MSE from a constant. It is long been used as a metric for image reconstruction quality for its ease of use and mathematical properties. However, PSNR suffers multiple issues as a metric for image generation tasks. Its insensitiveness to various distortion such as blurring, pepper noise, and mean-shifting means that image pairs that look very different to human can have very similar PSNR scores (Wang & Bovik, 2009; Wang et al., 2004a).

SSIM, or Structural similarity index measure, is a metric that aims to evaluate similarity in "structures" of image pairs – as opposed to absolute errors as in the cases of MSE and PSNR. Because the human visual system is more sensitive to structural distortions, this metric is more closely aligned with human evaluation compared to PSNR. However, some visually apparent distortions are not captured by the SSIM. Things like changing hue and brightness do not seem to affect SSIM much (Kotevski & Mitrevski, 2010) and Sharif et al. (2018) presents more examples of unpredictable ssim behaviours when presented with different distortions.

LPIPS, or Learned Perceptual Image Patch Similarity (Zhang et al., 2018), is a neural network based approach to accessing image similiarity. It does so by computing distance between some reprensentation of image patches. These representations are obtained by running image patches through some pre-defined neural network. LPIPS has been shown to math human perception well.

FVD, or Frechet Video Distance (Unterthiner et al., 2019), is a metric for generative video models. It is based on the Fréchet inception distance (FID) for images. Unlike the 3 other metrics that focuses on image frames and do not consider their temporal relationships, FVD is specifically designed for videos. It takes into consideration the visual quality, temporal coherence, and diversity of samples of videos generated by the

model under evaluation. The large scale human study carried out by the authors suggest FVD consistently agrees with human evaluation of videos.

D.2 Prediction Scores: Converting bounding box Frames to Binary Masks

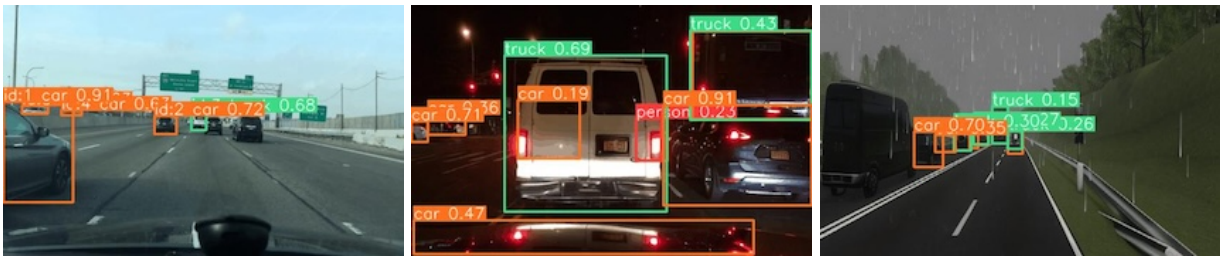
To convert ground-truth bounding box frames into binary masks, we combine all color channels and assign a value of 1 to all non-black pixels.

To transform generated bounding box frames into binary masks, we first need to perform post-processing on the frames. These frames originate from a network output, and although background pixels that seem black, they may not all be precisely zero. To prevent these pixels being detected as bounding box masks, we aggregate the pixel values across color channels, and for those channels whose sum is below 50, we convert them to black pixels. Subsequently, we convert the frames into binary masks by aggregating all channels and changing the colored pixel values to 1.

The maskIoU score measures the averaged ratio of the intersection area to the union area of the masks. The maskP score evaluates the averaged ratio of the intersection area to the predicted mask, while the maskR score quantifies the averaged ratio of the intersection area to the ground-truth mask.

Accurately evaluating bounding box generations remains challenging. We recognize that our current metrics (maskIoU, maskR and maskP), which rely on binary masks to evaluate the performance of bounding box generators, have limitations as they do not consider object tracking IDs (the colors of the bounding boxes). However, we still believe that the metric evaluations using binary masks can offer valuable insights into our model's performance.

D.3 Motion Control Assessment: YOLOv8 Object Detector Configurations



(a) Detection on ground-truth BDD frame. (b) Detection on generated BDD frame. (c) Detection on generated vKITTI frame.

Figure 9: YOLOv8 detections on BDD and vKITTI with confidence thresholds set to 0.1. The detection confidence scores are labeled on the detected bounding boxes.

We utilize the "yolov8x" variant of the YOLOv8 model, implemented by Ultralytics (Jocher et al., 2023), for object detection. We set the detector's Non-Maximum Suppression Intersection Over Union (NMS-IoU) threshold to be 0.35 across all experiments.

Since YOLOv8 was trained to detect objects from the MS COCO dataset (Lin et al., 2015), we adjust the class labels of our dataset to match the MS COCO labels using the following mappings:

The detection resolution is maintained at the same level as the generation/training resolution – 312×520.

YOLOv8 is a powerful tool, especially for object detection tasks. Its detection results are very impressive but not always perfect, which leaves rooms for errors when relying on its output to evaluate our model's performance. Two examples of YOLOv8's detection on generated and ground-truth BDD frames are illustrated in Figure 9.

In the ground-truth BDD frame detections, the network has missed detecting the two black SUVs on the right, even though the confidence threshold is set as low as 0.1 during detection.

KITTI	CAR–CAR; VAN–CAR; TRUCK–TRUCK; PEDESTRIAN–PERSON; PERSON–PERSON; CYCLIST–PERSON; TRAM–TRAIN
vKITTI	CAR–CAR; VAN–CAR; TRUCK–TRUCK; TRAM–TRAIN
BDD	PEDESTRIAN–PERSON; RIDER–PERSON; CAR–CAR; TRUCK–TRUCK; BUS– BUS; TRAIN–TRAIN
nuScenes	HUMAN.{ADULT, CHILD, CONSTRUCTION_WORKER, PERSONAL_MOBILITY, POLICE_OFFICER, WHEELCHAIR}–PERSON; VEHICLE.{BICYCLE, MOTORCYCLE}–PERSON; VEHICLE.{BUS, CONSTRUCTION, AMBULANCE, POLICE, TRAILER, TRUCK}–TRUCK; CAR–CAR

Table 6: Mappings of various dataset labels to MS COCO labels.

In the generated BDD frame detections, the network has falsely detected the reflections on the engine hood of the driving vehicle as a car. This is a common mislabeling we have observed across the experiments.

In the last example, we demonstrate that YOLOv8 can generate detections on virtual datasets such as vKITTI.

We have attempted to enhance our detection results across frames by using a tracker in conjunction with the YOLOv8 detector. Specifically, we use the BoT-SORT (Aharon et al., 2022) tracker. However, we've observed an increase in the precision but a significant drop in the recall of the detection scores. Thus, we've reverted to using only the YOLOv8 detection model to compute the AP metrics.

D.4 Motion Control Assessment: Average Precision Calculation

Average precision (AP) is the area under the recall-precision curve; it takes into account the trade-off between precision and recall at different confidence thresholds. It serves as a measure to evaluate how well the predicted or generated bounding boxes correspond to the actual ground-truth labels.

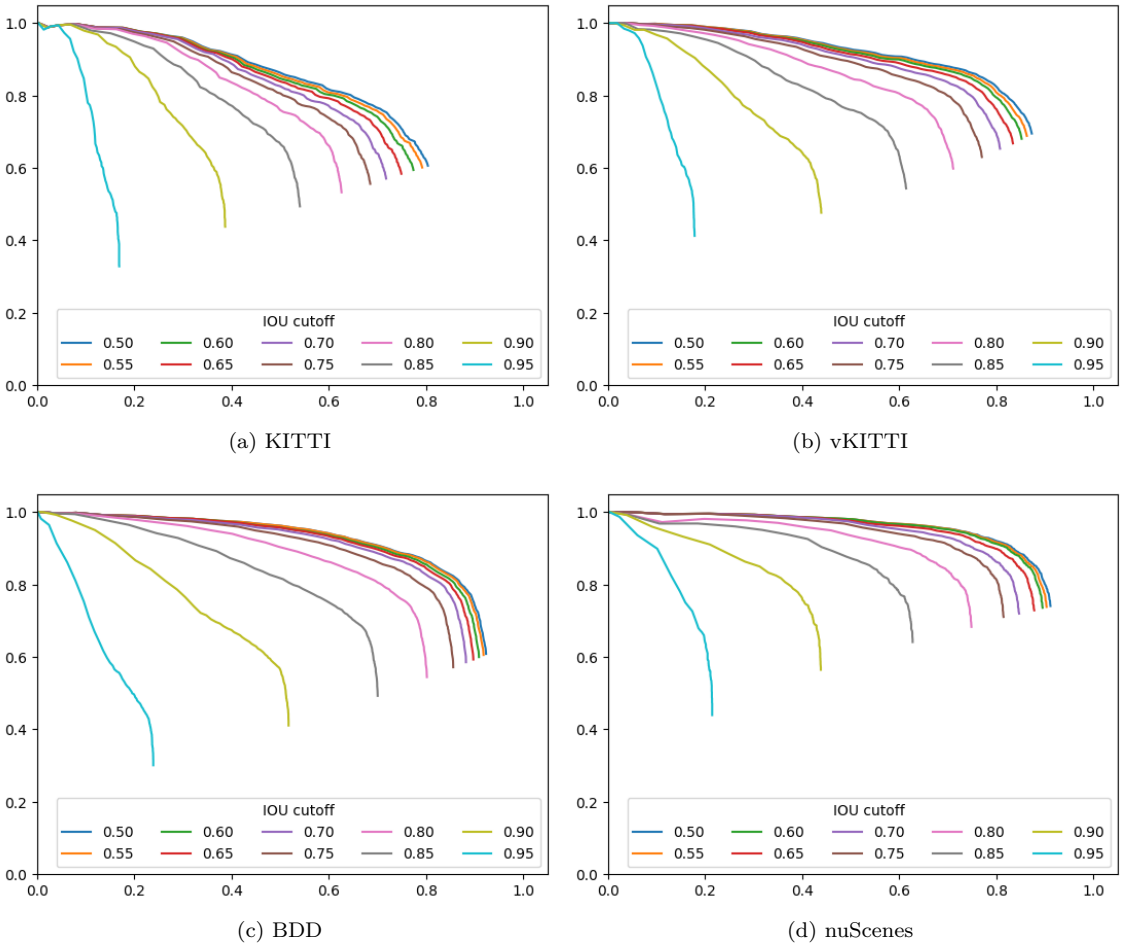
Prior to developing our custom AP metrics, we conduct experiments to assess YOLOv8's performance on our datasets. We evaluate YOLOv8 and YOLOv8+BoT-SORT on our three datasets using detection and tracking experiments. We employ Track-mAP to evaluate the alignment score between the detections on the ground-truth frames and actual ground-truth labels. The results are not satisfactory; the detection recall is low, suggesting a significant number of missed detections. This issue was exacerbated when using BoT-SORT for tracking. These results lead us to deviate from utilizing the ground-truth labeling and instead use the detections from the ground-truth frames as the labels for our mAP calculations.

We begin by employing a YOLOv8 detection model on both the generated and ground-truth videos, yielding a series of bounding box detections for each. When we generate bounding box detections on ground-truth videos, we keep the confidence cutoff fixed at 0.6.

Next, we calculate the AP scores at different IoU thresholds following the MS COCO protocol (Lin et al., 2015): compute 10 AP scores at IoU thresholds ranging from 0.5 to 0.95, in increments of 0.05. The AP score computed at threshold τ is referred to as $\text{AP}_{\tau\times 100}$. Mean average precision (mAP) is the average of all the AP scores. The computed results are reported in Table 3.

Figure 10 contains the precision-recall curves of the three datasets. The precision-recall curves are computed by dividing confidence cut-off into 100 intervals ranging from 0.01 to 1.00. Each curve presents the precision-recall trade-off by adjusting the confidence cutoff while keeping the IoU cut-off constant.

Our precision-recall curves typically lie above the diagonal line, indicating that our precision is generally higher than our recall. This further implies that if the detector identifies an object in our generated frame, there is a high likelihood that an object is detected in the same location in the ground-truth frame. On the



(a) KITTI (b) vKITTI

(c) BDD (d) nuScenes

Figure 10: Precision-recall curves: the x-axis represents the recall rate, and the y-axis represents the precision rate.

other hand, the lower recall rate implies that an object may not always be present (or detected) in the same location as identified by the detector in the ground-truth frame.

Several factors could have affected the recall rate:

- *Generation quality:* The quality of the generated images can influence the detectability of objects, thereby affecting the recall rate.
- *Detection error:* The performance of the detector has a direct impact on the recall rate. If there are many missed detections, the recall rate will also be high.
- *Data error:* This includes mislabeled samples from the dataset and our data preprocessing method. We limit the number of bounding boxes in a frame to a maximum of 15. Consequently, there may be missing bounding boxes for objects in the scene, which could affect our recall rate.

On another note, an ideal precision-recall curve would be a horizontal line at $y = 1$. This would indicate perfect precision and recall across all thresholds. Among the 3 experiments, our precision-recall curve for BDD is closest to this ideal line. One reason for this is that the BDD dataset contains significantly more data compared to the others, which can significantly boost our generation results.

E Generation Results

E.1 Stable Video Diffusion vs Box2Video: Generation Quality Comparison

The following are generation comparisons between the fine-tuned Stable Video Diffusion baseline model and the Box2Video conditioned on the ground-truth bounding box frames. All visualizations are generated using the BDD100k dataset.

(a) Stable Video Diffusion's generation at frame 13

(b) Box2Video's generation at frame 13

Figure 11: Comparison of an urban scene generation between the baseline Stable Video Diffusion (SVD) model and the teacher-forced Box2Video model.

(a) Stable Video Diffusion's generation at frame 13

(b) Box2Video's generation at frame 13

Figure 12: Comparison of a crowded intersection scene generation between the baseline Stable Video Diffusion (SVD) model and the teacher-forced Box2Video model.

(a) Stable Video Diffusion's generation at frame 13

(b) Box2Video's generation at frame 13

Figure 13: Comparison of a night scene generation between the baseline Stable Video Diffusion (SVD) model and the teacher-forced Box2Video model. Ground-truth bboxes are outlined.

E.2 BDD Result Visualizations: bounding box Generations and Motion-Controlled Video Generations

In the following section, we showcase a range of BDD generation results produced by our 3-to-1 generation pipeline in different scene scenarios, such as city, urban, highways, busy intersections and at night. Each visualization displays every 6th frame from a 25-frame clip, with the actual video generated at a frame rate of 5 fps. In the leftmost column labels, GT represents ground truth, GB represents generated bounding box frames, and GF represents generated frames.

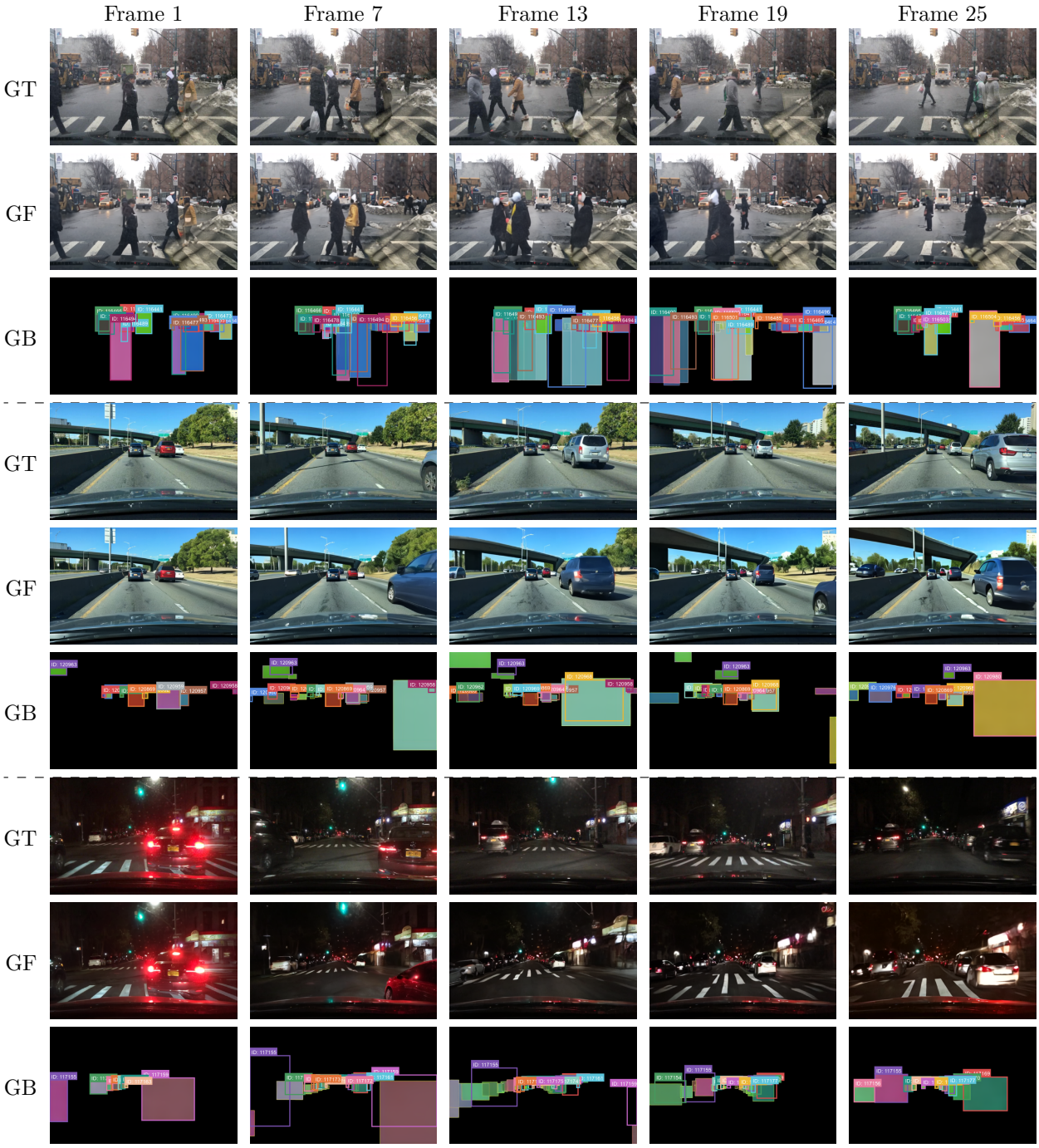


Figure 14: 2D bounding box frame generations and motion-controlled video generations for various scenes.

E.3 vKITTI Result Visualizations: 3D bounding box Generations and Motion Controlled Video Generations

In the following section, we showcase a range of vKITTI generation results produced by our 3-to-1 generation pipeline in different scene scenarios. In Figure 15, we showcase our generations on the vKITTI test split. However, it is worth noting that the vKITTI test split comprises samples from only one type of scene (unseen during training). Each visualization displays every 6th frame from a 25-frame clip, with the actual video generated at a frame rate of 7 fps. In the leftmost column labels, GT represents ground truth, GB represents generated bounding box frames, and GF represents generated frames.

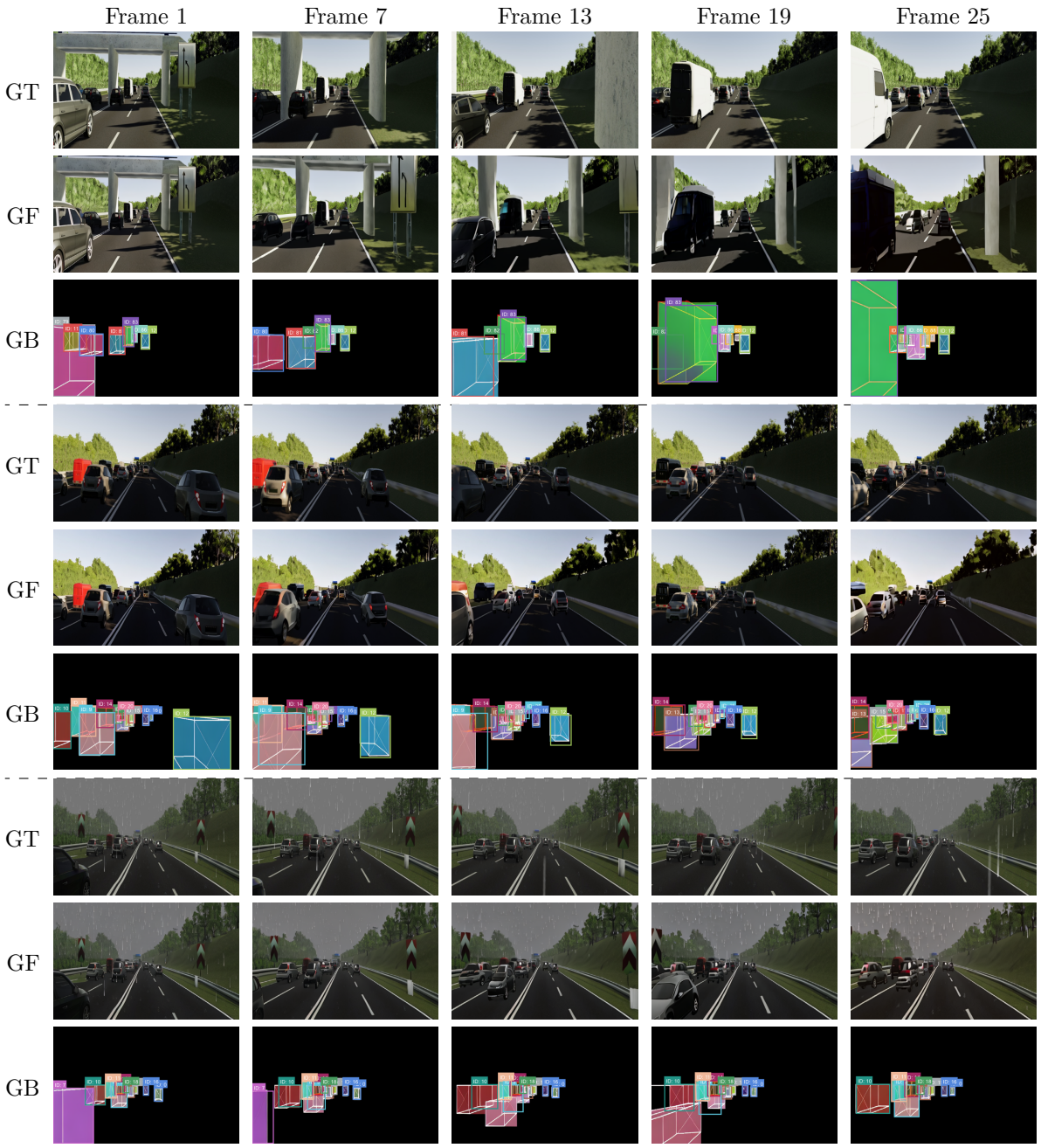


Figure 15: 3D bounding box generations and motion-controlled video generations for various scenes on vKITTI test-split.

E.4 KITTI Result Visualizations: bounding box Generations and Motion-Controlled Video Generations

In the following section, we showcase a range of KITTI generation results produced by our 3-to-1 generation pipeline in different scene scenarios, such as urban area, city streets and highway. Each visualization displays every 6th frame from a 25-frame clip, with the actual video generated at a frame rate of 7 fps. In the leftmost column labels, GT represents ground truth, GB represents generated bounding box frames, and GF represents generated frames.

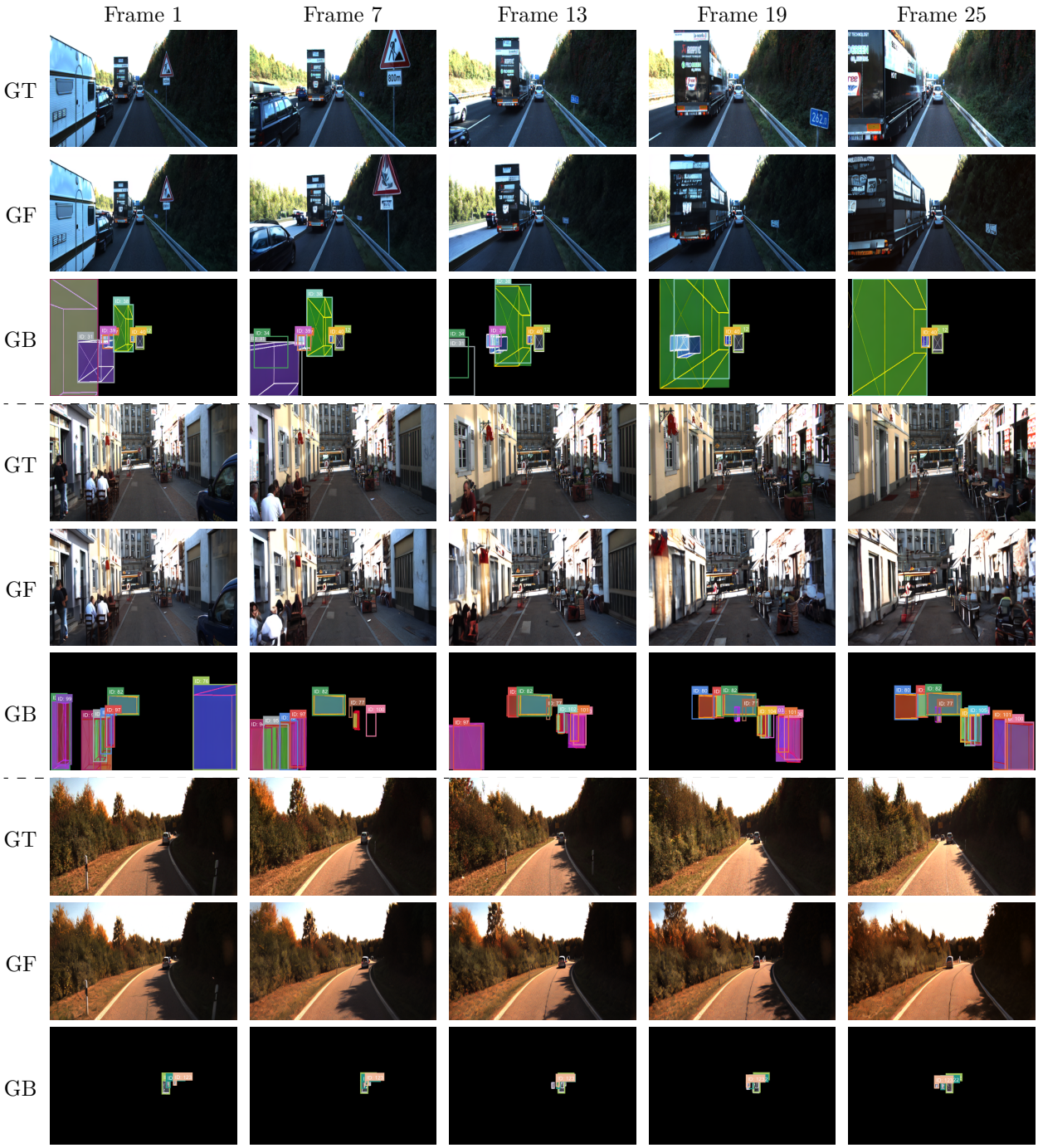


Figure 16: 3D bounding box Generations and motion-controlled video generations on KITTI test-split.

E.5 Nuscenes Result Visualization: bounding box Generations and Motion-Controlled Video Generation

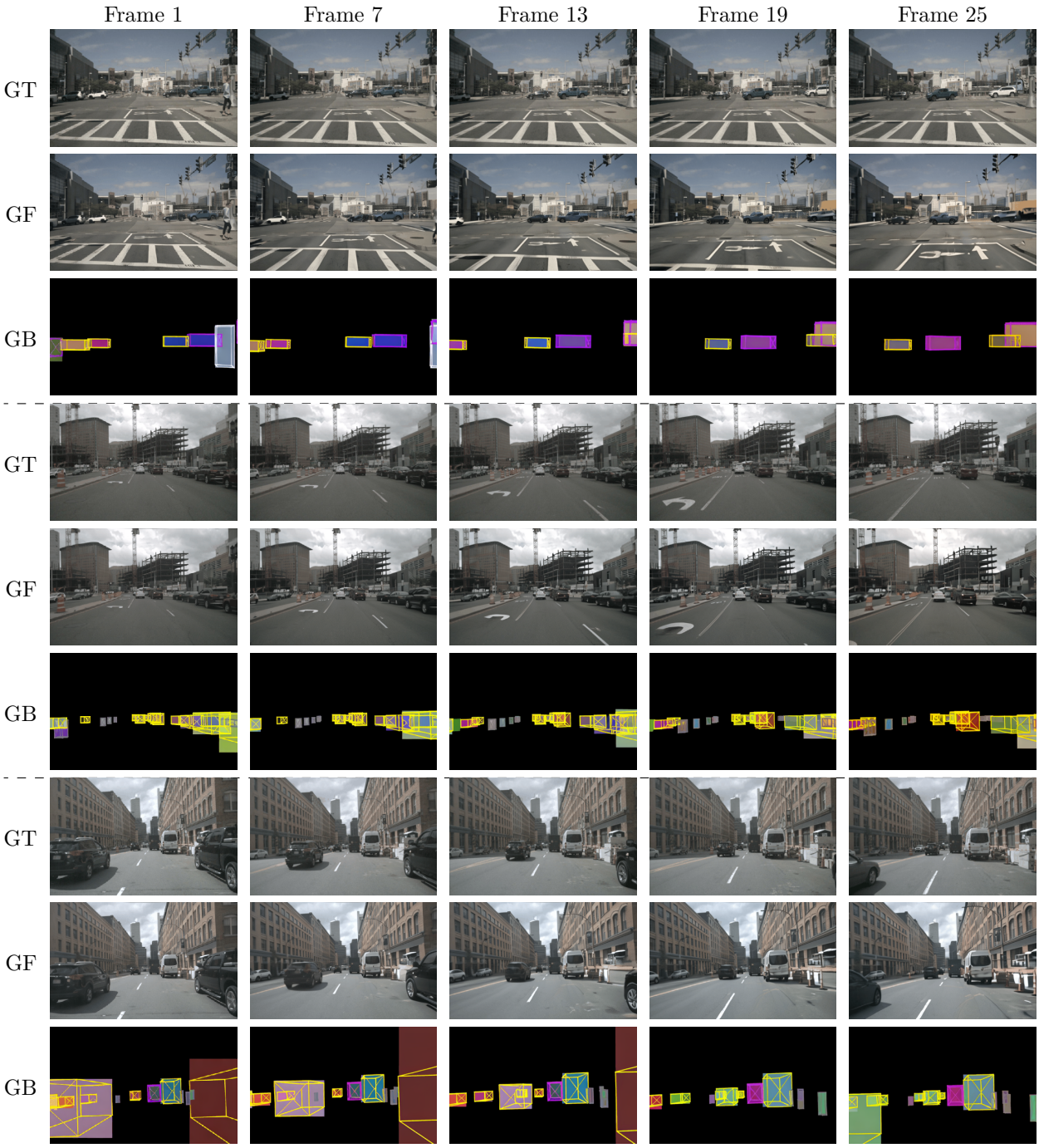


Figure 17: 3D bounding box generations and motion-controlled video generations for various scenes on NuScenes.

E.6 Special Scenarios: Turning

In the following section, we showcase a range of generation results with special scenarios, such as turning (day/night). These results are produced by our 3-to-1 generation pipeline. Each visualization displays every 6th frame from a 25-frame clip, with the actual video generated at a frame rate of 7 fps. In the leftmost column labels, GB represents generated bounding box frames, and GF represents generated frames.

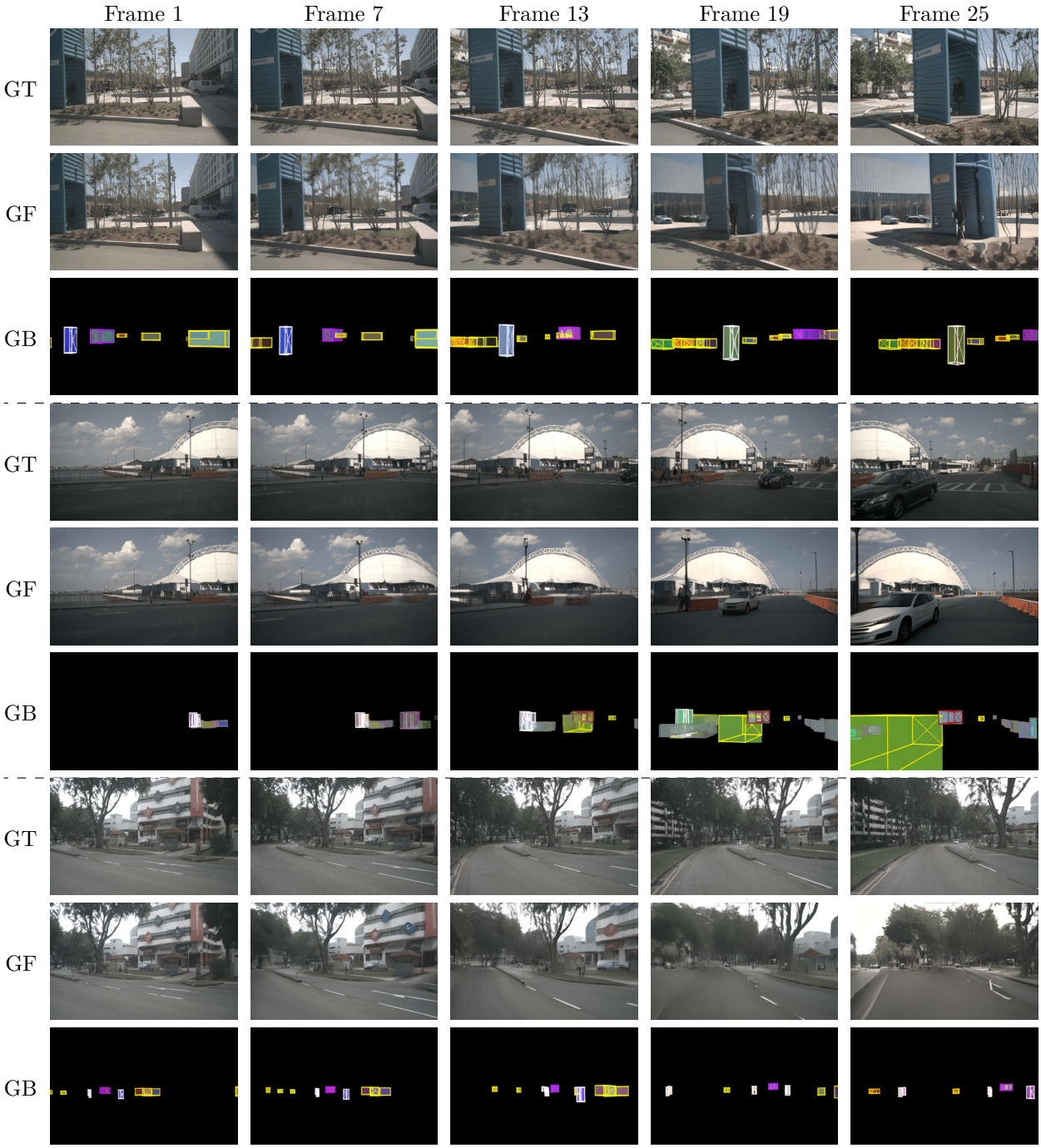


Figure 18: 3D bounding box generations and motion-controlled video generations for turning scenarios on NuScenes.

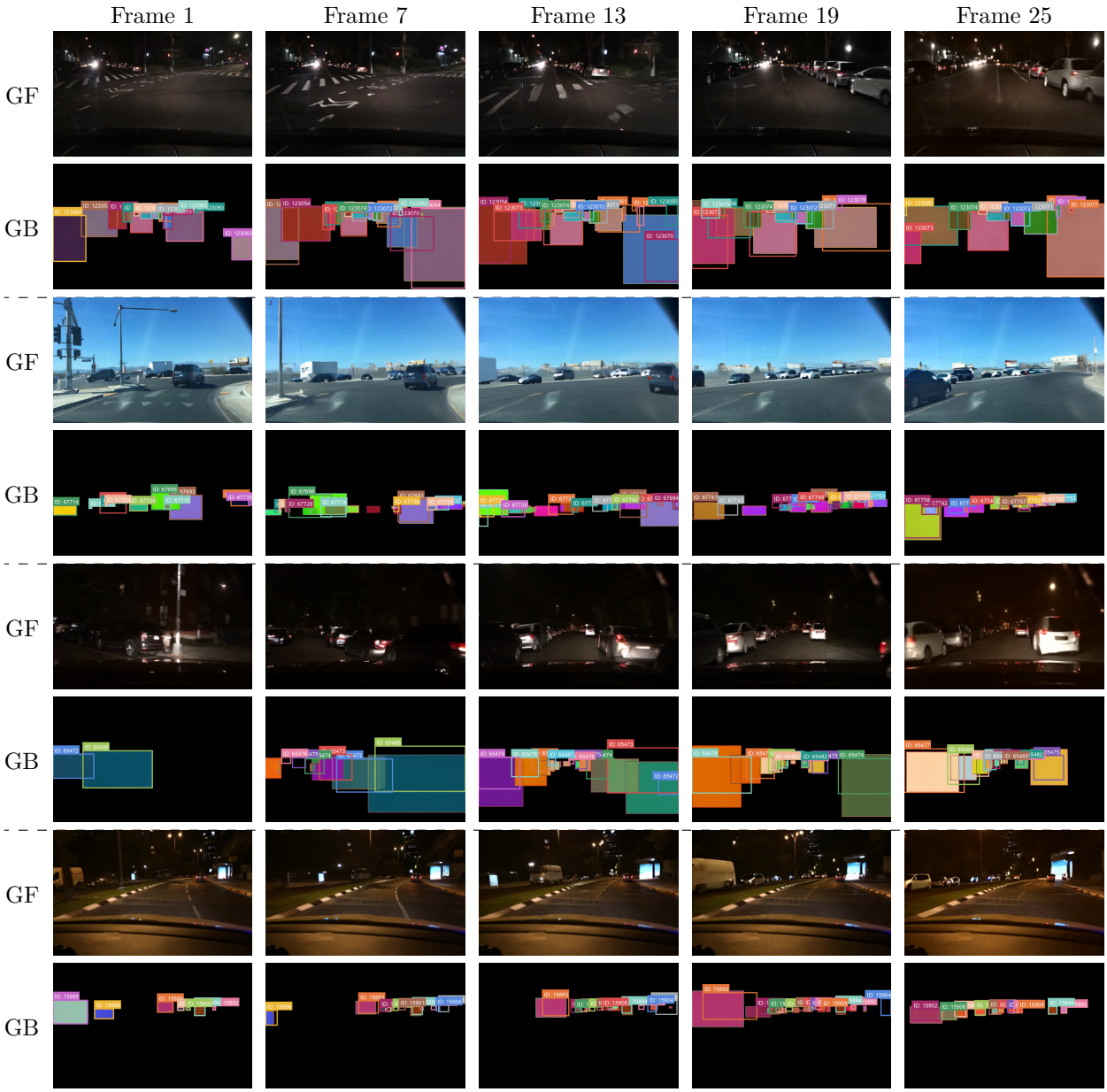


Figure 19: 2D bounding box generations and motion-controlled video generations for turning scenarios on BDD test-split.

E.7 Visualizing Motion Controlled Generation via Box2Video

Figure 20: A frame-by-frame visualization of Box2Video generation, conditioned on the ground-truth bbox frame sequence from the KITTI dataset, with conditions outlined in the plots.

E.8 BDD100K Results under Varying Conditioning Scenarios

We investigate the impact of the number of conditioning bounding box frames on the quality and alignment of the output to the ground truth. We train distinct models for each conditioning configuration. Our findings are summarized in Table 7 and Table 8.

Key insights include:

- Generation quality peaks when conditioned on the entire ground-truth bounding box trajectory.
- Quality metrics show slight disagreement on the worst-performing setting, with conditioning on either: first+last bounding box frames or first three frames.
- Comparing to the no-final-frame conditioning baseline, our results show that minimal conditioning with the central point of the final bounding box frame (termed "trajectory frame"; see Figure 8) significantly enhances generation quality.
- Table 8 demonstrates that incorporating the last bounding box frame significantly improves alignment between generated and ground-truth bounding box trajectories, as evidenced by the improved performance when conditioning on the first and last frames versus the first three frames.
- Figure 23 showcases two predicted bounding box trajectories generated by BBox Generator without conditioning on last-frame bounding box locations. Notably, despite inaccurate last-frame bounding boxes compared to ground-truth, the BBox Generator still predict convincing bounding box trajectories independently of last-frame bounding box locations.

Figure 21: A frame-by-frame visualization of Box2Video generation, conditioned on the ground-truth bbox frame sequence from the vKITTI dataset. The ground-truth bboxes are outlined in the plots.

- Figure 24 presents exemplary clips generated by the Box2Video model, conditioned on bounding box sequences derived without last-frame bounding box information.
- Figure 23 and Figure 24's visualization displays every 6th frame from a 25-frame clip, with the actual video generated at a frame rate of 7fps. In the leftmost column labels of Figure 24, GT represents ground-truth, GB represents generated bounding box frames, and GF represents generated frames.
- Comparing the ground-truth and generated clips in Figure 24 reveals notable discrepancies. In the first example, a novel vehicle emerges (darker yellow bounding box), not present initially. This demonstrates BBox Generator's ability to create new bounding box instances, which Box2Video then uses to generate corresponding visual content

Pipeline	**# Cond. BBox**	**FVD↓**	**LPIPS↓**	**SSIM↑**	**PSNR↑**
BBox Generator + Box2Video	1-to-1	412.8	0.2967	0.5470	17.52
BBox Generator + Box2Video	3-to-1	373.1	0.3071	0.5407	17.37
BBox Generator + Box2Video	3-to-0	389.7	0.3085	0.5401	17.11
BBox Generator$^{\text{Traj}}$ + Box2Video	3-to-1$^{\text{Traj}}$	375.8	0.2973	0.5481	17.55
Teacher-forced Box2Video	All	**348.9**	**0.2926**	**0.5836**	**18.39**

Table 7: Summary of generation quality metrics on BDD100K dataset: evaluating the impact of bounding box conditioning types on generation quality.

Figure 22: A frame-by-frame visualization of Box2Video generation, conditioned on the ground-truth bbox frame sequence from the BDD dataset. The ground-truth bounding boxes are outlined in the plots.

Method	# Cond. BBox	maskIoU↑	maskP↑	maskR↑	maskIoU↑ (first+last)	maskP↑ (first+last)	maskR↑ (first+last)
BBox Generator	1-to-1	.587 ± .214	.747 ± .187	.712 ± .194	.954 ± .047	**.955** ± .047	**.999** ± .002
	3-to-1	**.647** ± .176	**.784** ± .150	**.783** ± .156	**.955** ± .043	**.955** ± .042	.997 ± .001
	3-to-0	.522 ± .204	.686 ± .204	.673 ± .201	.578 ± .214	.735 ± .222	.734 ± .201
	3-to-1$^{\text{Traj}}$	.606 ± .201	.773 ± .162	.722 ± .189	.798 ± .155	.886 ± .137	.894 ± .118

Table 8: Comparing trajectory alignment: comparing generated and ground-truth trajectory alignment under varying conditioning.

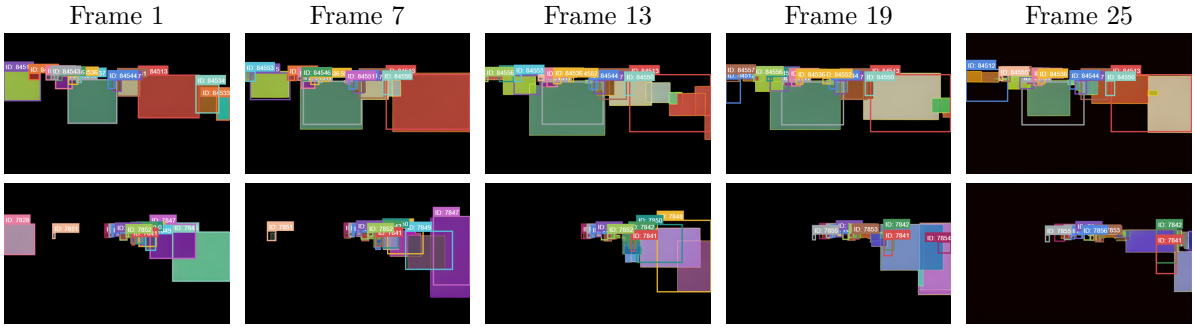


Figure 23: Bounding box trajectory generations are produced without conditioning on the last frame's bounding box locations. The provided demos display generated trajectories with ground-truth bounding box trajectories overlaid for visual comparison.

Figure 24: 2D bounding box frame generations and motion-controlled video generations on BDD100K test split where no final bounding box frame conditioning was provided.

F Trajeglish-Style Bounding Box Generator

To evaluate both the performance and simplicity of our bounding box generator model, we implemented a baseline method inspired by the Trajeglish model (Philion et al., 2023), which we refer to as the "Trajeglish-Style" model.

The original Trajeglish model employs a discrete sequence modeling approach to represent multi-agent road-user trajectories. Vehicle trajectories are modeled as a discrete sequence of actions, where each action corresponds to a token that describes the agent's relative displacement. The model utilizes a GPT-like encoder-decoder architecture, taking as input the agents' initial timestep information and map objects, and outputting a distribution over possible displacement actions. Trajeglish operates in bird's-eye view (BEV) and achieved state-of-the-art (SOTA) performance on the 2023 WOMD Sim Agents Benchmark.

To adapt this method to our problem, we made several modifications, which are detailed below. Our "Trajeglish-Style" model focuses on predicting 2D bounding boxes from the ego perspective (POV). Unlike the BEV representation, these bounding boxes can overlap in 3D space and are subject to resizing during rollout. Additionally, the boxes can dynamically enter and exit the frame at any point in the sequence.

For our baseline, we define the set of possible actions as the displacement of a bounding box corner. Each corner's displacement is represented by a 2D vector, where the vector's magnitude is discretized uniformly into 16 values, ranging from 0.0 to 0.1 of the image size in the respective dimension. The direction of the displacement is discretized into 24 values, covering the full 360-degree range. This joint discretization yields a vocabulary size of 384 (16 magnitudes * 24 directions). At each timestep, the movement of a bounding box is determined by selecting two actions: one for the displacement of the top-left corner and one for the bottom-right corner.

As input to the model's encoder, we provide the initial and final bounding box coordinates (1 or 3 frames), as well as the type of each agent (e.g., car, pedestrian, cyclist). In lieu of HDMap information (missing in most of the datasets used in this study) we use the initial natural image as contextual input to the encoder. The image is encoded using the same method as our diffusion-based bounding box generator (VAE image encoding combined with CLIP embeddings) to ensure fair comparison.

While the original Trajeglish model achieved SOTA performance in its intended task of predicting BEV agent trajectories, we do not claim that our "Trajeglish-Style" model reaches SOTA for the current task of ego POV 2D bounding box trajectory prediction. In fact, we consider this task to be significantly more challenging and less suited to the modeling approach of Trajeglish, which was not originally designed for the complexities predicting the motion of bounding boxes in the ego perspective, as outlined above. This increased difficulty accounts for the drop in performance, as seen in Table 1. Nevertheless, we believe that the "Trajeglish-Style" model serves as a valuable baseline for evaluating the performance of other models, including our own diffusion-based approach.

Despite the complexities inherent in this task, our diffusion-based BBox Generator, which operates directly on pixel images of bounding boxes, offers a compelling alternative due to its simplicity and strong performance.

The code for the "Trajeglish-Style" model implementation is available in the GitHub repository associated with this work.

G Key Insights and Future Directions

G.1 Failure Cases

Ctrl-V struggles to accurately encode and decode specific visual details, particularly road signs, building lettering, license-plate information and lane markings. This limitation stems from the constraints of the off-the-shelf VAE network used for image encoding and decoding. Figure 25 illustrates the VAE model's difficulty in accurately encoding and decoding the "MATTRESS FIRM" sign on the building. A potential direction for future research is exploring super-resolution techniques to recover fine-grained text details.

(a) Ground-Truth Image (b) VAE Reconstruction

Figure 25: Failure case: VAE's inability to reconstruct building signage.

Resolution degradation is a pervasive issue in video generation, primarily caused by error accumulation through temporal propagation. Figure 26 shows instances of degraded generation quality, where the last frame outputs suffer from a loss of realism. Future work can investigate multistage training paradigms, leveraging auxiliary networks to refine and enhance the visual quality of generated frames.

(a) Example of final frame generation (b) Example of final frame generation

Figure 26: Failure case: resolution degradation over time.

In general, generating realistic scenes during a turn is more challenging for the model. When the car is driving forward, distant background objects remain visible, providing consistent visual cues. However, during a turn, the model must infer and generate background elements that are not directly observed, making the task significantly harder. In addition, the ego video in the driving dataset primarily features forward motion. As a result, our model generates higher quality videos when the ego car is driving straight compared to when it is making a turn. During turns, the quality of the generated video tends to degrade. Figures 18 and 19

present examples of the performance of our model when the ego car is turning, where the backgrounds of the generated videos often lack detail.

Finally, our model has difficulty producing out-of-distribution situations involving complex physics, like car crashes. Due to the challenges of visualizing these cases effectively in the paper, we have provided an example of an out-of-distribution car crash generation on our website. We believe that training on specialized data may be necessary to improve the performance of our model in these scenarios.

G.2 Noteworthy Discoveries in our Box2Video Model

Figure 27: A frame-by-frame visualization of Box2Video generation, conditioned on the ground-truth bbox frame sequence from the BDD dataset, demonstrating the model's capability of handling incoming objects.

Our experiments have yielded findings suggesting that our Box2Video model not only understands the locations of the bounding boxes in scenes but also the encoded track IDs and the 3D bounding box orientation information in the rendered bounding box frames. Although these findings are not conclusive and require further investigation to confirm, they highlight potential directions for future projects. Here, we share some of these findings to guide further research and development:

1. During preprocessing, we assign a random color to each track ID and fill the bounding boxes with their corresponding track IDs' colors. When our Box2Video receives unreasonable generations, it still attempts to produce outputs that adhere to the given conditions, including the false bounding box IDs.

2. In our 3D bounding box plots, we encode the vehicle's orientation by marking the rear end with an "X". Occasionally, our BBox Generator incorrectly identifies the car's orientation, marking the front side with an "X". Therefore, during the video generation phase, we sometimes observe that the Box2Video model recognizes the wrong markings in the bounding box frames and generates a video of a car driving backward.

3. We also observe that Box2Video may recognize the encoded object information regarding its class in the bounding box frames. Specifically, when an object is not present in the initial frame but its bounding box label appears in the final frame, the model can decode its class information and generate a correct instance of that class in the clip. For example, observing Figure 27, we see the model successfully handles new objects entering the scene, accurately assigning the incoming car and pedestrian to their corresponding bounding boxes by leveraging encoded object IDs. However, we suspect that the object's class information could also be inferred from the bounding box shapes. More research is required.

4. Given the previous findings, we believe that plotting the bounding box outlines slightly thicker to make them more prominent could potentially improve our results.

G.3 Future Directions

For future work, we believe that it is worthwhile to develop a systematic approach to investigate the aforementioned observations. Additionally, it would be interesting to explore alternative methods for generating bounding boxes. Furthermore, it would be beneficial to create a systematic method to measure the likelihood of the bounding box frames, instead of directly computing their IoU, recall, and precision rate with ground truth.