# An Exploration Into AI-Generated Video Hallucination Detection

Neeloy Chakraborty

***Abstract*— Recent generative models like Sora have shown promise in producing high quality images and videos given text input, but they can also hallucinate inconsistent, undesirable, and irrelevant features. Existing video hallucination detectors, like Sora Detector, leverage another language model for image captioning, which itself can hallucinate. Recent works have shown promising results in exploiting extracted geometries from static images to detect hallucinations, but have not been applied to video sequences. In this work, we first present a short case study showcasing hallucinated predictions output by language models used for anomaly detection, motivating the need for structure-guided hallucination detectors. We then utilize state-of-the-art models to extract geometric cues from a custom-curated dataset of AI-generated and real videos, and directly apply a recent geometry-based hallucination detector on each static frame. Finally, we extend the static frame hallucination detector to consider the temporal consistency across consecutive frames by developing a Kalman filter that fuses predictions from different models. Project code is available at *https://github.com/TheNeeloy/temporal-projective-geometry*.**

## I. Motivation Through a Short Case Study

Recent advances in language modeling, generative AI, and foundation models have resulted in an explosion of creative models for image and video generation [1]. However, like any generative model, image and video generation models are susceptible to predicting undesired features at test time as a result of biases [2], low-quality prompts [3], and other reasons. The DALL·E system cards [4] provide plenty of examples of such characteristics with biased, stereotypical generations, deepfakes of public figures, and violent and racy imagery. Furthermore, these models have a tendency to hallucinate infeasible or unlikely generations, or in the case of AI-generated videos, implausible interactions among objects across frames.

Sora Detector, a new video hallucination detection method proposed by Chu *et al*. [5], relies on the generalizability of large language models to identify inconsistencies among a given prompt and extracted key frames from the generated video. In particular, Sora Detector first extracts key frames from a given video, described in Section II, and then, using a visual-language model (*e.g*., GPT-4), estimates knowledge graphs describing relations among objects across frames in the video. The combined knowledge graphs are input to the language model to predict a description for the video. Finally, the same model is used to identify inconsistencies among the generated and ground-truth prompt, and any static or dynamic hallucinations in the knowledge graphs. While the approach is intuitive, it is hindered by (1) ignoring non-key-frames, (2) problematic knowledge graph predictions, (3) hallucinations in generated text, and (4) high monetary deployment cost. To support these claims, we conduct a small case study comparing the predictions from Sora Detector on a generated and real video from the T2VHaluBench [5] and BDD100K [6] datasets.[1]

We first extract ten key frame clusters from both videos, and ask GPT-4 to predict static and dynamic knowledge graphs for each cluster and generate a description of the video. During this initialization phase, we find the framework hallucinates nonexistent objects in its descriptions and graphs, and misses key infeasible transformations of objects. For example, in the AI-generated video[2], GPT-4 mistakes a chair for a "robot dog" in the construction of a static knowledge graph, as shown in Fig. 4, and ignores sand morphing into a human in a dynamic knowledge graph, shown in Fig. 5. Hallucinations and misclassifications in knowledge graphs result in inaccurate generated video descriptions (*e.g*., "...the chair is tilted and breaks, while the individuals continue their activities with metal detectors and a robot dog nearby."), which inadvertently conflict with the actual prompt of the fake video. For the real video showcasing a dashcam view of a vehicle, Sora Detector predicts knowledge graphs accurately. However, when identifying conflicts between the ground-truth[3] and AI-generated descriptions, GPT-4 over-scores minor inconsistencies (*e.g*., "The

[1]Prompts and raw predictions from GPT-4 are in the code base.

[2]Original prompt was "Archaeologists discover a generic plastic chair in the desert, excavating and dusting it with great care."

[3]Manually set to "The video from a dashcam on a vehicle as it drives in a busy urban environment."

intensity of inconsistency is $8/10$, as there are notable differences in focus and interpretation between the two descriptions, particularly regarding activity level and perspective."), even though it states that the descriptions do not directly conflict (*e.g.*, "The AI's detailed scene descriptions do not directly conflict with the human description but highlight a difference in focus (2 points)."). Interestingly, the model consistently scores dynamic hallucinations as $0/10$ over all key frame clusters, acting conservatively for both fake and real videos. Finally, if using the cost estimate from Chu *et al*. [5], who state that each API call to GPT-4 is approximately $\$0.08$, the cost to score one video with ten key frames is $\$3.36$, with $42$ API calls.

Rather than relying on large visual language models that often hallucinate, we choose to explore extending the recent method from Sarkar *et al*. [7] for detecting AI-generated images with extracted geometric features, to video sequences. We hypothesize that this method will result in more consistent predictions, as it no longer hallucinates irrelevant features. Furthermore, we were particularly drawn to this problem-setting as our primary research is focused in anomaly detection.

## II. Approach

### A. Data Sources

Staying consistent with Sora Detector, we use T2VHaluBench as our dataset of AI-generated videos. The dataset consists of $59$ videos generated from state-of-the-art models, for a total of $14001$ fake testing frames. As the projective geometry hallucination detector we test in this work was pretrained on a dataset with real images from vehicle dashcams, we choose to use BDD100K to collect $59$ dashcam videos, for a total of $71320$ real frames at $256 \times 256$ resolution.

### B. Key Frame Extraction

To perform the case study on Sora Detector and evaluate the projective geometry-based classifier, we extract ten key frames from each video representing significant points. As such, we follow and implement a version of the density peak clustering algorithm presented by Chu *et al*.. The algorithm specifically extracts every $i^{\text{th}}$ frame from the video and computes a distance between each pair of frames. Chu *et al*. pose that frames which have a high density of features around them, and are far away from other clusters, are representative of key frames. Thus, using a kernel function, we first calculate the local density $\rho$ of distances per frame (*i.e.*, a high density for a frame means that there are many other frames close to it in distance). Then, for each frame, we compute $\delta$, the distance to the closest frame with higher $\rho$. Key frames are selected as the top $K$ frames with the highest values of $\gamma = \rho \cdot \delta$. Finally, Chu *et al*. also extract additional detail frames between each key frame to form clusters, by selecting frames with high distance between successive key frames.

### C. Projective Geometry-based Classification

Sarkar *et al*. empirically show that current image generation models fail to accurately depict real-world physical and geometric features, like shadows, perspective cues, and vanishing points. To do so, they train three separate binary classifiers to classify fake frames using these extracted geometries as input (completely ignoring the original images). As such, we aim to extend this method to video sequences. We note that, while Sora Detector is applied to key frames to minimize cost, this method can feasibly be applied to all frames in a sequence as it relies on local compute.

***Object-Shadow Predictions:*** We first extract detected object masks and their corresponding shadow masks for every frame in all videos using single-stage instance shadow detection (SSISv2 [8]). Since SSISv2 predicts a pair of segmentation masks for each object in an image, we combine all object masks and all shadow masks separately to produce two complete filtered images of predicted object and shadow pixels per frame. We find, however, that $23\%$ of real frames and $37\%$ of generated frames had no predicted masks.

***Perspective Field Estimates:*** Perspective fields [9] are represented by a predicted gravity field (*i.e.*, up-direction) and a latitude field (*i.e.*, angle between a light ray and the horizon). We use a pretrained perspective fields model to estimate both maps per image.

***Detecting Line Segments:*** The final geometric cue that Sarkar *et al*. rely on is detected line segments in images. Ideally, lines sampled from images that would be parallel in a 3D world, should intersect at a vanishing point in the image. Thus, the authors extract line segments from an image to classify abnormal generations. We similarly use DeepLSD [10] to detect line segments in every frame. We find real and fake images had an average of $183 \pm 51$ and $132 \pm 67$ lines.

***Classifying Hallucinated Static Frames:*** Sarkar *et al*. train a classification model for each input modality type – ResNet50 models for object-shadow and perspective field features, and PointNet for line segments. To curate varying difficulty test splits, we compute accuracy scores of a baseline ResNet50 model pretrained

to classify AI-generations from raw pixels. Easy videos are those that the baseline predicts frames correctly with at least $90\%$ accuracy, medium are those between $50-90\%$, and hard videos have less than $50\%$ accuracy. This split procedure results in $38|27$ easy, $15|14$ medium, and $6|18$ hard (real|fake) videos. We note that we test the generalizability of the classifiers, since none of the features have been seen during training.

### *D. Kalman-filter-based Temporal Fusion*

Directly applying the frame-level classification method from Sarkar *et al*. to each image in the video is a valid approach for detecting hallucinations. However, the methodology (1) does not fuse predictions from each classification model together for a final score and (2) ignores expected temporal consistencies across consecutive frames in the video. A recent anomaly detection method from Ji *et al*. [11] proposes treating scores from different classification models as noisy sensor measurements at each timestep. Then, a Kalman filter is designed to estimate the true de-noised classification score from each model, and the underlying fused anomaly score that averages the de-noised signals. We similarly implement a Kalman filter to fuse the models' scores together through time. More specifically, let $x_t = \left[\, x_t^{\text{os}} \; x_t^{\text{pf}} \; x_t^{\text{ls}} \; x_t^{\text{fus}} \,\right]^\intercal \in \mathbb{R}^4$ be the underlying state of the system at time $t$ that the Kalman filter estimates, where $x^{\text{os}}$, $x^{\text{pf}}$, and $x^{\text{ls}}$ are the anomaly scores of the object-shadow, perspective field, and line segment geometry-based classifiers, and $x^{\text{fus}}$ is the fused score. We initialize and start the Kalman filter at the first timestep that all models have a valid prediction:

$$x_1 = \begin{bmatrix} x_1^{\text{os}} & x_1^{\text{pf}} & x_1^{\text{ls}} & x_1^{\text{fus}} \end{bmatrix}^\intercal, \; x_1^{\text{fus}} = \frac{x_1^{\text{os}} + x_1^{\text{pf}} + x_1^{\text{ls}}}{3}$$

We initialize the estimate's covariance $P_1$ and the process noise covariance $Q$ matrices to a $4 \times 4$ matrix with $0.1$ along the diagonal. The dynamics matrix is:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \end{bmatrix}$$

Then, at each timestep, we first propagate our estimate of the state and its covariance:

$$\bar{x}_{t+1} = Ax_t + w_t, \; \bar{P}_{t+1} = AP_tA^\intercal + Q$$

where $w_t \sim \mathcal{N}(0, Q)$. Finally, for each model $s$ that has a valid prediction $z_{t+1}^s$ at the current timestep, we iteratively update our estimate of the state and error:

$$K = \bar{P}_{t+1}H^\intercal \left(H\bar{P}_{t+1}H^\intercal + 1\right)^{-1}$$

$$x_{t+1} = \bar{x}_{t+1} + K\left(z_{t+1}^s - H\bar{x}_{t+1}\right) + r_{t+1}$$

$$P_{t+1} = (I - KH)\,\bar{P}_{t+1}$$

where $K$ is the Kalman gain, $H \in \mathbb{R}^4$ is a row vector with a $1$ at the index representing $s$, and $r_{t+1} \sim \mathcal{N}(0, 1)$. Ideally, the Kalman filter will smooth out noisy predicted classifier scores and fuse them together to produce a clean final prediction. In practice, we compare results from three variants of the Kalman filter, which are run on classified labels, and raw and normalized probabilities from each classifier.

## III. Results

We now showcase qualitative and quantitative results for key frame extraction, geometric feature predictions, and effectiveness of classifiers on different difficulty test splits. First, we find the key frame extraction algorithm effectively identifies novel views and significant moments in real and fake videos, with an example set shown in Fig. 6. However, there exist cases where detail frames for any given cluster are very similar to one another, even though we use normalized cross correlation as the image distance metric. Poor detailed clusters may negatively impact methods like Sora Detector, which rely on informative detailed frames for dynamic hallucination detection, but does not affect feature-based methods like Sarkar *et al*. or our extension.

Object and shadow masks, line segments, and perspective fields are predicted for each image using their respective pretrained models. Example predictions for a frame in a generated video are shown in Fig. 7. Notably, we find SSISv2 misses certain shadows that could be considered to be mapped with an object when there are multiple possible choices for mappings. Furthermore, DeepLSD does pick up on several structural lines in the image on the bed's headboard, but it also predicts lines for shadows and patterns on the comforter.

We plot the receiver operating characteristic curve (ROC) for the baseline and each of the geometry-based classifiers for all frames in Fig. 1 and for key frames in Fig. 2. Like Sarkar *et al*., we see that the baseline model outperforms the geometry-based classifiers on its easy split. However, the image-based classifier showcases better generalizability. Interestingly, while we see the same trend in drop of AUROC across both the baseline and other classifiers, the geometry-based classifiers significantly outperform the baseline on the hard split. Overall, we find that the perspective fields anomaly detector outperforms the other geometric classifiers on

the complete and easy splits, but the object-shadow classifier is better on harder splits.

Finally, we analyze the effectiveness of our Kalman filter extension in Fig. 3. We can see that this extension outperforms the static methods on the easy split, and performs as well as the perspective fields-based model on the complete split. However, due to the drop in accuracy of the perspective geometry-based classifier on medium and hard difficulty splits, the Kalman filtered raw probabilities are outshined by the normalized probability ablation. Unfortunately, none of the Kalman filter approaches reach the same AUROC as the baseline on all frames on the complete split.

## IV. Implementation

The majority of this project was implemented in Python. The primary libraries used include Numpy (data processing and matrix math), scikit-learn (metric computations), Matplotlib (graphing), PyTorch (model inference), SciPy (filtering and sampling noise), and OpenCV/PIL (processing images). As stated in Section II, we rely on SSISv2, perspective fields, and DeepLSD pretrained models to extract geometric features, and run the classifiers from Sarkar *et al*. at test time. We perform data curation ourselves, implement the key frame extraction module based on the described algorithm from Sora Detector, and design and implement the Kalman filter from scratch. Additionally, as Chu *et al*. and Sarkar *et al*. do not provide hyperparameters for the keyframe and geometry data extraction models, we tuned parameters empirically and wrote custom inference scripts to extract the required features. Specifically, we chose normalized cross correlation as the distance metric, a gaussian kernel with cutoff distance set to $0.5$ for local density estimation, and select the top ten key frames during key frame selection. The confidence threshold for SSISv2 is set to $0.1$ and we use the latest perspective fields model pretrained on both indoor and outdoor scenes. DeepLSD is set to filter excess lines with a gradient threshold of $3$. We evaluate the geometry-based and baseline classifiers from Sarkar *et al*. trained on a combined dataset of indoor and outdoor scenes, and our Kalman filter pre-smooths scores with a $2^{\text{nd}}$ order low-pass Butterworth filter. Other parameters for the Kalman filter are discussed in Section II-D. To perform the case study in Section I, we rely on our access to Perplexity to generate GPT-4o predictions. Perplexity's API is restricted to $\$5$ worth of predictions, and does not offer access to GPT-4o, so we had to (painstakingly) manually query the model several times through the web interface to generate results. All other results are collected on a local machine with a Quadro P2000.

## V. Innovation

We tackle the challenging problem of video hallucination detection – a very recent area of research. Our work is innovative in that we are the first to extend a static frame hallucination detector to videos. Along the way, as discussed in Section IV, we made design decisions to account for vague descriptions in original works. Additionally, unlike Ji *et al*., our Kalman filter can account for missing classifier predictions at any timestep. As such, we believe that our work deserves full points for innovation and challenge.

## References


[1] OpenAI, "Video generation models as world simulators," *OpenAI blog*, 2024. [Online]. Available: https://openai.com/index/video-generation-models-as-world-simulators/

[2] V. Rawte, A. Sheth, and A. Das, "A Survey of Hallucination in Large Foundation Models," *arXiv:2309.05922*, 2023.

[3] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, *et al.*, "Evaluating Large Language Models Trained on Code," *arXiv:2107.03374*, 2021.

[4] OpenAI, "DALL·E 3 System Card," *OpenAI blog*, 2023. [Online]. Available: https://cdn.openai.com/papers/DALL_E_3_System_Card.pdf

[5] Z. Chu, L. Zhang, Y. Sun, S. Xue, Z. Wang, Z. Qin, and K. Ren, "Sora Detector: A Unified Hallucination Detection for Large Text-to-Video Models," *arXiv:2405.04180*, 2024.

[6] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2633–2642.

[7] A. Sarkar, H. Mai, A. Mahapatra, S. Lazebnik, D. Forsyth, and A. Bhattad, "Shadows Don't Lie and Lines Can't Bend! Generative Models Don't know Projective Geometry…for Now," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 28 140–28 149.

[8] T. Wang, X. Hu, P.-A. Heng, and C.-W. Fu, "Instance Shadow Detection With a Single-Stage Detector," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 3, pp. 3259–3273, 2023.

[9] L. Jin, J. Zhang, Y. Hold-Geoffroy, O. Wang, K. Blackburn-Matzen, M. Sticha, and D. F. Fouhey, "Perspective Fields for Single Image Camera Calibration," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 17 307–17 316.

[10] R. Pautrat, D. Barath, V. Larsson, M. R. Oswald, and M. Pollefeys, "DeepLSD: Line Segment Detection and Refinement with Deep Image Gradients," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 17 327–17 336.

[11] T. Ji, N. Chakraborty, A. Schreiber, and K. Driggs-Campbell, "An expert ensemble for detecting anomalous scenes, interactions, and behaviors in autonomous driving," *The International Journal of Robotics Research*, 2024.

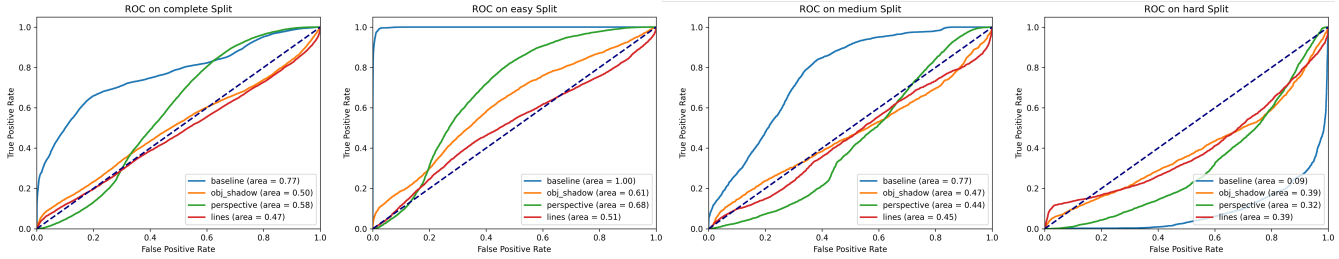


Fig. 1. **ROC graphs for static frame classifiers by test split difficulty on all frames.**

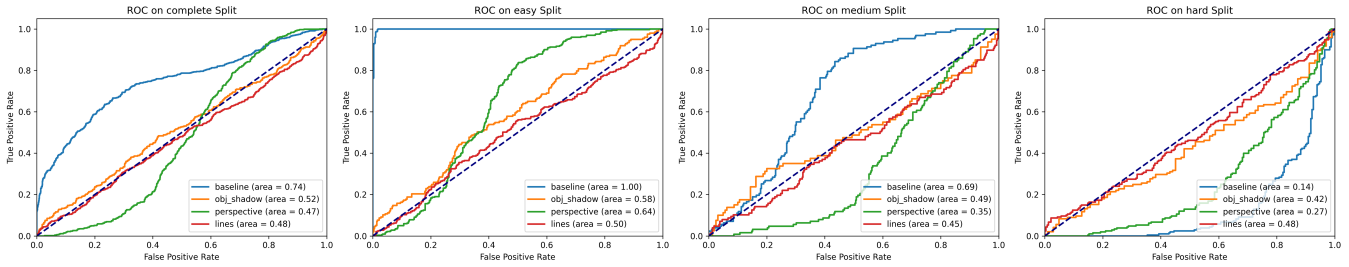


Fig. 2. **ROC graphs for static frame classifiers by test split difficulty on key frames.**

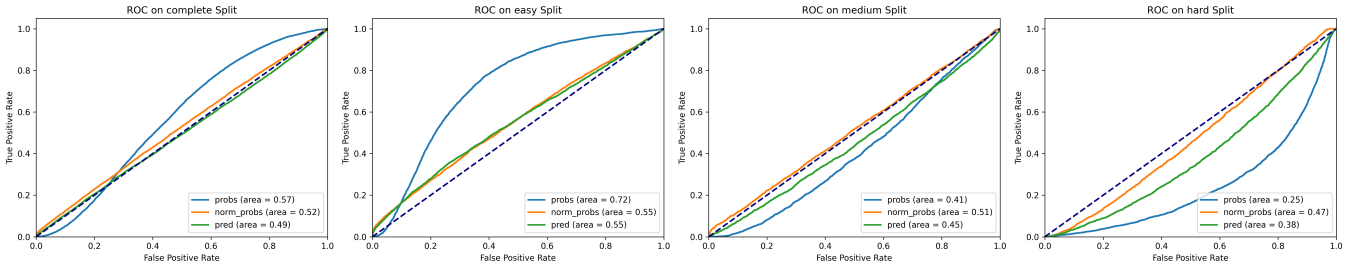


Fig. 3. **ROC graphs for Kalman filtered classifiers by test split difficulty on all frames.**

Fig. 4. **A hallucinated static knowledge graph:** {{“objects”: [“person1”, “person2”, “person3”, “robot_dog”], “relations”:[[“person1”, “near”, “robot_dog”], [“person2”, “near”, “robot_dog”], [“person3”, “near”, “robot_dog”]]}}

Fig. 5. **A hallucinated dynamic knowledge graph:** {{“changes”: { “chair1”: “unchanged”, “person1”: “unchanged”, “person2”: “unchanged”, “person3”: “unchanged”, “bag1”: “unchanged” }}}

Fig. 6. **Example keyframes from a real video.**


Fig. 7. **Example model predictions on a generated frame.**