RTD: A Video Dataset for Runway Tracking Applications

João Pedro Klock Ferreira*, João Paulo Lara Pinto*, Júlia Santos Moura*, Gabriel Lott*, Cristiano Leite de Castro*
*Graduate Program of Electrical Engineering, Universidade Federal de Minas Gerais

Abstract—**Recent advances in the aerial industry with the advent of computer vision have led to the need for large image datasets in this field. Specifically, there is a lack of video datasets for the task of tracking the runway during a flight landing process. In this paper, we propose a new dataset with 20 videos, where we propose 10 manually annotated videos of real planes landing, and 10 synthetic videos from the Video-LARD dataset [1] with new weather conditions and lightning variations, generated through UniSVST [2], a diffusion image-to-video model. We also perform careful experiments using a pre-trained version of the LoRAT tracking model [3] to verify the quality of our dataset, assessing its strengths and weaknesses. The results show that the model obtained better tracking performance when closer to the runway for real videos, and for synthetic videos, blurry climates such as fog or dark nights led to a worse performance. Overall, the dataset incorporates a variety of difficult tracking conditions, but synthetic video generation can still be improved. The dataset and any relevant code are publicly available at https://github.com/jpklock2/RTD**

Index Terms—**object tracking, video dataset, runway, diffusion model, plane landing**

I. Introduction

THE process of landing a plane is one of the most critical aspects of a flight journey [4], [5]. Although current systems already allow for completely automatic landings [6], it requires high attention from the pilot, in addition to the elevated maintenance costs. One workaround is the usage of computer vision solutions, that have been sought because of their lower cost and capacity of providing valuable sensing information for landing guidance. As such, a need appeared in the literature for image datasets related to plane landing.

Since it is not easy to obtain real data from landing planes due to the lack of recorded flights and the varying weather and illumination conditions, gathering diverse data to create an annotated dataset for this specific task is often not feasible. Among the few different datasets proposed in the literature, most deal with the problem by obtaining images from simulators rather than real life.

In this work, we first categorize the different datasets in the literature for runway identification, segmentation, and autonomous landing. We focus on publicly available datasets with images from the aircraft perspective, simulating a camera either coupled to the plane or inside the cockpit. We gather information from datasets that use both images from simulators and images obtained from real-life cameras.

One of the first proposed datasets is BARS [4], with images acquired from the X-Plane simulator. The authors collected 10,256 images with a total of 30,201 instances of semantic masks from 3 different categories: runway, aiming marking, and threshold marking, all annotated using the LabelMe toolbox [7]. It comprises around 40 airports from 15 countries, with images collected from 500 meters of altitude until landing on the runway.

In an attempt to collect images closer to real-life cases, the LARD dataset [6] used Google Earth Studio, where it is possible to create images with custom light conditions. The images were collected by creating a virtual geometric cone around the runway and varying the distance between the plane and the runway, along with several angles, creating a variety of perspectives and altitudes. This resulted in a total of 16,748 images from 111 different runways and 55 different airports. In addition to the simulated images, they also collected 1,811 images from 38 runways and 36 airports using YouTube videos of plane landings with different illumination and weather conditions, resulting in a combined dataset of simulated and real-world data.

In an improvement of the BARS dataset, in the FS2020 dataset [8] the images were acquired from Microsoft Flight Simulator 2020, which is able to vary both the illumination and the weather conditions. The authors gathered images from 60 runways and 91 tracks, totalizing 5,587 images in total, and used the LabelMe tool to annotate the dataset. Like the BARS dataset, they extracted semantic masks of 3 categories: runway area, aiming point marking, and threshold marking. Additionally, they also extracted line labels from the runway for 6 categories: left edge, right edge, center line, aiming point front, threshold rear, and PAPI lights.

In another improved version of the BARS dataset, the RLD dataset [5] also acquired images from the X-Plane simulator, but combined with a few images from YouTube videos. The authors collected a total of 12,239 images from 30 airports and also used the LabelMe tool to annotate the dataset, creating semantic masks for each runway. Their main focus was to create a dataset with different weather conditions and geographical environments.

The AeroRunway dataset [9], similar to BARS, also uses images from the X-Plane simulator, and is composed of 3,880 images from 28 airports with different weather conditions. The key difference is that the images from this dataset were generated with much higher altitude, reaching up to 3,000 meters. The main problem with this dataset is the lack of annotations of the runway.

The dataset proposed in [1] (here referenced as Video-LARD) is a variation of LARD, where the authors focus on the problem of tracking the runway. Using the LARD source code, they propose a variation of LARD in which they regenerate the dataset in the form of videos of the planes landing, as sequential frames approaching the runways, totalizing 14,003 images from 51 airports and 104 runways. The resulting labels marked as the four runway corner points are the same as the LARD dataset for each individual frame.

The GARD dataset proposed in [10] is another variation of the LARD dataset where images were modified using diffusion models to generate varied lightning and weather conditions with realistic backgrounds. By including different conditions such as time of day, occlusion, and other variations of several base images, a total of 45,486 images were generated, where each image has metadata containing the runway coordinates and the applied transforms and conditions.

Despite these datasets having their specific advantages, only the Video-LARD dataset is focused on video tracking; however, this dataset has an inherent problem of data diversity since it is only comprised of synthetic steady videos, where the plane follows a single trajectory to the runway. In addition, although the Video-LARD dataset contains varying light conditions from different acquisition times, it also lacks adverse weather conditions, which could cause poor visibility and provide a challenge for tracking models. This could lead the tracking results to be biased towards videos with a good visibility condition, invalidating them in real-case scenarios.

To deal with this lack of diverse video datasets for runway tracking in the literature, we propose the Runway Tracking Dataset (RTD) composed of real-life footage with per frame annotation, along with synthetic videos with diverse illumination and weather conditions, also annotated per frame. We focus on commercial runways to acquire the data, proposing 20 videos that target standard landing cases for a generalist dataset. We use the LoRAT tracking model [3] trained with the Video-LARD steady synthetic videos, to test our dataset and check for consistency. In summary, our main contributions are as follows:

- 10 YouTube real landing videos with 3 minutes of annotation (runway bounding boxes), with different camera perspectives and weather conditions.
- 10 synthetic videos generated from videos of the Video-LARD dataset with climate and illumination effects, and a steady trajectory towards the runway. The GARD dataset images were taken as the source of image stylization.
- Benchmark and results on our real and synthetic landing videos showing the generalization ability of a state-of-the-art tracking model that was trained only with simulated clear weather data.

A summary of the datasets in this study is presented in Table I, where each dataset is compared and briefly described. Even though our dataset contains fewer runways than other datasets in the literature, the increase in the number of images, along with the diverse climate conditions, should provide enough information to generalize tracking algorithms.

II. Methodology

The methodology to create our dataset can be divided into two parts. The first explains how we annotated the YouTube real landing videos. The second explains how we use diffusion models to give diversity (e.g. weather conditions) to the simulated videos. The complete pipeline to generate our dataset can be seen in Figure 1.

A. Real Videos

For our real videos, we chose 10 videos from the LARD test dataset. In LARD, the authors gathered several videos from different YouTube channels, but labeled only a few frames per video. This is suitable for object detection purposes, but not for object tracking. Among the 10 videos from LARD, 6 of them correspond to the Nominal cases and 4 to the Edge cases. Unlike LARD, we annotated the landing segment of those videos. We sampled videos with different weather, visibility, and illumination conditions, from different runways, from different zones (Urban and Rural), and from different perspectives with the cockpit visible or not.

To label our videos, unlike previous works that mainly used the LabelMe toolbox, a tool suited for single images, we chose another tool that is more suitable for videos: the Vidat tool [11]. This tool allows the user to annotate objects with bounding boxes, regions (or semantic masks), and skeletons in videos, but the main advantage of the tool is the possibility to annotate only a few keyframes and linearly interpolate the labels between frames. Although we may lose label accuracy this way, it allows for a faster and viable annotation of several videos.

Since our goal is to provide a dataset for object tracking, we need only the bounding box around the runways, so this is our only annotated feature from the real videos. To perform a viable manual annotation of thousands of frames, we sampled 1 keyframe for every 2 seconds of video when close to the runway and 1 keyframe for every 3 seconds when distant from the runway. This is motivated by the fact that the farther the runway is, the fewer visual changes per frame are perceived. An overview of the characteristics of the labeled videos can be seen in Table II, and some examples of images with the runway bounding box are presented in Figure 2. Each video was annotated with 30 frames per second and a resolution of 1920x1080 pixels.

B. Synthetic Videos

The possibility of generating synthetic videos is crucial to propose a runway tracking dataset because despite the fidelity of real-world videos, it is unfeasible to create a diverse variety of situational real cases. To address that, we also chose to generate synthetic videos.

In our previous work [1], we had already proposed the Video-LARD dataset with a total of 104 simulated videos. But despite it being a large dataset, their videos are all composed of ideal cases, where the plane lands in a straight and steady line, which is not ideal. In addition, although those videos have variations in terms of illumination and day time, they do

TABLE I
RUNWAY DATASETS COMPARISON

Dataset	Images Origin	Target Application	Images	Airports	Runways	Available Labels
BARS [4]	Simulator (X-Plane)	Runway Segmentation	10,256	40	-	semantic masks of runway, aiming marking, and threshold marking
LARD [6]	Simulator (Google Earth Studio) and Real (YouTube videos)	Runway Detection	18,559	91	149	runway corner coordinates, distance to runway, plane angles related to the runway
FS2020 [8]	Simulator (Microsoft Flight Simulator 2020)	Runway Segmentation	5,587	-	60	semantic masks of runway, aiming marking, and threshold marking, line labels of left edge, right edge, center line, aiming point front, threshold rear, and PAPI lights
RLD [5]	Simulator (X-Plane) and Real (YouTube videos)	Runway Segmentation	12,239	30	-	semantic masks of runway
AeroRunway [9]	Simulator (X-Plane)	Runway Detection	3,880	28	-	-
Video LARD [1]	Simulator (Google Earth Studio)	Runway Tracking	14,003	51	104	runway corner coordinates, distance to runway, plane angles related to the runway, per frame annotation
GARD [10]	Simulator (Google Earth Studio + diffusion models)	Runway Segmentation	45,486	-	-	runway corner coordinates, image transformation and scene generation records
RTD (Ours)	Simulator (Google Earth Studio + diffusion models) and Real (YouTube videos)	Runway Tracking	29,480	18	16	runway corner coordinates, distance to runway, plane angles related to the runway, per frame annotation

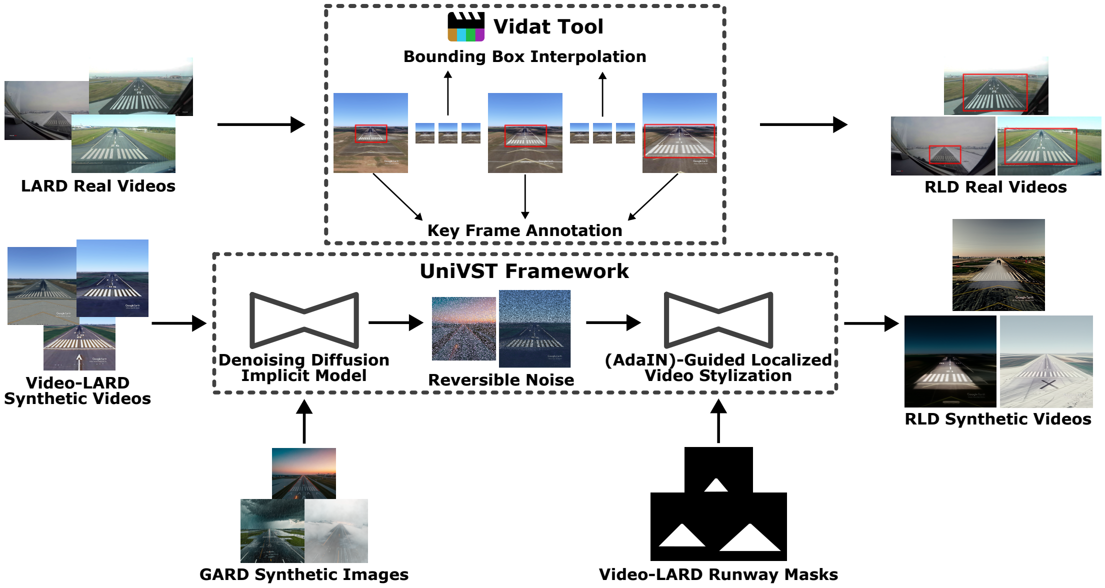

Fig. 1. Pipeline used to generate RLD real videos (top) and synthetic videos (bottom).

not present variations of weather conditions, causing models to be biased towards perfect landings.

In the GARD dataset [10] the author proposes using diffusion models to perform image-to-image style transfer to generate a variety of different conditions to runway images, such as adverse weather or occlusion. Combined with the LARD dataset, which has the potential to create a large number of scenarios, this allowed the author to create a vast and complete dataset with different difficulties for an object detection model.

To extend this solution to a video, one possible solution would be to generate a new synthetic image for every frame while using the same style image as the baseline. The problem with this approach is that different frames would often look different, making it difficult to keep the overall consistency of the videos. Instead, we decided to search for an image-to-video diffusion model, which is a class of models that already deals with the consistency problem.

TABLE II
LABELED YOUTUBE VIDEOS

Video URL	Frames	Weather	Time	Zone
LTAI_36C	3,600	Clear	Day	Urban
ESSA_26	3,600	Clear	Day	Rural
KMIA_9	3,600	Clear	Dawn	Urban
LTAI_36C	3,600	Clear	Day	Urban
EDDF_7R	3,600	Fog	Day	Semi-Urban
EHAM_18R	3,600	Clear	Day	Rural
UGTB_31L	1,800	Fog	Dawn	Urban
LQSA_12	1,800	Snow	Day	Urban
ZBAA_36R	1,800	Fog	Dawn	Urban
EBBR_25L	1,800	Rain	Day	Rural

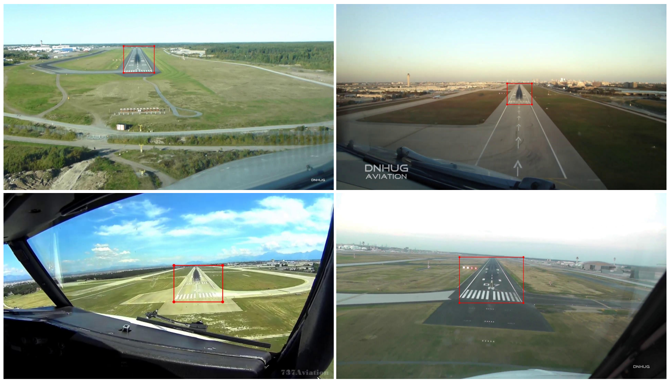

Fig. 2. Examples of frames from real videos with runway bounding box.

When searching for the ideal model [12], we wanted one that required the minimum amount of work from the user to generate the videos, allowing for possible customizations and dataset additions in the future. Although some state-of-the-art works provide good results [13], using local and global features to ensure a consistent stylized video, one challenge of our dataset is that in the first frames the runway is very small compared to later frames, causing the object to change a lot during the video length, which is a challenge for several image-to-video models.

To address these problems, we decided to use UniVST [2], a framework that does not require training and focuses on localized video style transfer, which is suitable for our need to focus mainly on the runway. In UniVST, instead of providing only an image and a video, you also need to provide a semantic mask of the object that the model must focus on. The model works in 3 steps, where first features from both the image and video are extracted, by passing them through a Denoising Diffusion Implicit Model (DDIM) [14], which transforms the images (or each frame for the video) into a reversible noise.

In the second step they perform a Mask Propagation step where the initially provided object mask is propagated to the subsequent frames using the previously extracted features. Given the particularity of our constantly size-changing object, preliminary results were not good, so we decided to skip this step by providing the semantic masks for the runway, which are easily obtained from the LARD features present in the Video-LARD dataset.

In the third step, all features along with the frame masks are fed to their proposed Adaptative Instance Network (AdaIN)-Guided Localized Video Stylization framework, which leverages the features while performing the network inference. Finally, when reconstructing the frames with their network features, they also employ a Sliding Window Consistent Smoothing step, which ensures the spatiotemporal consistency of the video.

We generated 10 videos using the UniVST framework, 5 from the training set, and 5 from the test set of the Video-LARD dataset. We used different images from the GARD dataset as baseline styles, targeting different weather and illumination conditions. The characteristics of the 10 videos are shown in Tables V and VI, presented in the Results section.

III. RESULTS

A. Experimental Setup

In this section, we perform some benchmark tests to ensure that our dataset is viable for other applications. First, we test our real footage dataset by inferring with an already pretrained ViT-based tracking model, named LoRAT [3]. Then we evaluate the consistency of our synthetic dataset by performing a qualitative analysis, and finally we also perform an inference with the same pretrained tracking model. All experiments were carried out with a single RTX 5090.

B. Real Data Labeling Evaluation

To test our proposed video dataset, we perform inference using the LoRAT model trained with the Video-LARD dataset [1]. This model was trained using only synthetic steady sequences that go straight to the runway; therefore, the results in a real-world dataset may vary. In addition to that, as explained in Section II-A, we used different sampling rates when labeling, so our labels are expected to be more accurate closer to the runway, and this algorithm works by receiving the first bounding box and using it to track in subsequent frames, so the accuracy of the first frame is of great importance.

Given these considerations, since our nominal sequences have around 2 minutes each we decided to perform 3 experiments where we start with the first frame around 40, 80 and 120 seconds before the plane lands, until it hits the runway. For the edge cases, the videos were smaller, so we selected intervals around 20, 40 and 60 seconds. The idea behind this experiment is that the runway will be more visible when the plane is closer, so the model is expected to perform better in the first experiment. This will help to reduce the bias of only testing the algorithm from a distant position, which could lead to bad results due to the methodology, not the accuracy of the dataset.

To evaluate the results of the model, we use the One-Pass Evaluation (OPE) metrics [15]:

- Success Score (SUC): indicates the average IoU between the ground truth and predicted bounding boxes along the video frames.

- Precision (P): indicates the average distance between the center of the ground truth and the predicted bounding boxes, with a threshold of 20 pixels.
- Normalized Precision (P_{Norm}): is the same as precision but normalized by the size of the ground-truth bounding box.
- Success Rate at IoU >= 0.5 ($SR_{0.5}$): indicates the percentage of frames where IoU is greater than 0.5.
- Success Rate at IoU >= 0.75 ($SR_{0.75}$): indicates the percentage of frames where IoU is greater than 0.75.

The results of this experiment can be seen in Tables III and IV, where we show the mean results for all the runways in each experiment, and also in Figures 3 and 4, where we show the IoU along the duration of the video.

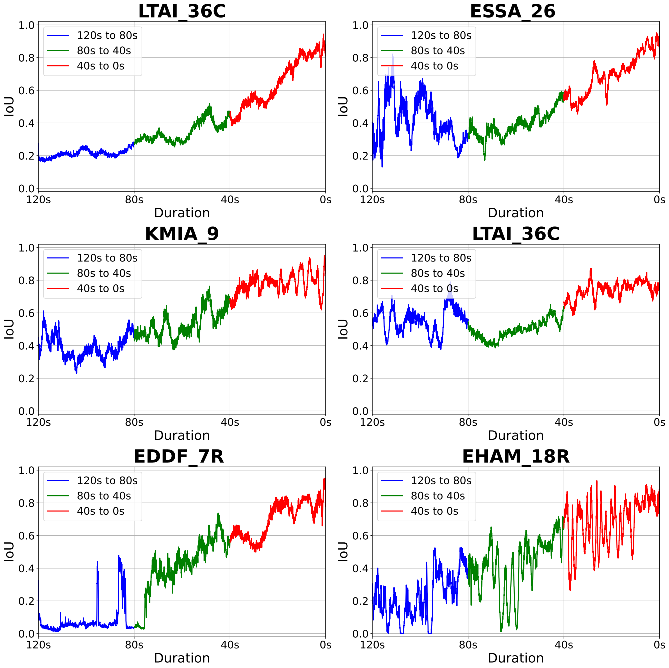

Fig. 3. IoU of real nominal videos along the duration of the video.

TABLE III
REAL VIDEOS TRACKING RESULTS FOR NOMINAL CASES

Seconds to land	SUC	P	P_{Norm}	$SR_{0.5}$	$SR_{0.75}$
40 until 0	0.70	0.97	0.95	0.95	0.45
80 until 0	0.57	0.97	0.92	0.62	0.22
120 until 0	0.49	0.98	0.84	0.48	0.15

TABLE IV
REAL VIDEOS TRACKING RESULTS FOR EDGE CASES

Seconds to land	SUC	P	P_{Norm}	$SR_{0.5}$	$SR_{0.75}$
20 until 0	0.58	0.90	0.85	0.72	0.10
40 until 0	0.37	0.76	0.56	0.35	0.04
60 until 0	0.35	0.82	0.49	0.31	0.03

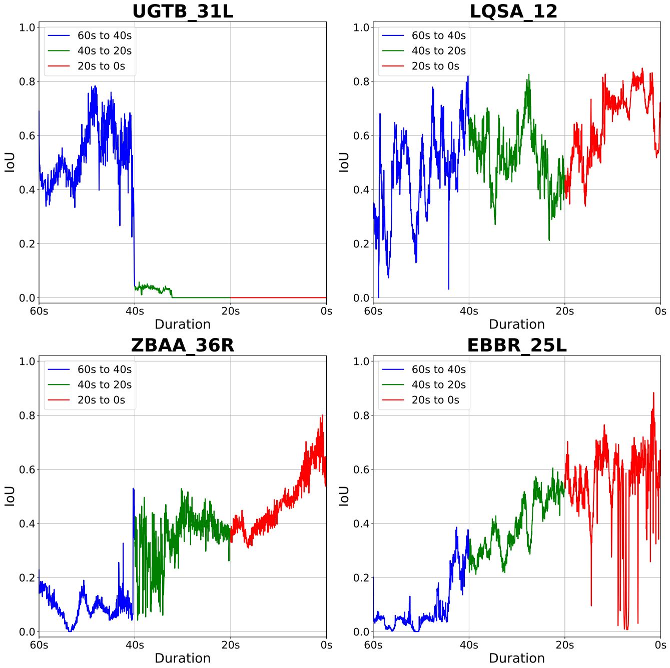

Fig. 4. IoU of real edge videos along the duration of the video.

For the nominal cases, the SUC metric indicates that the average IoU is higher when the plane is closer to the runway, and both the precision metrics indicate that the center of the ground truth and predicted bounding boxes are closer when the plane is more distant. Both metrics can be explained because when the plane is distant from the runway, the bounding boxes are small, so minor changes to their positions will have a high impact in their overlap, causing the IoU metric to be lower, but their centers will be closer, and as the plane approaches the runway and the bounding boxes become larger, the centers will be further apart, causing the precision to decrease.

This can also be verified in Figure III, where we can see that the IoU is improving as the plane approaches the runway. We can also see in the plots that all of them get worse when the plane is very close to the runway. In some of the last frames the plane had already landed and since the Video-LARD dataset used to train the model only had images until almost reaching the runway, the high amount of visibility and extensive size of the runway (occupying more than 50% of the images) probably hinders the model performance for these frames.

Regarding the edge cases, results follow a similar pattern, but they are generally worse, indicating that the tracking model performs worse with the climate occlusion artifacts. Specifically for the UGTB_31L runway, in the 60 seconds experiment (as seen in Fig. 4) the model performs poorly and the model is not able to keep tracking the runway with the provided bounding boxes, but results from the 20 seconds experiment from Table IV indicate that as the plane is closer to the runway, the tracking significantly improves.

Overall, since the model was trained using synthetic steady sequences, the performance is acceptable for the real images dataset. Additionally, to the best of our knowledge, this paper is the first to evaluate Vision-based Tracking models on real landing videos from the LARD test subset.

C. Synthetic Data Quality

Before evaluating the synthetic data in the tracking task, we first perform a qualitative evaluation of the generated videos, which are shown in Figures 5 and 6.

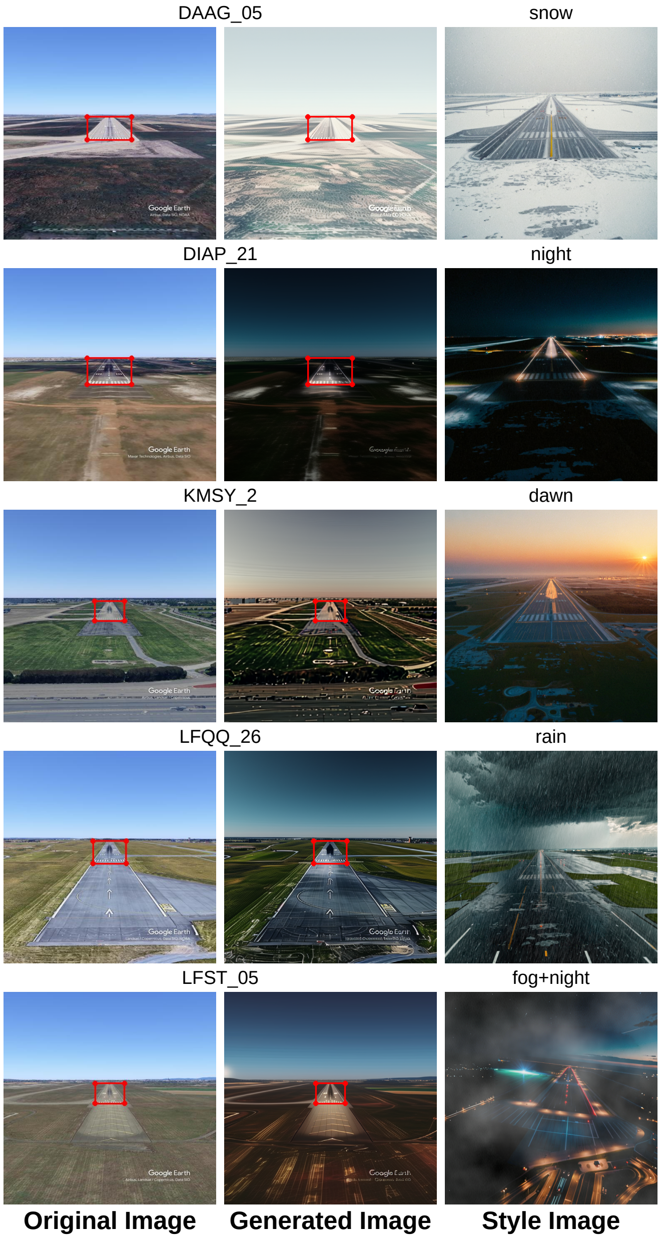

Fig. 5. Examples of frames from the generated videos with the original reference and the style image used.

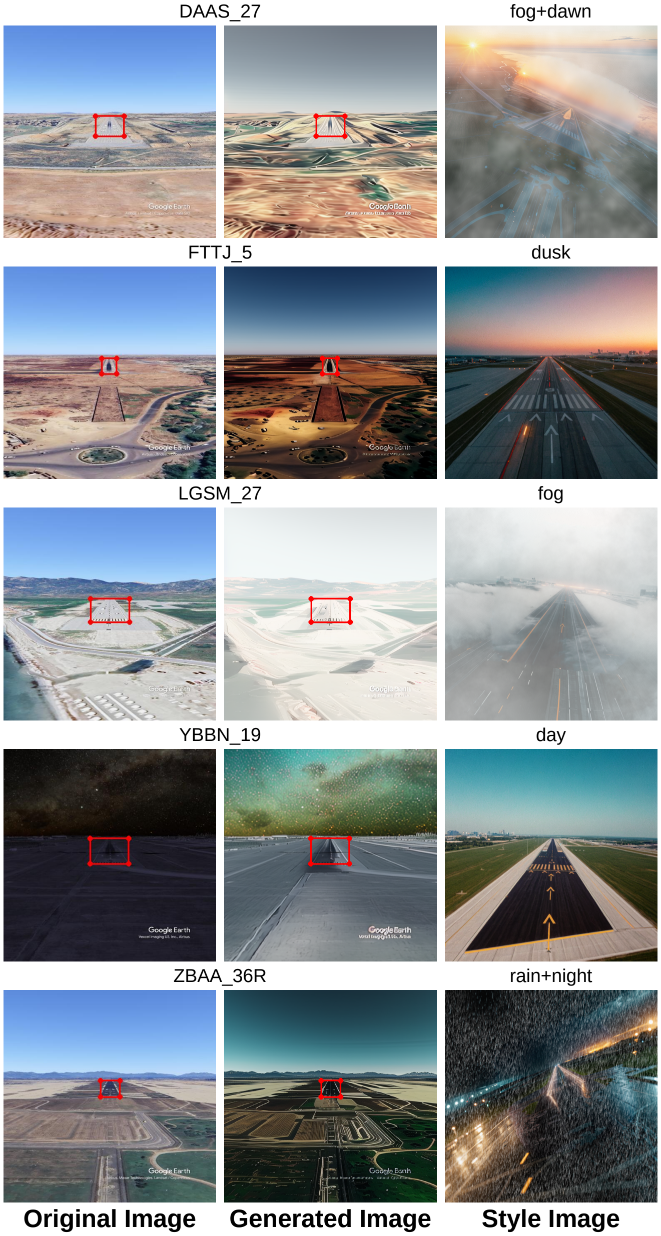

Fig. 6. Examples of frames from the generated videos with the original reference and the style image used.

The figures show that even though the UniVST framework is able to change some aspects of the video style, such as color and minor details, the runway is mostly preserved. Some larger details from the scene, such as snow present in the soil, are not well propagated, turning the image brighter, but not necessarily appearing in the generated image. And it was also noticed that artifacts such as the sun or lights in the runway or city are not propagated among the generated features.

Weather details such as falling rain, fog, or falling snow are also not propagated, which means that they are probably treated as noise by the network, rather than a desired artifact. To deal with this, one possibility would be to use handcraft methods, as in GARD [10], to apply changes to each frame, but this could cause disconnectivity of the weather artifacts between frames, causing the video to be chaotic and noisier. Another possibility would be to use another diffusion model focused on climate changes to apply these effects, such as WeatherWeaver [16], but this would increase the overall complexity of the problem.

Finally, in the video YBBN_19 (see Fig. 6) the night sky originally had stars, but when the model was supposed to change the image to a bright day sky, the stars remained, so it ended up looking like a clear night, which showcases the inability of the model to change large details from the images. In general, since the LARD methodology to collect images [6] allows for changes in time of day, if it is only desirable to change the time and color of the videos, using their methodology would still be preferable. But in the case of real videos, or if other climate changes are desirable, one could still benefit from using a diffusion model.

D. Synthetic Data Tracking Evaluation

To evaluate the quality of the videos generated in the tracking test, we also use the Video-LARD tracking model [1]. For that, we evaluate the tracking model on the original videos, which are supposed to achieve better results since they are from the original dataset used to train and validate the model, and then we obtain the same metrics for the generated videos.

Since we generated videos based on both the training and test sets of the Video-LARD model, we divide the results into two, where we first show the tracking metrics for the videos used in the training set in Table V, and for the test set in Table VI. In the tables, there is a brief description of the original and generated style for each pair of videos, the number of frames denoted by **F**, and the tracking metrics. In the end, we also show the mean results for the original and generated videos. And in Figure 7 we show the tracking IoU for each frame of both Real and Generated videos.

TABLE V
Generated Videos Tracking Results for Train Set

Video ID	Style	F	SUC	P	P_{Norm}	$SR_{0.5}$	$SR_{0.75}$
DAAG_05	day	462	0.93	0.98	1.00	1.00	1.00
Generated	snow		0.78	1.00	0.98	1.00	0.68
DIAP_21	day	473	0.93	0.99	1.00	1.00	1.00
Generated	night		0.49	1.00	0.60	0.48	0.23
KMSY_2	day	462	0.92	0.99	1.00	1.00	1.00
Generated	dawn		0.62	1.00	0.82	0.81	0.21
LFQQ_26	day	459	0.92	0.99	1.00	1.00	1.00
Generated	rain		0.66	1.00	0.71	0.83	0.34
LFST_05	day	464	0.93	0.99	1.00	1.00	1.00
Generated	fog+night		0.74	1.00	1.00	1.00	0.46
Mean Ori.	-	-	0.93	0.99	1.00	1.00	1.00
Mean Gen.	-	-	0.66	1.00	0.82	0.82	0.38

The training results show a decrease of 27% in the SUC score, which means that the model had more trouble finding the correct bounding boxes properly, although their centers are still properly tracked, as shown in the precision metrics. These indicate that in most cases the bounding box is properly placed, but it has the wrong size, being either smaller or larger.

Looking at individual results, the worst values were reached for the SUC and $SR_{0.75}$ metrics when converting from day to night and dawn. Looking at the images in Figure 5, we can see that for the DIAP_21 result the runway almost blends in with the environment, which can explain why the model was unable to track the runway properly. This is also supported by looking at the poor performance in the first frames of the IoU plot in Figure 7.

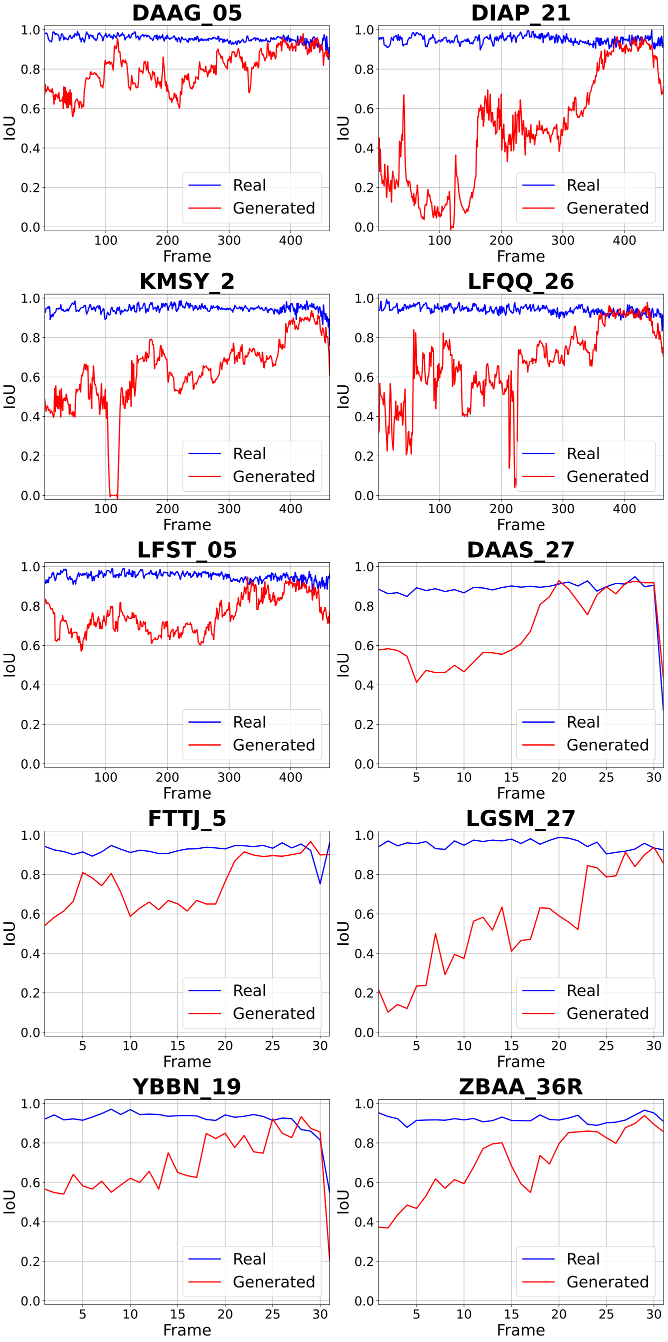

Fig. 7. Comparison of Real and Generated IoU per Frame.

Regarding the KMSY_2 experiment, although the IoU plot seems roughly stable, the IoU for the 0.75 threshold had the worst performance. When looking into the actual stylized images, the conversion generated several artifacts in the image, which could look similar to the runway when the plane is distant, causing the bounding boxes to be of the wrong size.

The test results were very similar to the training results, with

TABLE VI
GENERATED VIDEOS TRACKING RESULTS FOR TEST SET

Video ID	Style	F	SUC	P	P_{Norm}	$SR_{0.5}$	$SR_{0.75}$
DAAS_27	day	33	0.86	0.97	0.97	0.97	0.97
Generated	fog+dawn		0.68	1.00	0.79	0.79	0.42
FTTJ_5	day	31	0.91	0.97	1.00	1.00	1.00
Generated	dusk		0.75	1.00	0.97	1.00	0.52
LGSM_27	day	32	0.93	1.00	1.00	1.00	1.00
Generated	fog		0.55	1.00	0.75	0.59	0.31
YBBN_19	night+stars	32	0.89	0.91	1.00	1.00	0.97
Generated	day		0.68	0.97	0.88	0.97	0.41
ZBAA_36R	day	32	0.91	1.00	1.00	1.00	1.00
Generated	rain+night		0.69	1.00	0.84	0.84	0.50
Mean Ori.	-	-	0.90	0.97	0.99	0.99	0.99
Mean Gen.	-	-	0.67	0.99	0.84	0.84	0.43

the LGSM_27 video having the worst overall performance. We can clearly see in Image 6 that the fog was not well propagated, causing the entire image to blur and the runway to hardly appear. This is also shown in Figure 7, as it is clear that when away from the runway the IoU is very small, which means that the runway was barely recognized.

In general, the model was able to track the runway in the generated images, but the amount of artifacts caused the metrics to be much lower than expected, showcasing the limitations of using a diffusion model to translate the videos to another style.

IV. CONCLUSION

In this paper, we proposed a new video dataset for the task of tracking runways with per-frame labeling. We divided the dataset into two sets, one consisting of 10 real videos with planes approaching the runway and another with 10 synthetic videos from a literature dataset, but with new styles created through an image-to-video diffusion model.

For the real videos, we performed an analysis using a runway tracking model from the literature to verify the quality of the dataset. Even though the model, which was trained using only synthetic videos, had trouble to perform the tracking, it made it possible to understand the possible flaws in the annotation process of the videos and in which parts of the video the labels needed adjustments. For future work, adjustments could be considered beforehand while annotating the dataset.

As for the synthetic videos, we used videos that were used to train and test the same literature tracking model, and generated new videos using a diffusion model to transfer their style and increase the amount of varied conditions in the original dataset. We verified that even though there were some good quality videos generated, overall the tracking results were worse for the generated videos, indicating that their frames content was further than expected from the source material. For future work, we would like to measure the realism and quality of the synthetic videos, along with their distance from the source videos, allowing us to better select the style images and final dataset videos. We would also like to train a model using these diffused videos along with the original dataset videos, to check if this new generated content would improve the tracking model performance on real videos.

ACKNOWLEDGMENT

This work is supported by FAPEMIG - Fundação de Amparo à Pesquisa do Estado de Minas Gerais and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brazil (CAPES).

REFERENCES

[1] J. P. Ferreira, J. P. Pinto, J. Moura, Y. Li, C. L. Castro, and P. Angelov, "Vision-Based Landing Guidance Through Tracking and Orientation Estimation," in *Proceedings - 2025 IEEE Winter Conference on Applications of Computer Vision, WACV 2025*. IEEE, feb 2025, pp. 9681–9689. [Online]. Available: https://ieeexplore.ieee.org/document/10943679/ 1, 2, 3, 4, 7

[2] Q. Song, M. Lin, W. Zhan, S. Yan, L. Cao, and R. Ji, "Univst: A unified framework for training-free localized video style transfer," 2025. [Online]. Available: https://arxiv.org/abs/2410.20084 1, 4

[3] L. Lin, H. Fan, Z. Zhang, Y. Wang, Y. Xu, and H. Ling, "Tracking meets lora: Faster training, larger model, stronger performance," in *ECCV*, 2024. 1, 2, 4

[4] W. Chen, Z. Zhang, L. Yu, and Y. Tai, "BARS: a benchmark for airport runway segmentation," *Applied Intelligence*, vol. 53, no. 17, pp. 20 485–20 498, sep 2023. [Online]. Available: https://link.springer.com/10.1007/s10489-023-04586-5 1, 3

[5] Q. Wang, W. Feng, H. Zhao, B. Liu, and S. Lyu, "VALNet: Vision-Based Autonomous Landing with Airport Runway Instance Segmentation," *Remote Sensing*, vol. 16, no. 12, 2024. 1, 3

[6] M. Ducoffe, M. Carrere, L. Féliers, A. Gauffriau, V. Mussot, C. Pagetti, and T. Sammour, "LARD - Landing Approach Runway Detection - Dataset for Vision Based Landing," Apr. 2023, working paper or preprint. [Online]. Available: https://hal.science/hal-04056760 1, 3, 7

[7] K. Wada, "Labelme: Image Polygonal Annotation with Python," https://github.com/wkentaro/labelme. 1

[8] M. Chen and Y. Hu, "An image-based runway detection method for fixed-wing aircraft based on deep neural network," *IET Image Processing*, vol. 18, no. 8, pp. 1939–1949, jun 2024. [Online]. Available: https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/ipr2.13087 1, 3

[9] N. Bor, N. Pervan Akman, and A. Berkol, "AeroRunway: Diverse Weather and Time of Day Aerial Dataset for Autonomous Landing Training," *Journal of Advanced Research in Natural and Applied Sciences*, vol. 10, no. 3, pp. 735–746, sep 2024. [Online]. Available: http://dergipark.org.tr/en/doi/10.28979/jarnas.1500916 1, 3

[10] G. de Paula, "LANDING IN THE LATENT SPACE - building labeled synthetic runway datasets with a data augmentation pipeline that uses diffusion models," Bachelor's Thesis, University of London, 2025. 2, 3, 6

[11] J. Zhang, S. Gould, and I. Ben-Shabat, "Vidat—ANU CVML video annotation tool," https://github.com/anucvml/vidat, 2020. 2

[12] Z. Yin, K. Chen, X. Bai, R. Jiang, J. Li, H. Li, J. Liu, Y. Xiang, J. Yu, and M. Zhang, "Asurvey: Spatiotemporal consistency in video generation," 2025. [Online]. Available: https://arxiv.org/abs/2502.17863 4

[13] Z. Ye, H. Huang, X. Wang, P. Wan, D. Zhang, and W. Luo, "Stylemaster: Stylize your video with artistic generation and translation," 2024. [Online]. Available: https://arxiv.org/abs/2412.07744 4

[14] J. Song, C. Meng, and S. Ermon, "Denoising Diffusion Implicit Models," *ICLR 2021 - 9th International Conference on Learning Representations*, 2021. 4

[15] M. Müller, A. Bibi, S. Giancola, S. Alsubaihi, and B. Ghanem, "TrackingNet: A Large-Scale Dataset and Benchmark for Object Tracking in the Wild," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 11205 LNCS, pp. 310–327. [Online]. Available: https://link.springer.com/10.1007/978-3-030-01246-5_19 4

[16] C.-H. Lin, Z. Wang, R. Liang, Y. Zhang, S. Fidler, S. Wang, and Z. Gojcic, "Controllable weather synthesis and removal with video diffusion models," 2025. [Online]. Available: https://arxiv.org/abs/2505.00704 6