Generating Synthetic Training Data for Deep Learning-based UAV Trajectory Prediction

Stefan Becker[1][a], Ronny Hug[1][b], Wolfgang Huebner[1][c],
Michael Arens[1][d] and Brendan T. Morris[2][e]
[1] *Fraunhofer Center for Machine Learning, Fraunhofer IOSB, Ettlingen, Germany*
[2] *University of Nevada, Las Vegas, U.S.A.*

Keywords: Unmanned-Aerial-Vehicle (UAV), Synthetic Data Generation, Trajectory Prediction, Deep-learning, Recurrent Neural Networks (RNNs), Training Data, Quadrotors.

Abstract: Deep learning-based models, such as *recurrent neural networks* (RNNs), have been applied to various sequence learning tasks with great success. Following this, these models are increasingly replacing classic approaches in object tracking applications for motion prediction. On the one hand, these models can capture complex object dynamics with less modeling required, but on the other hand, they depend on a large amount of training data for parameter tuning. Towards this end, we present an approach for generating synthetic trajectory data of *unmanned-aerial-vehicles* (UAVs) in image space. Since UAVs, or rather quadrotors are dynamical systems, they can not follow arbitrary trajectories. With the prerequisite that UAV trajectories fulfill a smoothness criterion corresponding to a minimal change of higher-order motion, methods for planning aggressive quadrotors flights can be utilized to generate optimal trajectories through a sequence of 3D waypoints. By projecting these maneuver trajectories, which are suitable for controlling quadrotors, to image space, a versatile trajectory data set is realized. To demonstrate the applicability of the synthetic trajectory data, we show that an RNN-based prediction model solely trained on the generated data can outperform classic reference models on a real-world UAV tracking dataset. The evaluation is done on the publicly available *ANTI-UAV* dataset.

1 INTRODUCTION

The rise of *unmanned-aerial-vehicles* (UAVs), such as quadrotors, in the consumer market has led to concerns about associated potential risks for security or privacy. The potential intended or unintended misuses pertain to various areas of public life, including locations, such as airports, mass events, or public demonstrations (Laurenzis et al., 2020). Thus, automated UAV detection and tracking have become increasingly important for security services for anticipating UAV behavior. Video-based solutions offer the benefit of covering large areas and are cost-effective to acquire (Sommer et al., 2017). The basic components of such a video-based approach are the appearance model and the prediction model which is traditionally realized with Bayesian filter. The prediction model tasks within detection and tracking pipelines, among others, are the prediction of the object behavior and bridging detection failures. Following the success of deep learning-based models in various sequence processing tasks, these models become the standard choice for object motion prediction. A significant drawback of deep learning models is the requirement of a large amount of training data.

In order to overcome the problem of limited training data in the context of UAV tracking in image sequences, this paper proposes to utilize methods from planning aggressive UAV flights to generate suitable and versatile trajectories. The synthetically generated 3D trajectories are mapped into image space before they serve as training data for deep learning prediction models.

Despite the increasing number of trajectory datasets for object classes like pedestrians (e.g., *TrajNet++* dataset (Kothari et al., 2021)) or vehicles (e.g., *InD* dataset (Bock et al., 2020)), datasets with

[a] https://orcid.org/0000-0001-7367-2519
[b] https://orcid.org/0000-0001-6104-710X
[c] https://orcid.org/0000-0001-5634-6324
[d] https://orcid.org/0000-0002-7857-0332
[e] https://orcid.org/0000-0002-8592-8806

Becker, S., Hug, R., Huebner, W., Arens, M. and Morris, B.
Generating Synthetic Training Data for Deep Learning-based UAV Trajectory Prediction.
DOI: 10.5220/0010621400003061
In *Proceedings of the 2nd International Conference on Robotics, Computer Vision and Intelligent Systems (ROBOVIS 2021)*, pages 13-21
ISBN: 978-989-758-537-1

UAV trajectories and UAV in general are very limited. For the aforementioned object classes of pedestrians and vehicles, mostly RNN-based deep learning models are successfully applied for trajectory prediction (Alahi et al., 2016). Independent of the existing deep learning variants for trajectory prediction relying on *generative adversarial networks* (GANs) (Amirian et al., 2019; Gupta et al., 2018), *temporal convolution networks* (TCNs) (Becker et al., 2018; Nikhil and Morris, 2018), and *transformers* (Giuliari et al., 2021; Saleh, 2020), an RNN-based prediction model is chosen as reference. The reader is referred to these surveys (Rasouli, 2020; Rudenko et al., 2020; Kothari et al., 2021) for a comprehensive overview of current deep learning-based approaches for trajectory prediction.

Although the focus is on motion prediction and not on appearance modeling for UAV detection and tracking, we throw a brief glance at some of the current approaches that rely on images or use other modalities. Besides image-based methods, common modalities for UAV detections are RADAR, acoustics, radio-frequencies, and LIDAR. Comparisons of key characteristics of different deep learning-based approaches on single or fused modality information are presented in Samaras et al. (2019); Taha and Shoufan (2019); Unlu et al. (2019). A review with the focus on radar-based UAV detection methods is given by Christnacher et al. (2016). Approaches for acoustic sensors are, for example, presented in the work of Kartashov et al. (2020) and Jeon et al. (2017). An example to detect and identify UAVs based on their radio frequency signature is the approach of Xiao and Zhang (2019). For LIDAR, there exist, for example, the approaches of Hammer et al. (2019, 2018). Image-based approaches can be divided into using electro-optical sensors or infrared sensors. Their appearance modeling, however, is very similar. A further division can be made into one-stage strategies or two-stage strategies. In one-stage strategies, a direct classification and localization is applied. In two-stage strategies, a general (moving) object detection is followed by a classification step. For fixed cameras, the latter strategy is preferred. Different image-based approaches are, for example, presented in Schumann et al. (2018); Schumann et al. (2017), Sommer et al. (2017), Müller and Erdnüß (2019), and Rozantsev et al. (2017).

The paper is structured as follows. Section 2 presents the proposed methods for generating realistic UAV trajectory data in image sequences. Section 3 briefly introduces the used RNN-based UAV trajectory prediction model. In addition to an analysis of the diversity of the synthetically generated data, section 4 includes an evaluation on the real-world *ANTI-UAV* dataset (Jiang et al., 2021). Finally, a conclusion is given in section 5.

2 SYNTHETIC DATA GENERATION

In this section the proposed approach for generating realistic, synthetic trajectory data is presented.

Minimum Snap Trajectories (MST): UAVs can not fly arbitrary trajectories due to the fact that they are dynamical systems with strict constraints on achievable velocities, accelerations and inputs. These constraints determine optimal trajectories with a series of waypoints in a set of positions and orientations in conjunction with control inputs (Mellinger, 2012). Thus, the goal of trajectory generation in controlling UAVs is to generate inputs to the motion control system, which ensures that the planned trajectory can be executed. Here, we apply an explicit optimization method that enables autonomous, aggressive, high-speed quadrotor flight through complex environments. In the remainder of this paper, the terms quadrotor and UAV are used interchangeably, although there exist various other UAV concepts. The principle procedure can be adapted to all other designs. Since we are only interested in the trajectory data, the actual control design can, to some extent, be neglected as long as the planned target trajectory is suitable for control.

Minimum snap trajectories (MST) have proven very effective as quadrotor trajectories since the motor commands and attitude accelerations of the UAV are proportional to the snap, or the fourth derivative, of the path (Richter et al., 2016). Of course, the difference between the target trajectory and the executed trajectory depends on the actual controller and physical limitations (e.g., maximum speed) of a UAV. Firstly, physical constraints can be considered in planning. Secondly, in most cases, a well-designed controller can closely follow the target trajectory. For our purpose, the trajectory of the actual flight can, in a way, be seen as only a slight variation of the target trajectory. However, quadrotor dynamics relying on four control inputs $(i_1, \cdots, i_4)$ for nested feedback control (inner attitude control loop and outer position control loop, see for example Michael et al. (2010)) are differential flat (Mellinger and Kumar, 2011). In other words, the states and the inputs can be written as functions of four selected *flat outputs* and their derivatives. This facilitates the automated generation of trajectories since any smooth trajectory (with reasonably bounded derivatives) in the space of *flat outputs* can

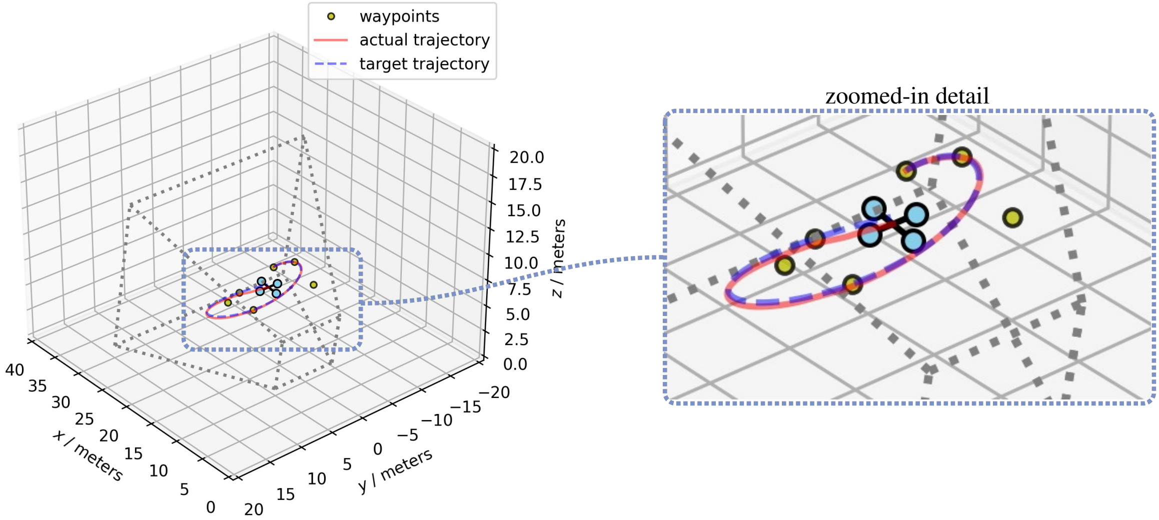


Figure 1: Visualization of a generated MST (target trajectory) along waypoints in the viewing frustum of a camera. The MST is highlighted in purple. The sampled waypoints in the viewing frustum of the camera are shown in yellow. The corresponding UAV flights with a PD controller as proposed from Michael et al. (2010) are shown in red.

be followed by the quadrotor. Following Mellinger et al., the *flat outputs* are given by $p = [\mathbf{r}^\intercal, \psi]^\intercal$, where $\mathbf{r} = [x, y, z]^\intercal$ are the coordinates of the center of mass in the world coordinate system and ψ is the yaw angle. The *flat outputs* at a given time t are given by $p(t)$, which defines a smooth curve. A waypoint denotes a position in space along a yaw angle. Trajectory planning specifies navigating through m waypoints at specified times by staying in a safe corridor. Trivial trajectories such as straight lines lead to discontinuities in higher-order motion. Thus, such trajectories are undesirable because they include infinite curvatures at waypoints, which require the quadrotor to stop at each waypoint. The differentiability of polynomial trajectories makes them a natural choice for use in a differentially flat representation of the quadrotor dynamics. Thus, for following a specific trajectory $p_T(t) = [\mathbf{r}_T^\intercal, \psi_T(t)]^\intercal$ (with controller for a UAV), the smooth trajectory $p_T(t)$ is defined as piecewise polynomial functions of order n over m time intervals:

$$p_T(t) = \begin{cases} \sum_{i=0}^{n} p_{T_{i1}}(t^i) & t_0 \leqslant t \leqslant t_1 \\ \sum_{i=0}^{n} p_{T_{i2}}(t^i) & t_1 \leqslant t \leqslant t_2 \\ \vdots \\ \sum_{i=0}^{n} p_{T_{im}}(t^i) & t_{m-1} \leqslant t \leqslant t_m \end{cases} \tag{1}$$

The goal is to find trajectories that minimize functionals which can be written using these basis functions. These kinds of problems can be solved with tools from the calculus of variations and are standard problems in robotics (Craig, 1989). Hence, in order to find the smooth target trajectory $p_T(t)_{tar}$, the integral of the $k_r{}^{\text{th}}$ derivative of position squared and the $k_\psi{}^{\text{th}}$ derivative of yaw angle squared is minimized:

$$p_T(t)_{tar} = \arg\min_{p_T(t)} \int_{t_o}^{t_m} c_r \left\| \frac{\mathrm{d}^{k_r}\mathbf{r}_T}{\mathrm{d}t^{k_r}} \right\|^2 + c_\psi \frac{\mathrm{d}^{k_\psi}\psi_T}{\mathrm{d}t^{k_\psi}}^2 \mathrm{d}t \tag{2}$$

Here, c_r and c_ψ are constants to make the integrand non-dimensional. Continuity of the k_r derivatives of $\mathbf{r}_T$ and k_ψ derivatives of ψ_T is enforced as a criterion for smoothness. In other words, the continuity of the derivatives determines the boundary conditions at the waypoints. As mentioned above, some UAV control input depends on the fourth derivative of the positions and the second derivative of the yaw. Accordingly, $p_T(t)_{tar}$ is calculated by minimizing the integral of the square of the norm of the snap ($k_r = 4$), and for the yaw angle, $k_\psi = 2$ holds.

The above problem can be formulated as a quadratic problem QP (Bertsekas, 1999). Thereby, $p_{T_{ij}} = [x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}]^\intercal$ are written as a $4nm \times 1$ vector $\vec{c}$ with decision variables $\{x_{T_{ij}}, y_{T_{ij}}, z_{T_{ij}}, \psi_{T_{ij}}\}$:

$$\begin{aligned} &\min \vec{c}^\intercal Q \vec{c} + \vec{f}^\intercal \vec{c} \\ &\text{subject to } A\vec{c} \leq \vec{b}. \end{aligned} \tag{3}$$

Here, the objective function incorporates the minimization of the functional while the constraints can be used to satisfy constraints on the flat outputs and their derivatives and thus constraints on the states and the inputs. The initial conditions, final condition, or intermediate condition on any derivative of the trajectory are specified as equality constraint in 3. For a more detailed explanation on generating MSTs, how to incorporate corridor constraints, and how to calculate the angular velocities, angular accelerations, total thrust, and moments required over the entire trajectory for the controller, the reader is referred to the work of Mellinger and Kumar (2011). Richter et al. (2016) presented an extended version of MST generation, where the solution of the quadratic problem is numerically more stable.

Training Data Generation: With the described method, we can generate MSTs suitable to aggressive maneuver flights for UAV control in a 3D environment. In order to generate versatile trajectory data in image space, further steps are required. The overall generation pipeline is explained in the following. Firstly, a desired camera model with known intrinsic parameters is selected. The selection depends on the targeted sensor set-up of the detection and tracking system. In our case, we choose a pinhole camera model without distortion effects loosely orientated on the *ANTI-UAV* dataset (Jiang et al., 2021) with regard to an intermediate image resolution (in pixels) (1176×640) between the infrared (640×512) and electro-optical camera (1920×1080) resolutions of the *ANTI-UAV* dataset. In case all camera parameters are known, the corresponding distortion coefficient should be considered. In the experiments the focal length (in pixels) is set to 1240, the principle point coordinates (in pixels) are set to $p_x = 579$, $p_y = 212$. For setting up the external parameters, the camera is placed close to the ground plane sampled from an uniform distribution $Uni(1m, 2m)$ to set the height above ground. The inclination angle is sampled from $Uni(10°, 20°)$. Given the fixed camera parameters, the viewing frustum is calculated for a chosen near distance to the camera center ($d_{near} = 10m$) and a far distance to the camera center ($d_{far} = 30m$). For generating a single MST, a set of waypoints inside the viewing frustum is sampled. The number of waypoints is randomly varied between 3 and 7. The travel time for each segment is approximated by using the Euclidean distance between two waypoints and a sampled constant speed for the UAV $Uni(1m/s, 8m/s)$. The resulting straight-line trajectory serves as an initial solution of the MST calculation. The frame rate of the camera is sampled from $Uni(10fps, 20fps)$ for every run. In our experiments, we assumed a completely free space in the viewing frustum. As mentioned, corridor constraints, which can be used for simulating an object to fly through, can be integrated with the method of Mellinger and Kumar (2011). For the synthetic training dataset, 1000 MSTs are generated. The main steps for the synthetic data generation pipeline of a single run are as follows:

- Selection of a desired camera model with known intrinsic parameters.
- Extrinsic parameters are sampled from the height and inclination angle distribution.
- Calculation of the corresponding viewing frustum with d_{near} and d_{far}.
- Sampling of waypoints inside viewing frustum.
- Estimation of the initial solution with fixed, sampled UAV speed.
- MST trajectory generation using the method of Mellinger and Kumar (2011).
- Projection of the 3D center of mass positions of the MST to image space using the camera parameters at fixed time intervals (reciprocal of the drawn frame rate of a single run).

This procedure is repeated until the desired number of samples are generated. Note that sanity checks and abort criteria for the trajectory are applied at the end and during every run. For example, requirements on the minimum and maximum length of consecutive image points. In Figure 1, generated MSTs along sampled waypoints in a 3D-environment are visualized. The viewing frustum for a fixed inclination angle of 15° is shown as a dotted gray line. The actual flight trajectory of a UAV is realized with a *proportional–derivative* (PD) controller as is proposed by Michael et al. (2010). Although the controller design is relatively simple, the UAV can closely follow the target trajectory. Thus, directly using the planned MST seems to be a legitimate design choice.

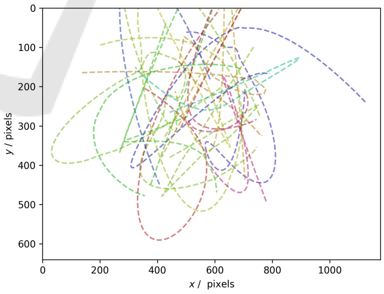


Figure 2: Visualization of generated UAV trajectories as observed from the camera.

In Figure 2, exemplary generated training samples are depicted. The figure illustrates the diversity in the generated trajectory data reflecting several motion prototypes present in trajectory data. A more detailed analysis of the diversity and suitability of the generated synthetic data is given in section 4. Before that, we will briefly introduce the deep learning-based reference prediction model.

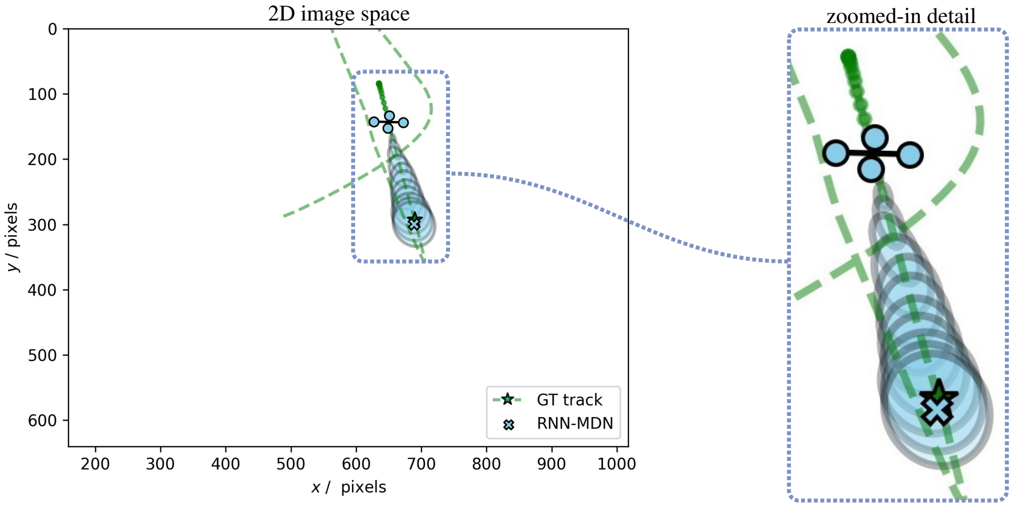


Figure 3: Visualization of the predicted distribution of an RNN-MDN on a synthetic UAV trajectory in image space for 12 steps into the future (on the left). A corresponding zoomed-in detail is shown on the right. The corresponding position of the ground truth (GT) track is highlighted with a green star.

3 PREDICTION MODEL

For trajectory prediction, an RNN-based model is considered, which predicts a distribution over the next N image positions and is conditioned on the previous observations $\vec{u}^{0:t}$. $\vec{u}^{0:t}$ are the image coordinates from time 0 up to t. As mentioned above, RNN-based models are a common choice from the variants of deep learning-based models for trajectory prediction. For generating the distribution over the next positions, the model outputs the parameter of a mixture density network (MDN). Such RNN-MDNs are applied to capture the motion of different object types. Originated from a model introduced by Graves (2014), modified versions have been successfully utilized to predict pedestrian (Alahi et al., 2016), vehicle (Deo and Trivedi, 2018) or cyclist (Pool et al., 2021) motions. Although these modified versions also incorporate some contextual cues (e.g., interactions from other objects), single objects motion is encoded with such an RNN-MDN variant.

Given an input sequence $\mathcal{U}$ of consecutive observed image positions $\vec{u}^t = (u^t, v^t)$ at time step t along a trajectory, the model generates a probability distribution over future positions $\{\vec{u}^{t+1}, \ldots, \vec{u}^{t+N}\}$. The model is realized as an RNN encoder. With an embedding of the inputs and using a single Gaussian component, the model can be defined as follows:

$$\begin{aligned}\vec{e}^t &= \mathrm{EMB}(\vec{u}^t; \vec{\Theta}_e),\\ \vec{h}^t &= \mathrm{RNN}(\vec{h}^{t-1}, \vec{e}^t; \vec{\Theta}_{RNN}),\\ \{\hat{\vec{\mu}}^{t+n}, \hat{\Sigma}^{t+n}\}_{n=1}^{N} &= \mathrm{MLP}(\vec{h}^t; \vec{\Theta}_{MLP})\end{aligned} \tag{4}$$

Here, $\mathrm{RNN}(\cdot)$ is the recurrent network, $\vec{h}$ the hidden state of the RNN, $\mathrm{MLP}(\cdot)$ the multilayer perceptron, and $\mathrm{EMB}(\cdot)$ an embedding layer. $\vec{\Theta}$ represents the weights and biases of the MLP, EMB, or respectively of the RNN. The model is trained by maximizing the likelihood of the data given the output Gaussian distribution. This results in the following loss function:

$$\mathcal{L}(\mathcal{U}) = -\sum_{n=1}^{N} -\log \mathcal{N}(\vec{u}^{t+n} | \hat{\vec{\mu}}^{t+n}, \hat{\Sigma}^{t+n}). \tag{5}$$

Implementation Details: The RNN-MDN is implemented using *Tensorflow* (Abadi et al., 2015) and is trained for 2000 epochs using an ADAM optimizer (Kingma and Ba, 2015) with a decreasing learning rate, starting from 0.01 with a learning rate decay of 0.95 and a decay factor of 1/10. The RNN hidden state and embedding dimension is 64. For the experiments, the *long short-term memory* (LSTM) (Hochreiter and Schmidhuber, 1997) RNN variant is utilized.

An example prediction of the RNN-MDN on a synthetically generated UAV trajectory in image space is depicted in Figure 3. On the left, the predicted distribution for 12 steps into the future is shown in blue. The corresponding ground truth position is marked as a green star. On the right, the corresponding 3D UAV trajectory is shown in green.

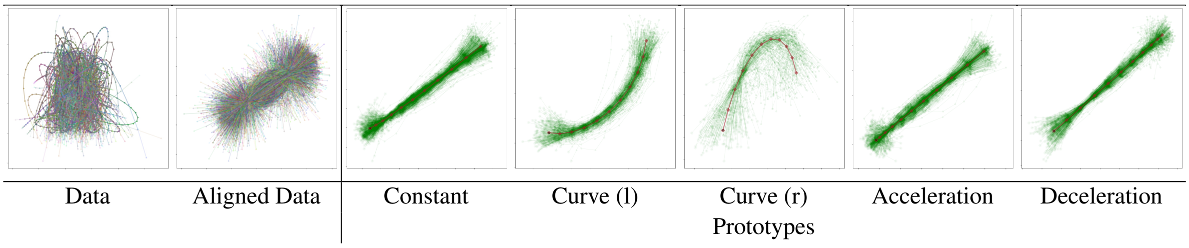


Figure 4: Data, aligned data and learned prototypes for the synthetic UAV trajectories projected to image space. The prototypes represent different motion patterns (from left to right): Constant velocity, curvilinear motion (left and right curve), acceleration and deceleration.

Table 1: Results for a comparison between the an RNN-MDN prediction model trained on the synthetic generated data, a Kalman filter with CV motion model, and using linear interpolation. The prediction is done for 8, 10, and 16 frames into the future.

	EO (1920 × 1080)					
	8/8		8/10		8/12	
Approach	FDE/pixels	σ_{FDE}/pixels	FDE/pixels	σ_{FDE}/pixels	FDE/pixels	σ_{FDE}/pixels
RNN-MDN	60.304	35.202	82.780	46.124	121.453	55.489
Kalman filter (CV)	81.061	60.333	110.041	84.530	179.459	104.408
Linear interpolation	86.998	67.113	119.106	89.067	183.558	108.522

	IR (640 × 512)					
	8/8		8/10		8/12	
Approach	FDE/pixels	σ_{FDE}/pixels	FDE/pixels	σ_{FDE}/pixels	FDE/pixels	σ_{FDE}/pixels
RNN-MDN	20.849	18.604	41.378	21.100	61.459	24.490
Kalman filter (CV)	22.340	24.630	43.172	32.522	68.012	37.047
Linear interpolation	24.235	26.517	45.434	35.837	69.665	37.714

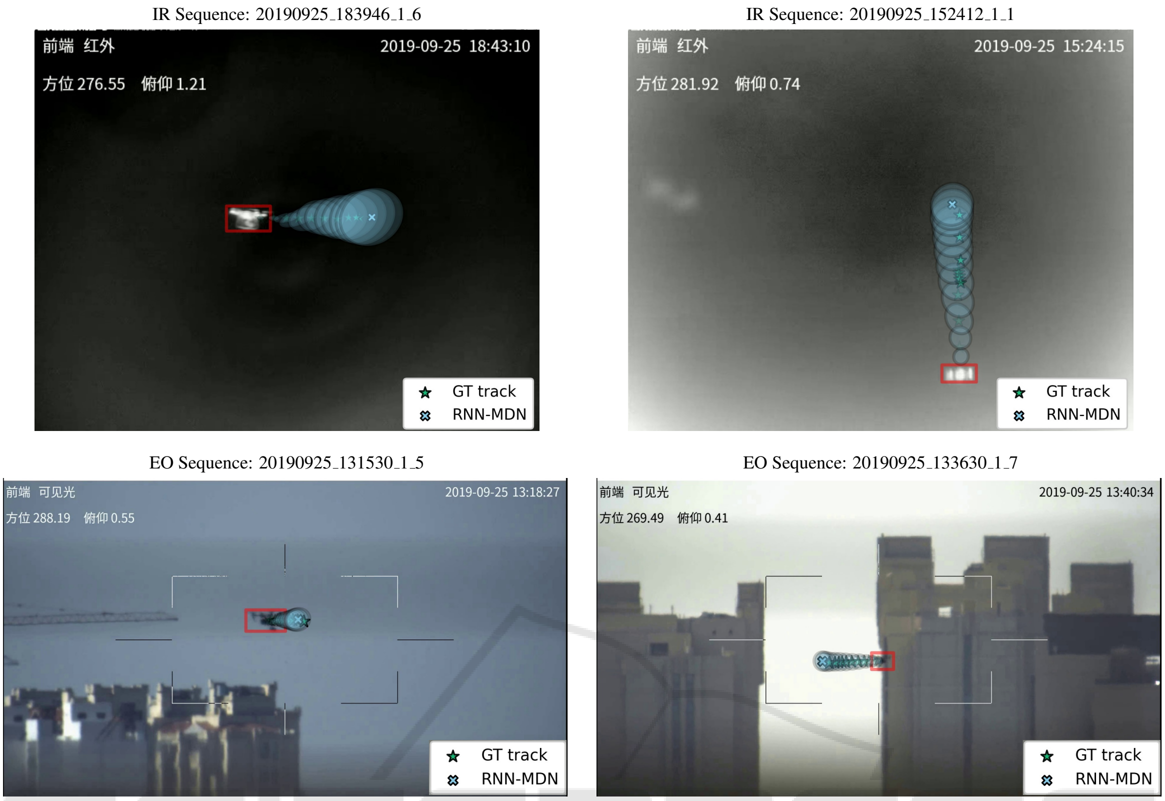
4 EVALUATION & ANALYSIS

This section analyzes the diversity of the generated synthetic data and the suitability for training deep-learning prediction models.

Diversity Analysis: For analyzing the diversity of the synthetically generated data, we use the approach of Hug et al. (2021). The approach learns a representation of the provided trajectory data by first employing a spatial sequence alignment, which enables a subsequent learning vector quantization (LVQ) stage. Each trajectory dataset can be reduced to a small number of prototypical sub-sequences specifying distinct motion patterns, where each sample can be assumed to be a variation of these prototypes (Hug et al., 2020). Thus, the resulting quantized representation of the trajectory data, the prototypes, reflect basic motion patterns, such as constant or curvilinear motion, while variations occur primarily in position, orientation, curvature and scale. For further details on the dataset analysis methods, the reader is referred to the work of Hug et al. (2021). The resulting prototypes of the generated training data are depicted in Figure 4.

The learned prototypes show that the generated synthetic data includes several different motion patterns. Besides, the diversity of the learned prototypes is visible. The dataset consists of, for example, distinguishable motion patterns reflecting constant velocity motion, curvilinear motion, acceleration, and deceleration.

Suitability Analysis: In order to analyze the suitability of the generated trajectory data, the RNN-MDN is trained as an exemplary deep learning-based prediction model according to section 3. For evaluation, the real-world *ANTI-UAV* dataset (Jiang et al., 2021) is used. The dataset consists of 100 high-quality, full HD video sequences (both electro-optical (EO) and infrared (IR)), spanning multiple occurrences of multi-scale UAVs, annotated with bounding boxes. As inputs for the prediction models, the center positions of the provided annotations are used. As classical reference models, a Kalman filter with a constant velocity (CV) motion model and linear interpolation are utilized. For the Kalman filter, the observation noise is assumed to be a white Gaussian noise process $\vec{w}^t \sim \mathcal{N}(0, (1.5pixels)^2)$. Thereby, the uncer-


IR Sequence: 20190925_183946_1_6

IR Sequence: 20190925_152412_1_1

EO Sequence: 20190925_131530_1_5

EO Sequence: 20190925_133630_1_7

Figure 5: Example predictions for the *Anti-UAV* dataset Jiang et al. (2021). The top images show two samples from the IR sequences. The bottom images depict two samples from the EO sequences.

tainty in the annotation is considered. The process noise is modeled as the acceleration increment during a sampling interval (white noise acceleration model Bar-Shalom et al. (2002)) with $\sigma_{CV}^2 = 0.5 {}^{pixels}/_{frame^2}$. For dealing with the minor annotation uncertainty, a small white Gaussian noise is added to the generated trajectory positions corresponding to the assumed observation noise. Since RNN are able to generalize from noisy inputs up to an extent (see for example (Becker et al., 2018)), the noisy trajectories are used for conditioning during training of the RNN-MDN. The performance is compared with the final displacement error (FDE) for three different time-horizons, in particular, 8 frames, 10 frames, and 12 frames into the future. The FDE is calculated as the average Euclidean distance between the predicted final positions and the ground truth positions.

The results for the EO and the IR video sequences are summarized in table 1. Although the RNN-MDN is solely trained on the synthetically generated UAV data, the model could outperform the traditional reference models on the EO and the IR sequences of the *ANTI-UAV* dataset. It should be noted that no camera motion compensation is applied. Thus, the results of all considered prediction models can be further improved. For longer prediction horizons, all errors are relatively high due to the maneuvering behavior of the UAVs. However, in case the prediction model is utilized to bridge detection failures for supporting the appearance model of the detection and tracking pipeline, the number of subsequent failures should be lower than the shown 12 frames time horizon. Since the RNN-MDN relying on synthetic data achieved better performance than the reference models, it is better suited to anticipate the short-term UAV behavior. In Figure 5, the predicted distributions of future positions are visualized for the EO sequences or respectively IR sequences. The ground truth future positions are highlighted as green stars. The covariance ellipses around the predicted position are shown in blue. The final predicted positions are marked as a blue cross.

5 CONCLUSION

This paper presents an approach for generating synthetic trajectory data of UAVs in image space. By utilizing methods for planning maneuvering UAV flights, minimum snap trajectories along a sequence of 3D waypoints are generated. By selecting the desired camera model with known camera parameters, the trajectories can be projected to the observer's perspective. To demonstrate the applicability of the synthetic trajectory data, we show that an RNN-MDN prediction model solely trained on the synthetically generated data is able to outperform classic reference models on a real-world UAV tracking dataset.

REFERENCES

Abadi, M. et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. 5

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., and Savarese, S. (2016). Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 961–971. 2, 5

Amirian, J., Hayet, J.-B., and Pettre, J. (2019). Social Ways: Learning Multi-Modal Distributions of Pedestrian Trajectories With GANs. In *Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2964–2972. 2

Bar-Shalom, Y., Kirubarajan, T., and Li, X.-R. (2002). *Estimation with Applications to Tracking and Navigation*. John Wiley & Sons, Inc., New York, NY, USA. 7

Becker, S., Hug, R., Hübner, W., and Arens, M. (2018). RED: A simple but effective Baseline Predictor for the TrajNet Benchmark. In *The European Conference on Computer Vision Workshops (ECCVW)*, volume 11131 of *Lecture Notes in Computer Science*, pages 138–153. Springer. 2, 7

Bertsekas, D. (1999). *Nonlinear Programming*. Athena Scientific. 3

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., and Eckstein, L. (2020). The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections. *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934. 1

Christnacher, F., Hengy, S., Laurenzis, M., Matwyschuk, A., Naz, P., Schertzer, S., and Schmitt, G. (2016). Optical and acoustical UAV detection. In Kamerman, G. and Steinvall, O., editors, *Electro-Optical Remote Sensing X*, volume 9988, pages 83 – 95. International Society for Optics and Photonics, SPIE. 2

Craig, J. (1989). *Introduction to Robotics: Mechanics and Control*. Addison-Wesley Longman Publishing Co., Inc., USA, 2nd edition. 3

Deo, N. and Trivedi, M. M. (2018). Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1179–1184. 5

Giuliari, F., Hasan, I., Cristani, M., and Galasso, F. (2021). Transformer Networks for Trajectory Forecasting. In *International Conference on Pattern Recognition (ICPR)*, pages 10335–10342. 2

Graves, A. (2014). Generating sequences with recurrent neural networks. 5

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. (2018). Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2255–2264. 2

Hammer, M., Borgmann, B., Hebel, M., and Arens, M. (2019). UAV detection, tracking, and classification by sensor fusion of a 360° lidar system and an alignable classification sensor. In Turner, M. D. and Kamerman, G. W., editors, *Laser Radar Technology and Applications XXIV*, volume 11005, pages 99 – 108. International Society for Optics and Photonics, SPIE. 2

Hammer, M., Hebel, M., Laurenzis, M., and Arens, M. (2018). Lidar-based detection and tracking of small UAVs. In Buller, G. S., Hollins, R. C., Lamb, R. A., and Mueller, M., editors, *Emerging Imaging and Sensing Technologies for Security and Defence III; and Unmanned Sensors, Systems, and Countermeasures*, volume 10799, pages 177 – 185. International Society for Optics and Photonics, SPIE. 2

Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780. 5

Hug, R., Becker, S., Hübner, W., and Arens, M. (2021). Quantifying the complexity of standard benchmarking datasets for long-term human trajectory prediction. *IEEE Access*, 9:77693–77704. 6

Hug, R., Becker, S., Hübner, W., and Arens, M. (2020). A short note on analyzing sequence complexity in trajectory prediction benchmarks. In *Workshop on Long-term Human Motion Prediction (LHMP)*. 6

Jeon, S., Shin, J., Lee, Y., Kim, W., Kwon, Y., and Yang, H. (2017). Empirical study of drone sound detection in real-life environment with deep neural networks. In *European Signal Processing Conference (EUSIPCO)*, pages 1858–1862. 2

Jiang, N., Wang, K., Peng, X., Yu, X., Wang, Q., Xing, J., Li, G., Zhao, J., Guo, G., and Han, Z. (2021). Anti-uav: A large multi-modal benchmark for uav tracking. 2, 4, 6, 7

Kartashov, V., Oleynikov, V., Koryttsev, I., Sheiko, S., Zubkov, O., Babkin, S., and Selieznov, I. (2020). Use of acoustic signature for detection, recognition and direction finding of small unmanned aerial vehicles. In *2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET)*, pages 1–4. 2

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*. 5

Kothari, P., Kreiss, S., and Alahi, A. (2021). Human trajectory forecasting in crowds: A deep learning perspective. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–15. 1, 2

Laurenzis, M., Rebert, M., Schertzer, S., Bacher, E., and Christnacher, F. (2020). Prediction of MUAV flight behavior from active and passive imaging in complex environment. In *Laser Radar Technology and Applications*, volume 11410, pages 10–17. International Society for Optics and Photonics, SPIE. 1

Mellinger, D. (2012). *Trajectory Generation and Control for Quadrotors*. PhD thesis, University of Pennsylvania. 2

Mellinger, D. and Kumar, V. (2011). Minimum snap trajectory generation and control for quadrotors. In *International Conference on Robotics and Automation (ICRA)*, pages 2520–2525. 2, 4

Michael, N., Mellinger, D., Lindsey, Q., and Kumar, V. (2010). The grasp multiple micro-uav testbed. *IEEE Robotics Automation Magazine*, 17(3):56–65. 2, 3, 4

Müller, T. and Erdnüß, B. (2019). Robust drone detection with static VIS and SWIR cameras for day and night counter-UAV. In *Counterterrorism, Crime Fighting, Forensics, and Surveillance Technologies*, volume 11166, pages 58–72. International Society for Optics and Photonics, SPIE. 2

Nikhil, N. and Morris, B. T. (2018). Convolutional Neural Network for Trajectory Prediction. In *The European Conference on Computer Vision Workshops (ECCVW)*, volume 11131 of *Lecture Notes in Computer Science*, pages 186–196. Springer. 2

Pool, E., Kooij, J. F. P., and Gavrila, D. M. (2021). Crafted vs. learned representations in predictive models - a case study on cyclist path prediction. *IEEE Transactions on Intelligent Vehicles*, pages 1–1. 5

Rasouli, A. (2020). Deep Learning for Vision-based Prediction: A Survey. *arXiv preprint arXiv:2007.00095*. 2

Richter, C., Bry, A., and Roy, N. (2016). *Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments*, pages 649–666. Springer, Cham. 2, 4

Rozantsev, A., Lepetit, V., and Fua, P. (2017). Detecting flying objects using a single moving camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(5):879–892. 2

Rudenko, A., Palmieri, L., Herman, M., Kitani, K. M., Gavrila, D. M., and Arras, K. O. (2020). Human Motion Trajectory Prediction: A Survey. *The International Journal of Robotics Research*, 39:895 – 935. 2

Saleh, K. (2020). Pedestrian Trajectory Prediction using Context-Augmented Transformer Networks. *arXiv preprint arXiv:2012.01757*. 2

Samaras, S., Diamantidou, E., Ataloglou, D., Sakellariou, N., Vafeiadis, A., Magoulianitis, V., Lalas, A., Dimou, A., Zarpalas, D., Votis, K., Daras, P., and Tzovaras, D. (2019). Deep learning on multi sensor data for counter uav applications—a systematic review. *Sensors*, 19(22). 2

Schumann, A., Sommer, L., Klatte, J., Schuchert, T., and Beyerer, J. (2017). Deep cross-domain flying object classification for robust uav detection. In *International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. 2

Schumann, A., Sommer, L., Müller, T., and Voth, S. (2018). An image processing pipeline for long range UAV detection. In *Emerging Imaging and Sensing Technologies for Security and Defence III; and Unmanned Sensors, Systems, and Countermeasures*, volume 10799, pages 186–194. International Society for Optics and Photonics, SPIE. 2

Sommer, L., Schumann, A., Müller, T., Schuchert, T., and Beyerer, J. (2017). Flying object detection for automatic uav recognition. In *International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. 1, 2

Taha, B. and Shoufan, A. (2019). Machine learning-based drone detection and classification: State-of-the-art in research. *IEEE Access*, 7:138669–138682. 2

Unlu, E., Zenou, E., Riviere, N., and Dupouy, P.-E. (2019). Deep learning-based strategies for the detection and tracking of drones using several cameras. *IPSJ Transactions on Computer Vision and Applications*, 11(1):7. 2

Xiao, Y. and Zhang, X. (2019). Micro-uav detection and identification based on radio frequency signature. In *International Conference on Systems and Informatics (ICSAI)*, pages 1056–1062. 2