Transformer-Based State Estimation for Multi-Target Tracking: Sensitivity Analysis against Varying Kinematic Parameters and Clutter Density

Valentin Sonntag[1,2], Jean-Marc Le Caillec[2], Alain Peres[1] and Stéphane Devaud[1]
[1] Thales Land and Air Systems, 3 Avenue Charles Lindbergh, 94150 Rungis, France
[2] Lab-STICC UMR CNRS 6285, IMT Atlantique, 655 Avenue du Technopôle, 29280 Plouzané, France
Email: {valentin.sonntag, alain.peres, stephane.devaud}@thalesgroup.com, jm.lecaillec@imt-atlantique.fr

***Abstract*—An exploration of a Transformer's behavior is proposed in the context of multi-target tracking. We investigate the behavior properties of the state predictions made by the Transformer-based method through sensitivity analysis. The experiments focus on varying kinematic parameters, clutter density, and number of objects. The Transformer-based method demonstrates consistent accuracy on the training domain and generalization capability relative to clutter density. The results show that the Transformer Tracker outperforms the Kalman Filter and Extended Kalman Filter in terms of overall accuracy. Moreover, it is capable of adapting to the training data, building data knowledge to improve prediction accuracy. The experiments highlight insufficient generalization on the number of objects. They also provide preliminary insight into the system's explainability. Finally, we discuss potential limitations and identify future research directions.**

***Index Terms*—State Estimation, Multi-target Tracking, Data Association, Attention Mechanism, Transformer, Kalman Filter**

I. Introduction

Air defense is facing increasingly complex scenarios. In fact, the challenges are growing in these theaters of operation with the increasing diversity of threats, ranging in size from different orders of magnitude, and which may be hypersonic, highly maneuverable, stealthy, and autonomous. Moreover, these threats are becoming more numerous and tend to work collaboratively. In a high-intensity conflict, control of the airspace provides enhanced intelligence capabilities and freedom of action for air, naval, and ground forces and assets, while limiting those of opposing forces [1]. Air defense aims to reduce the effectiveness of hostile airborne operations, such as reconnaissance and electronic warfare platforms, troop and material transport, fighters, bombers, conventional or nuclear missiles, in order to protect armed forces, civilians, military infrastructures, sensitive and high-value industry assets. It consists of detection, command, and control systems, as well as the means deployed to counter these threats: operational readiness, air interception, jamming, active air and missile defense. Therefore, these systems need to be upgraded accordingly to the complexity of emerging threats and air defense scenarios.

This work was supported by the French Defense Innovation Agency (AID) and Thales Land and Air Systems.

The state of an airspace is assessed using methods to detect and identify the various objects within it, while monitoring and tracking them by determining their kinematics with tracking techniques [2]. This also enables predicting the future state of this airspace. A major challenge lies in associating the available data from various sensors with previously determined kinematics. Managing the detection and tracking of multiple objects in an electromagnetically dense environment complicates the task, leading to false or missed detections. While some methods can associate this data based on specified criteria [3], their complexity often hinders the real-time processing constraint. Additionally, these methods may require information about the environment or objects that is often limited or unavailable [2]. Once the data association task is completed, techniques like Kalman Filters (KF) [4] are used to update the motion properties of the tracked objects. However, these methods are limited by assumptions about the kinematic models, which may not accurately reflect reality, posing a challenge for more complex movements such as those of maneuvering or hypersonic objects.

Recent methods based on Artificial Intelligence (AI), and in particular works emphasizing the use of deep learning, have attempted to overcome the limitations of non-AI methods, in order to perform data association, such as for the traveling salesman problem [5], [6] or the multi-object tracking problem [7]. These techniques are based on the use of an encoder-decoder architecture for sequence-to-sequence prediction. Other solutions [8]–[13] focus on improving certain elements or replacing the object kinematics update methods entirely, in order to overcome their weaknesses. These include reduced algorithmic complexity, more flexible and adaptable algorithms through the ability to learn properties of the environment and tracked objects through supervised model training. Finally, the attention mechanism used in Transformers [14] enables input data to be processed independently of the input sequence order, which is a limitation with sequential methods such as recurrent networks.

In this paper, we investigate a Transformer-based approach that aims to simultaneously address data association and state estimation [15], by adapting the architecture of the Transformer for the multi-target tracking problem. This work compares the proposed method with Kalman Filters, and studies its performance under varying scenario parameters, such as clutter density, the number of objects, or their kinematics.

Transformers

The attention mechanism is the core process involved in Transformers. It enables the model to selectively focus on relevant information, while simultaneously considering the input elements. Thus it is permutation invariant and is capable of handling long-range dependencies. The attention mechanism computes attention scores that highlight the importance of different context elements. The attention scores are based on the compatibility between the input elements, the queries Q, and the context elements, the keys K. They are used to weigh the values V of the keys to compute the context of each query relative to the keys. This is similar to searching for a query in a database, with the queries being the elements of Q, and the database represented by K and V.

The attention mechanism can be applied simultaneously across n_{heads} multiple attention heads, unlocking the ability to focus on different aspects of the inputs, and therefore consequently weigh the context depending on the aspects that are learned during learning.

The Transformer follows an encoder-decoder architecture. The encoder is a stack of N encoder blocks which are the succession of a multi-head self-attention layer and a FeedForward Network (FFN) layer. The decoder is a stack of M decoder blocks, which are the succession of a multi-head self-attention layer, a multi-head cross-attention layer, and an FFN layer. In addition, each layer has a residual connection, which allows better propagation of the gradient and information through the neural network.

II. TRANSFORMER-BASED TRACKER

The following approach proposes to estimate the kinematics of the potential objects present in a spatial and temporal observation window, using single-sensor measurements.

A. Architecture

The approach is based on the Transformer architecture, which is suitable for approximating sequence-to-sequence functions. Thus, it comes down to a translation problem, where the input data represented in the ”measurement language” of the sensor, are summarized into a ”state language”, which describes the characteristics of each detected object. In addition, we assume that a single common input token format for the data reduces the complexity of the system, making it less challenging to design, test, update, and maintain than more conventional tracking methods. The work presented in [15] aims at adapting this architecture for the multi-target tracking problem, as shown in Fig. 1.

The input data is transformed by affine transformations into the latent space, both in encoder and decoder, and are learned through training. Similarly, the output of the decoder passes from the latent space to the output space through a final affine transformation.

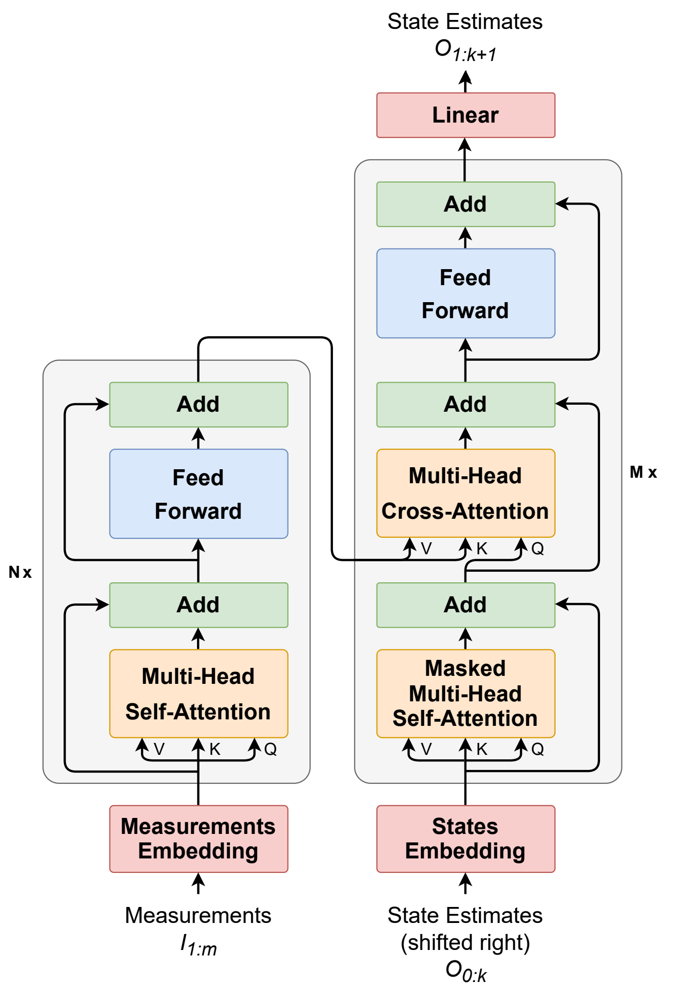


Fig. 1. Illustration of the Transformer architecture applied for multi-target tracking.

The encoder input $I_{1:m}$ of the Transformer, i.e. the input data, is composed of m measurements, which correspond to measurements occurring within a fixed time window and generated by a sensor, with $I_j = [t_j, x_j^{meas}, y_j^{meas}]$, $j \in [1..m]$, where t_j is the measurement time, x_j^{meas} is the measured position on the x-axis, and y_j^{meas} is the measured position on the y-axis.

The decoder output $O_{1:k+1}$, i.e. the ”prediction” is composed of k states and the end state, indicating the end of the decoding process such as the end token in Natural Language Processing (NLP), and which all variables are equal to val_{end}. $O_{1:k}$ corresponds to the k current state estimates of the corresponding objects, with $O_i = [x_i, y_i, v_{xi}, v_{yi}, \dot{\theta}_i]$, $i \in [1..k]$, where x_i is the estimated x-axis position, y_i is the estimated y-axis position, v_{xi} is the estimated x-axis speed, v_{yi} is the estimated y-axis speed, and $\dot{\theta}_i$ is the estimated turn rate. We extend these notations to the end state O_{k+1}.

The input data are encoded to a better representation through learning in the encoder part. The prediction is then processed in the decoder part, according to the previous predictions and the representation of the encoded input data. The previous predictions are provided through the decoder input $O_{0:k}$, which is a shifted view of the prediction $O_{1:k+1}$. Indeed, predicting the i-th prediction token will make the Transformer attend the i previous tokens $O_{0:i-1}$, with O_0 corresponding to a token indicating the start of the decoding process, such as the start token in NLP, and which all variables are equal to val_{start}. This enable using parallelism instead of sequentially computing each prediction tokens, which speeds up the learning process. This requires to prevent each prediction token to attend the previous ones, by putting their corresponding attention compatibility score to 0. However, this parallelism trick is applied during training only, otherwise autoregressive inference has to be performed.

The current states estimates $S_{1:k}$ are derived from $O_{1:k}$ following (1), with $S_i = [x_i, y_i, v_i, \theta_i, \dot{\theta}_i]$, $i \in [1..k]$, where v_i is the estimated speed norm, and θ_i is the estimated orientation, i.e. the object's heading. These state estimates $S_{1:k}$ are used to predict the future kinematics of objects.

$$\begin{cases} v_i = \sqrt{v_{xi}^2 + v_{yi}^2} \\ \theta_i = \arccos\left(\dfrac{v_{xi}}{v_i}\right) \operatorname{sgn}(v_{yi}) \end{cases} \tag{1}$$

The loss function used to train the model is set as

$$\mathcal{L} = \lambda_{pos}\mathcal{L}_{pos} + \lambda_{spd}\mathcal{L}_{spd} + \lambda_{\dot{\theta}}\mathcal{L}_{\dot{\theta}} + \lambda_{end}\mathcal{L}_{end} \tag{2}$$

where partial losses are defined for the ground-truth states $G_{1:k+1}$, with $G_i = [x_i^g, y_i^g, v_{xi}^g, v_{yi}^g, \dot{\theta}_i^g]$, $i \in [1..k]$, and $G_{k+1} = [val_{end}, val_{end}, val_{end}, val_{end}, val_{end}]$, as

$$\begin{cases} \mathcal{L}_{pos} = \dfrac{1}{k}\displaystyle\sum_{i=1}^{k} \sqrt{(x_i - x_i^g)^2 + (y_i - y_i^g)^2} \\ \mathcal{L}_{spd} = \dfrac{1}{k}\displaystyle\sum_{i=1}^{k} \sqrt{(v_{xi} - v_{xi}^g)^2 + (v_{yi} - v_{yi}^g)^2} \\ \mathcal{L}_{\dot{\theta}} = \dfrac{1}{k}\displaystyle\sum_{i=1}^{k} |\dot{\theta}_i - \dot{\theta}_i^g| \\ \mathcal{L}_{end} = \|O_{k+1} - G_{k+1}\|_2 \end{cases} \tag{3}$$

III. Experiments

A. Dataset

The training and validation datasets are constructed from simulations. They consist in generating measurements from a sensor that attempts to measure the characteristics of the objects present in specific spatial and temporal observation windows. The input data is composed of these measurements, while the ground truth, i.e. the Transformer expected output, is composed of the characteristics of the objects we try to predict. We distinguish 2 training datasets: the first one without clutter and the second one including clutter.

The following units, such as time, positions, speeds, turn rates, noises, and error metrics, are arbitrary units. Indeed, they can be considered as real values that have been rescaled for speeding up the learning process.

The training datasets are composed of n_{train} air situation instances, where an instance describes up to $n_{targets}$ objects. Each object has a constant speed norm and a constant turn rate. However, turn rate transitions are possible in a measurement time window, i.e. an instantaneous turn rate change drawn from the same initial turn rate distribution, in order to provide tracking capability for highly maneuvering targets and trajectory changes. Since disabling transition on the extreme parts of the trajectory improves the learning process, a transition can occur only at a time $t_{trans} \in [t_{trans,min}, t_{trans,max}]$. Initial positions are uniformly drawn in $[-pos_{max}, pos_{max}]$ on all axes, speeds in $[v_{min}, v_{max}]$, initial orientations in $[-\pi, \pi]$, and turn rates in $[-\dot{\theta}_{max}, \dot{\theta}_{max}]$. The value of pos_{max} is determined so that each object is located within a fixed limit range $range_{max}$ at all times, considering the maximum speed v_{max} and the maximum duration T of the time window.

Sensor measurements are generated by adding a zero mean Gaussian noise $\mathcal{N}(0, \sigma^2)$ to the actual positions of the objects at each fixed timestep Δt in a time window of duration T. It is assumed that there are no false detections for the first training dataset. In the second training dataset, at each timestep, clutter is generated by uniformly drawing and adding up to $n_{clutter}$ positions located in a spatial squared window of length $2\,dist_{clutter}$ centered on the object real position. In addition, additional clutter is uniformly drawn with up to $n_{clutter}$ positions located in the acceptable space, i.e. having the x-axis and y-axis positions in $[-range_{max}, range_{max}]$. This enables correlated and uncorrelated false detections, as well as increasing the robustness to various clutter locations. Concerning sequence generation and formatting, the start value, end value, and padding value are set relatively to $range_{max}$, respectively $val_{start} = -1.5\,range_{max}$, $val_{end} = 1.5\,range_{max}$ and $val_{pad} = 2\,range_{max}$.

In this work, $n_{train} = 320K$, $n_{targets} = 5$, $t_{trans,min}$ and $t_{trans,max}$ are respectively equal to 0.12 and 0.28, $v_{min} = 0.4$ and $v_{max} = 2.0$, $\dot{\theta}_{max}$ is taken equal to 7.85 which represents a half turn in 10 timesteps, with $\Delta t = 0.04$. The maximum duration of a time window is $T = 0.36$, $range_{max} = 4.0$, $pos_{max} = 3.28$, $val_{start} = -6.0$, $val_{end} = 6.0$, and $val_{pad} = 8.0$. Concerning noise and clutter parameters, $\sigma = 0.0133$, $n_{clutter} = 2$, and $dist_{clutter} = 0.2$.

B. Transformer Tracker

The Transformer-based state estimation method is described in section II. The parameters selected for the experiments are an embedded latent dimension $d_{model} = 64$, hidden dimension of FFN $d_{hidden} = 256$, FFN activation function $= GELU$, number of attention heads $n_{heads} = 4$, no dropout, number of encoder blocks $N = 4$, number of decoder blocks $M = 4$.

The selected training parameters for the Transformer-based method are a batch size of 512, with 50 epochs, a gradient clipping of 1.0, the Adam optimizer [16] with no weight decay and $(\beta_1, \beta_2) = (0.9, 0.99)$, a OneCycleLR scheduler [17] with sinusoidal annealing strategy: an initial learning rate of 5e-8, a final learning rate of 1e-6, a maximum learning rate of 8e-4,

TABLE I
ABSOLUTE ERRORS ON POSITIONS PREDICTED 1 TIMESTEP FORWARD USING KALMAN FILTER, EKF, AND THE TRANSFORMER TRACKER.

Tracking Method	Straight Line		Constant Turn	
	Mean	Median	Mean	Median
Kalman Filter	0.0180	0.0155	0.0386	0.0327
EKF	**0.0175**	**0.0148**	0.0243	0.0221
Transformer Tracker	0.0228	0.0206	**0.0226**	**0.0200**

and warm-up proportion of 0.2. The selected architecture has a total of 465,349 parameters.

The loss parameters in eq. 2 are: $\lambda_{pos} = 1.0$, $\lambda_{spd} = 0.5$, $\lambda_{\dot{\theta}} = 0.25$, $\lambda_{end} = 0.1$.

C. Results

The results presented in Table I are the mean and median absolute errors of the estimated position at one time step after the last measurement of an object making either a constant turn or going in a straight line. The estimations are made with the proposed method, a Kalman Filter with a constant velocity model, and an Extended Kalman Filter (EKF) with a constant turn model. Both Kalman Filter and EKF have their noise matrices chosen such as using the exact measurement noise level σ. First, this shows the Kalman Filter and the Transformer-based tracker are performing worse than the EKF in a straight line. In this case, the Kalman Filter kinematics model matches the real object kinematics, while the EKF and the Transformer Tracker are more prone to deviate from ground truth because of noise. Depending on the noise, the more probable trajectory given the measurements could be a constant turn rather than a straight line, which is allowed by the EKF and the Transformer Tracker kinematics models. However, the Kalman Filter and EKF are tuned for better overall performances on constant turn, which makes the EKF still better than the Kalman Filter in the straight line case. In addition, the Transformer Tracker is performing worse than the other filters, which may be explained by the initialisation of the Kalman Filter and the EKF, which is based on the ground truth, i.e. an ”oracle”. In practice, the ground truth is unknown but the results are chosen to use this initialisation scheme to compare the Transformer Tracker with strong baselines. Moreover, in a constant turn case, the Transformer Tracker is performing better than the EKF and the Kalman Filter. In this case, the non-linear kinematics of the object makes the accuracy of the Kalman Filter way worse than EKF, which has a non-linear kinematic model. Finally, in the case of a kinematics transition, such as a sharp change in the turn rate, the difference in accuracy between the Transformer Tracker and EKF is greater, demonstrating the better reactivity of our method in the event of kinematic change. This can be observed in Fig. 2, which illustrates an example of a trajectory of an object performing two successive constant-turn.

The mean absolute errors on the next timestep position seems to be independant on the position of the object, as shown in Fig. 3 and Table II. The differences in the errors are due to aleatoric uncertainty of the sampling, decreasing when the number of samples increases. Thus, our method shows consistent results when varying position variables.

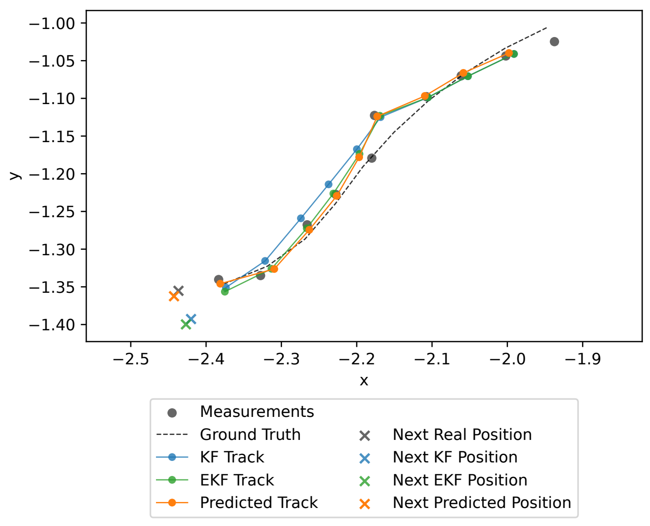


Fig. 2. Trajectory of a target performing two successive constant turn, with the Kalman filter, EKF, and the Transformer Tracker corresponding predictions. Gray dotted line and dots respectively represent the object trajectory and corresponding sensor measurements, colored dots represent the predicted positions at each time step, and colored x's represent positions predicted 1 timestep forward.

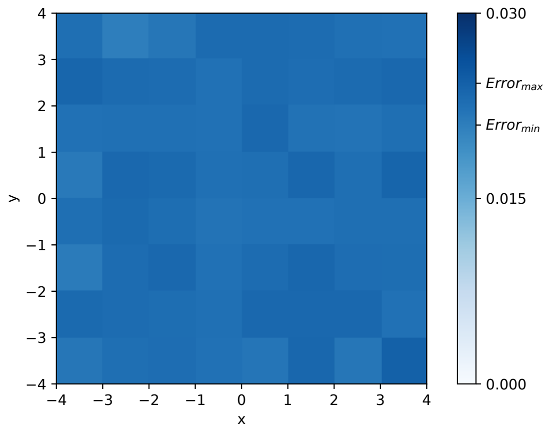


Fig. 3. Mean absolute errors on position predicted 1 timestep forward using the Transformer Tracker, depending on target position. $Error_{min}$ and $Error_{max}$ respectively represent the minimum and maximum error in the position grid.

This mean absolute errors on the next timestep position slightly increases when the speed of the object is increasing, as shown in Table III. It also seems to be consistent when the turn rate is changing, as shown in Table IV, although there is a slight increase for the highest turn rate category. The error associated with turn rates near zero is influenced by samples containing only 2 measurements, where the turn rate is set to zero to enhance the learning process. As demonstrated later, the mean error for samples with only 2 measurements is

TABLE II
MEAN ABSOLUTE ERRORS ON POSITION PREDICTED 1 TIMESTEP FORWARD USING THE TRANSFORMER TRACKER, DEPENDING ON TARGET RANGE r FROM WINDOW CENTER POSITION.

r	[0.0, 1.0]	[1.0, 2.0]	[2.0, 3.0]	[3.0, 4.0]	[4.0, 5.0]	[5.0, 6.0]
Error	0.0225	0.0230	0.0229	0.0228	0.0229	0.0222

TABLE III
MEAN ABSOLUTE ERRORS ON POSITION PREDICTED 1 TIMESTEP FORWARD USING THE TRANSFORMER TRACKER, DEPENDING ON TARGET SPEED v.

v	[0.4, 0.75]	[0.75, 1.0]	[1.0, 1.25]	[1.25, 1.5]	[1.5, 1.75]	[1.75, 2.0]
Error	0.0211	0.0222	0.0227	0.0234	0.0236	0.0241

TABLE IV
MEAN ABSOLUTE ERRORS ON POSITION PREDICTED 1 TIMESTEP FORWARD USING THE TRANSFORMER TRACKER, DEPENDING ON TARGET TURN RATE $\dot{\theta}$.

$\dot{\theta}$	[0.0, 1.3]	[1.3, 2.6]	[2.6, 3.9]	[3.9, 5.2]	[5.2, 6.5]	[6.5, 7.9]
Error	0.0264*	0.0216	0.0216	0.0214	0.0213	0.0222

* 0.0217 when discarding cases with only 2 measurements.

TABLE V
MEAN ABSOLUTE ERRORS ON POSITION PREDICTED 1 TIMESTEP FORWARD USING THE TRANSFORMER TRACKER, DEPENDING ON TIME t_{trans} WHERE TURN RATE TRANSITION OCCUR.

t_{trans}	no transition	[0.12, 0.16]	[0.16, 0.2]	[0.2, 0.24]	[0.24, 0.28]
Error	0.0228	0.0227	0.0228	0.0230	0.0231

higher than that for samples with more than 2 measurements. Consequently, these samples are mainly concentrated in the lower turn rate category, which overall increases the mean error. When discarding the samples with only 2 measurements, the mean error of the lowest turn rate category reduces to the error level of the other categories.

The impact of a turn rate transition on the mean absolute error of the following timestep position is presented in Table V. It shows that the error remains consistent whether or not there is a transition, and if there is a transition wherever it occurs. A marginal increase in the error can be observed when the transition occurs lately in the trajectory. In this scenario, the decreasing number of measurements available to estimate the new turn rate is reduced compared to the total measurements. Moreover, late transitions only appear in the case of trajectories with the highest number of measurements, and as seen in Fig. 5, also those with the best accuracy, which may then have an impact on the prediction accuracy.

As mentioned previously, Fig. 5 illustrates the mean absolute error of the following timestep position prediction of the Transformer Tracker, and it shows that increasing the number of measurements globally enhances the prediction accuracy. However, we still observe a slight decrease in accuracy for higher number of measurements, which is due to transitions that can occur more lately. In addition, the data generation process decreases the proportion of trajectories with no transition when increasing the number of measurements, further amplifying the phenomenon. The results of the proposed method are compared with the mean absolute error of a linear regression in the case of a zero turn rate. The linear regression expected mean error is described in (4). This shows the impact of the space dimension D of the problem, the noise level σ, the prediction time t_{new}, which increase the expected mean absolute error when increased, while increasing the number of measurements m decreases this error.

$$\sqrt{2}\,\frac{\Gamma\left(\dfrac{D+1}{2}\right)}{\Gamma\left(\dfrac{D}{2}\right)}\,\sigma\sqrt{\frac{1}{m}+\frac{\left(t_{new}-\bar{t}\right)^2}{\sum_{i=1}^{m}\left(t_i-\bar{t}\right)^2}} \tag{4}$$

where $D = 2$ is the space dimension, and t_{new} the time at which the prediction is made, here it is one timestep after the last measurement.

The linear regression error effectively decreases as the number of measurements increases, and it is globally lower than the Transformer Tracker error. This may be due to the more complex non-linear regression, with more degrees of freedom, that the Transformer Tracker has to perform in the case of a turn or during a transition. In addition, the error of the Transformer Tracker is lower than the linear regression for only 2 measurements. After further investigations, this is explained by the knowledge of our system, acquired during training, of the object's possible kinematics. In particular, this reduces regression errors due to highly noisy samples, which can be identified as highly unlikely, so that their processing can be adapted accordingly, improving overall accuracy. This difference fades as the number of measurements increases, i.e. as the effect of aleatoric uncertainty decreases.

The Transformer Tracker trained with up to 2 clutter measurements per timestep shows robustness and generalization capabilities, as shown in Fig. 6. An example trajectory is also illustrated in Fig. 4. The proposed method succeeds in scaling its accuracy relative to clutter, even up to 10 clutter measurements per timestep. In addition, it maintains its enhanced accuracy compared to Kalman Filter and EKF with the use of a Probabilistic Data Association Filter (PDAF) [18] in the presence of various levels of clutter. Finally, as already mentioned, the initialisation of Kalman Filter, EKF, and PDAF is based on the ground truth, which gives additional information that are unknown in practice. This highlights the performance of our method that still surpasses the Kalman Filter and EKF with PDAF.

The Transformer Tracker simultaneously performs data association and state estimation, making its inference time dependent only on the number of measurements and predicted states, resulting in quadratic complexity. This allows for parallelization and GPU acceleration, as shown in Table VI, where the inference times for single-object trajectories of the Transformer Tracker are comparable to those of EKF with PDAF. However, on a CPU, our method is generally slower

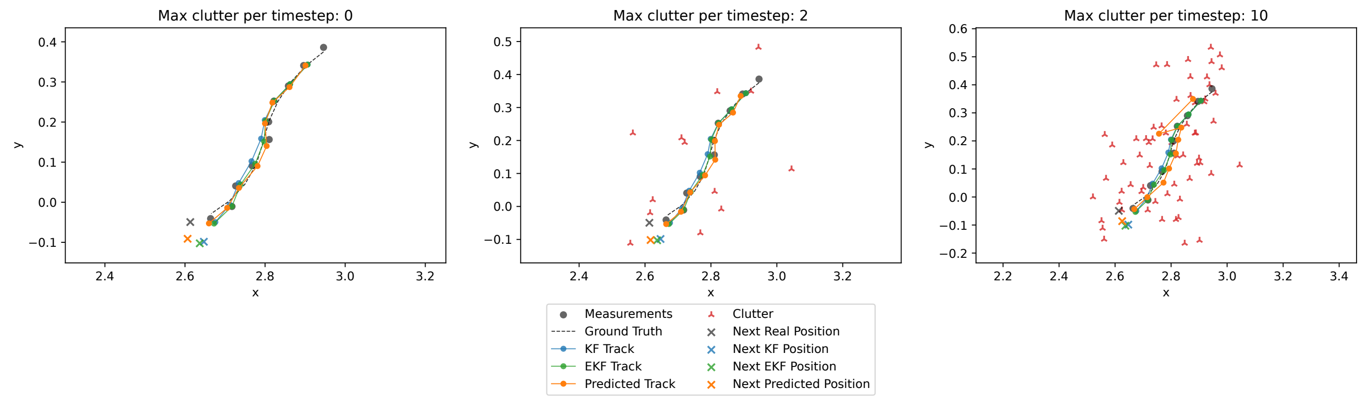


Fig. 4. Trajectory of a target performing two successive smooth turns, with the Kalman Filter with PDAF, EKF with PDAF, and the Transformer Tracker corresponding predictions. Colored dots represent the predicted positions at each time step, orange line and x corresponds to the Transformer Tracker predicted track and position predicted 1 timestep forward.

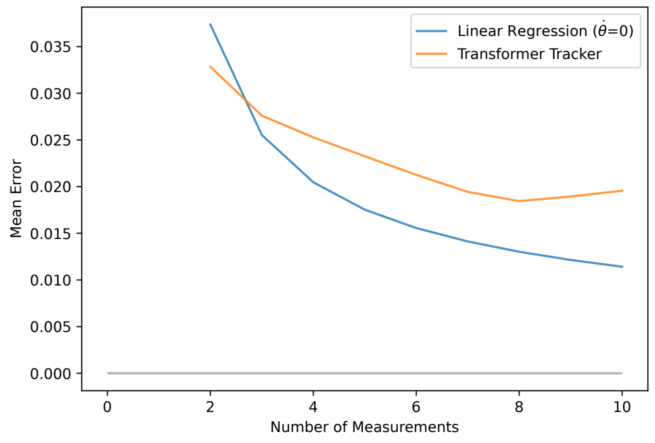


Fig. 5. Mean absolute errors on position predicted 1 timestep forward using the Transformer Tracker and a linear regression, depending on number of measurements. For the linear regression, only straight line trajectories ($\dot{\theta} = 0$) are considered.

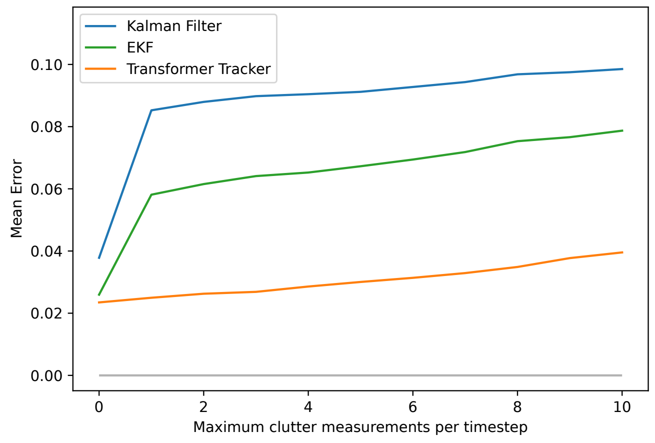


Fig. 6. Mean absolute errors on positions predicted 1 timestep forward using Kalman Filter with PDAF, EKF with PDAF, and the Transformer Tracker, depending on the maximum number of clutter measurements per timestep.

than EKF with PDAF, and becomes impractical with a large number of measurements. Additionally, when using EKF with PDAF, inference time increases with longer time windows due to more EKF updates and PDAF steps. While methods like Joint-PDAF offer finer associations, they suffer from factorial complexity, which our method overcomes with its quadratic scaling.

The Transformer-based tracker multi-target capability is shown in Fig. 7. Moreover, the generalisation on the number of objects is investigated in Fig. 8. More details about this behavior are given in the Appendix. It shows that the Transformer Tracker succeeds in predicting the correct number of objects on the training range, i.e. up to 5 objects, and the performance start decreasing after this threshold, to the point where there are almost no good predictions at 8 objects and above. This inability to generalize on the number of objects may be because the Transformer overfits on the maximum number of measurements which are proportional to the number of objects here, or/and overfits on the number of predicted states which correspond to the number of objects and the end token. This behavior matches the Transformers behavior for NLP applications [19].

TABLE VI
MEAN INFERENCE TIMES FOR SINGLE-TARGET TRAJECTORIES, DEPENDING ON NUMBER OF MEASUREMENTS, TRAJECTORY DURATION, AND DEVICE, USING THE EKF WITH PDAF AND THE TRANSFORMER TRACKER.

Tracking Method	Number of measurements			
	10	100	1,000	5,000
EKF with PDAF ($T = 0.36$)[1]	9.6 ms	12.0 ms	33.4 ms	121 ms
EKF with PDAF ($T = 3.96$)[1]	—	93.8 ms	118 ms	218 ms
Transformer Tracker[1]	12.2 ms	15.9 ms	230 ms	5710 ms
Transformer Tracker (GPU)[2]	16.3 ms	16.4 ms	21.4 ms	141 ms

[1] One core of an Intel i7-9850H CPU.
[2] Nvidia GeForce GTX 1080 8GB GPU.

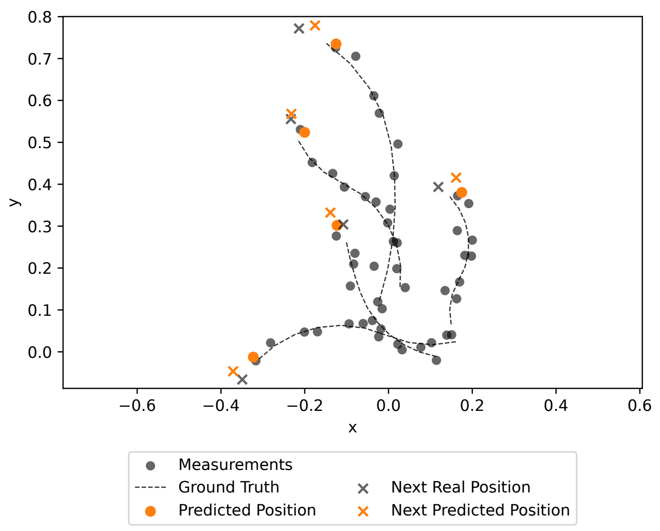


Fig. 7. Trajectory of 5 objects of various kinematics, with the Transformer Tracker corresponding predictions.

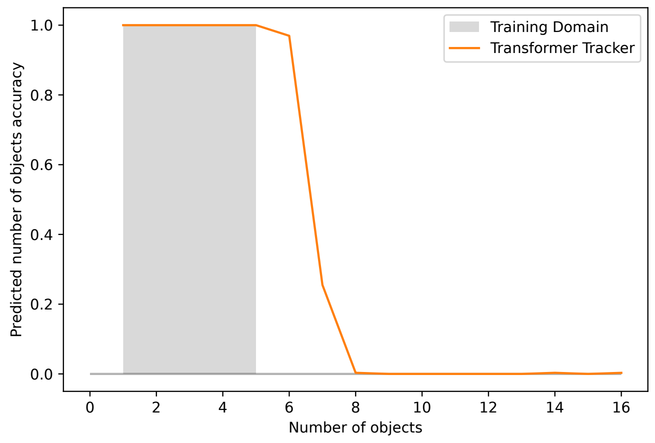


Fig. 8. Number of objects prediction accuracy of the Transformer Tracker, when it is trained with up to 5 objects in an air picture.

Finally, we investigate how the information is used by the Transformer Tracker to make the prediction. We use the attention rollout method [20] to output how the input tokens, i.e. the measurements, and the already estimated states are propagating the information to new state estimates, as shown in Fig. 9. The attention rollout propagates the attention weights to determine a unique attention map that represents how much the Transformer is paying attention to its encoder and decoder inputs, i.e. the measurements and previously predicted states. This is applied to an example trajectory, without clutter. For visibility purposes, the measurements are ordered in ascending time for each object, which are arranged in ascending order of position on the x-axis, to match the output order. Thus, the first measurements correspond to the first target, the next measurements to the second, and so on. For each object, i.e. its corresponding state, the Transformer Tracker seems to mainly

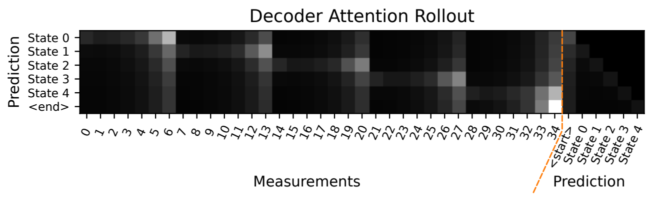


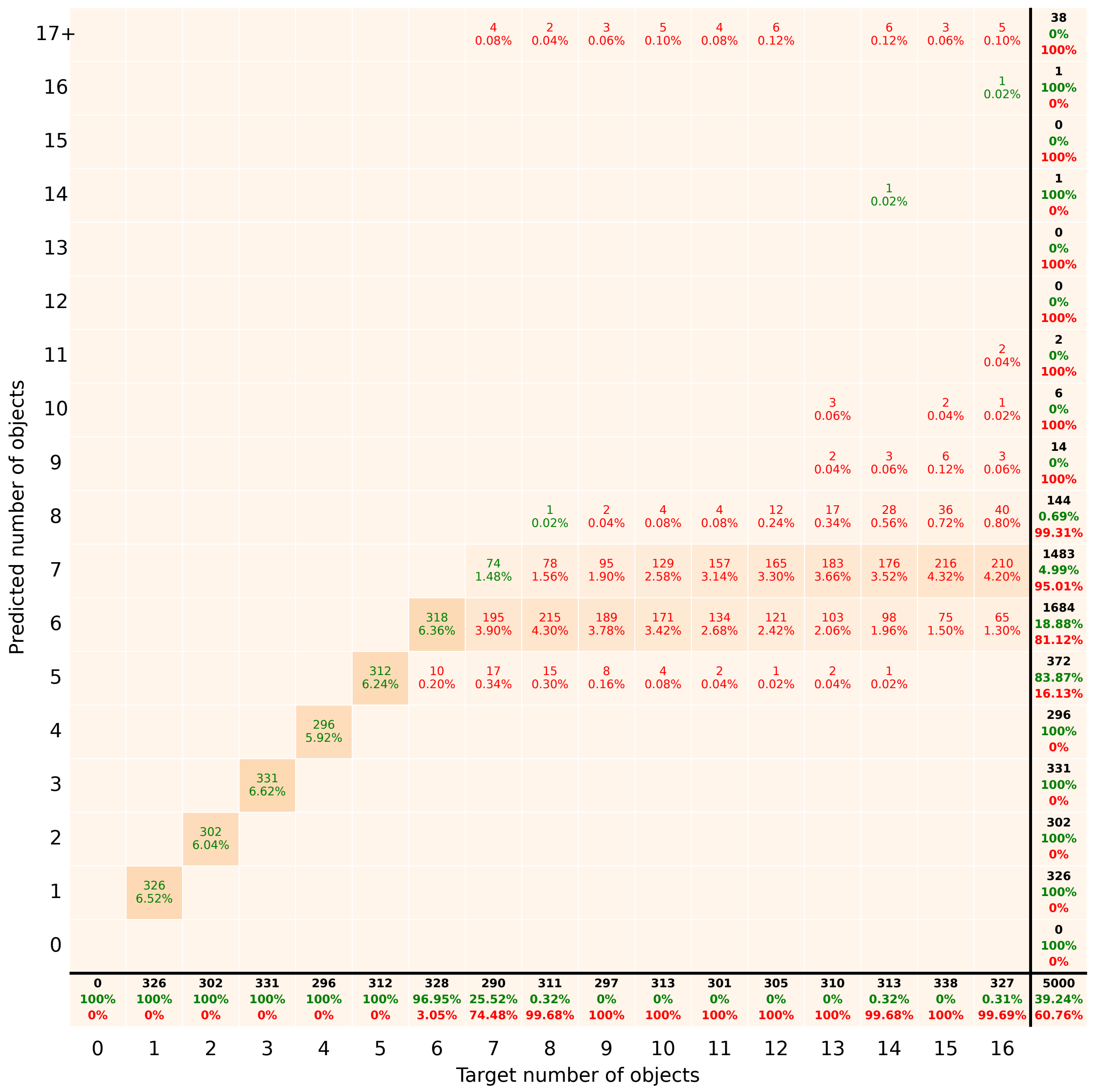
Fig. 9. Transformer Tracker decoder attention rollout for measurements and corresponding predictions of an air picture of duration $6\Delta t$ with 5 objects of various kinematics. Each row corresponds to a state estimation, where the intensity of its cells represents "how much" the measurements and the previous predictions are involved in the state estimation, i.e. how much the Transformer is paying attention to them during the prediction for this specific state estimation.

pay attention to measurements of this object, with a particular emphasis on the most recent ones.

IV. Conclusions and Future Work

In this work, we performed an exploratory analysis of the behavior of a Transformer applied to multi-target tracking under various kinematics parameters, clutter density, and number of objects. The evaluation was conducted in a simulated environment and aimed to assess the accuracy and consistency of the Transformer-based tracker method. The method was compared with the Kalman Filter and Extended Kalman Filter with adapted process and noise covariance matrices. It showed better state estimation in the general case, coming from better reactivity and knowledge building from learning data. This last behavior proves the Transformer Tracker yield better prediction capability than more general models. However, the results show a lack of generalization capability on the number of objects. Finally, the attention mechanism provides preliminary insight into the system's explainability. The reported findings provide valuable insights into the efficiency and adaptability of Transformer-based tracking methods in dynamic object motion estimation scenarios, performing object detection and state estimation simultaneously.

As a consequence, interesting future research directions include testing the Transformer-based tracking method for more realistic and complex target tracking scenarios. In particular, objects with kinematics of different order of magnitude such as speed or maneuverability could represent a great challenge for the Transformer Tracker. The lack of generalization on the number of objects raises valuable directions for research that can also be relevant to other fields, such as in natural language processing. Additional exploration could be conducted relating to multi-target tracking capability such as in crowded air situations, crossing trajectories, swarming, as well as using multiple sensors. Track management also needs to be investigated and tested for dynamic simulations, such as asynchronous track creation and deletion. Finally, the attention mechanism could be further exploited, in particular by transposing techniques used in natural language processing and computer vision for more convincing explainability and explicability results.


Predicted number of objects
Target number of objects

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
17+								4 0.08%	2 0.04%	3 0.06%	5 0.10%	4 0.08%	6 0.12%		6 0.12%	3 0.06%	5 0.10%	38 0% 100%
16																	1 0.02%	1 100% 0%
15																		0 0% 100%
14															1 0.02%			1 100% 0%
13																		0 0% 100%
12																		0 0% 100%
11																	2 0.04%	2 0% 100%
10														3 0.06%		2 0.04%	1 0.02%	6 0% 100%
9														2 0.04%	3 0.06%	6 0.12%	3 0.06%	14 0% 100%
8									1 0.02%	2 0.04%	4 0.08%	4 0.08%	12 0.24%	17 0.34%	28 0.56%	36 0.72%	40 0.80%	144 0.69% 99.31%
7								74 1.48%	78 1.56%	95 1.90%	129 2.58%	157 3.14%	165 3.30%	183 3.66%	176 3.52%	216 4.32%	210 4.20%	1483 4.99% 95.01%
6							318 6.36%	195 3.90%	215 4.30%	189 3.78%	171 3.42%	134 2.68%	121 2.42%	103 2.06%	98 1.96%	75 1.50%	65 1.30%	1684 18.88% 81.12%
5						312 6.24%	10 0.20%	17 0.34%	15 0.30%	8 0.16%	4 0.08%	2 0.04%	1 0.02%	2 0.04%	1 0.02%			372 83.87% 16.13%
4					296 5.92%													296 100% 0%
3				331 6.62%														331 100% 0%
2			302 6.04%															302 100% 0%
1		326 6.52%																326 100% 0%
0																		0 100% 0%
	0 100% 0%	326 100% 0%	302 100% 0%	331 100% 0%	296 100% 0%	312 100% 0%	328 96.95% 3.05%	290 25.52% 74.48%	311 0.32% 99.68%	297 0% 100%	313 0% 100%	301 0% 100%	305 0% 100%	310 0% 100%	313 0.32% 99.68%	338 0% 100%	327 0.31% 99.69%	5000 39.24% 60.76%

Fig. 10. Confusion matrix of the number of objects, for a Transformer Tracker trained with up to 5 objects in an air picture. In the confusion matrix cells, the first number is the number of samples for a given number of objects (column) and a given predicted number of objects (line), and the percentage is the proportion of the test samples they represent. These numbers are displayed in green when the prediction is correct and in red if not. In the last line, for each column representing a given number of objects, black numbers represent how many samples have the corresponding number of objects, green numbers represent the percentage of correct predicted number of objects, and red numbers represent the percentage of wrong predicted number of objects. In the last column, for each line representing a given predicted number of objects, black numbers represent how many samples have the corresponding predicted number of objects, green numbers represent the percentage of correct predicted number of objects, and red numbers represent the percentage of wrong predicted number of objects.

References

[1] P. Steininger, *Les Fondamentaux de la puissance aérienne moderne*. L'Harmattan, 2020.

[2] Y. Bar-Shalom and X.-R. Li, *Multitarget-Multisensor Tracking: Principles and Techniques*. Yaakov Bar-Shalom, 1995.

[3] L. Rakai, H. Song, S. Sun, W. Zhang, and Y. Yang, "Data association in multiple object tracking: A survey of recent techniques," *Expert Systems with Applications*, vol. 192, p. 116300, 2022.

[4] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering*, vol. 82, pp. 35–45, 03 1960.

[5] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, (Cambridge, MA, USA), p. 2692–2700, MIT Press, 2015.

[6] X. Bresson and T. Laurent, "The transformer network for the traveling salesman problem," 2021.

[7] J. Pinto, G. Hess, W. Ljungbergh, Y. Xia, L. Svensson, and H. Wymeersch, "Next generation multitarget trackers: Random finite set methods vs transformer-based deep learning," 2021.

[8] A. Milan, S. H. Rezatofighi, A. Dick, I. Reid, and K. Schindler, "Online multi-target tracking using recurrent neural networks," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, Feb. 2017.

[9] K. Yoon, D. Y. Kim, Y.-C. Yoon, and M. Jeon, "Data association for multi-object tracking via deep neural networks," *Sensors*, vol. 19, no. 3, 2019.

[10] H. Liu, H. Zhang, and C. Mertz, "Deepda: Lstm-based deep data association network for multi-targets tracking in clutter," in *2019 22th International Conference on Information Fusion (FUSION)*, pp. 1–8, 2019.

[11] Y. Yao, I. Smal, I. Grigoriev, A. Akhmanova, and E. Meijering, "Deep-learning method for data association in particle tracking," *Bioinformatics (Oxford, England)*, vol. 36, 07 2020.

[12] S. Jouaber, S. Bonnabel, S. Velasco-Forero, and M. Pilté, "Nnakf: A neural network adapted kalman filter for target tracking," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4075–4079, 2021.

[13] G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. G. van Sloun, and Y. C. Eldar, "Kalmannet: Neural network aided kalman filtering for partially known dynamics," *IEEE Transactions on Signal Processing*, vol. 70, pp. 1532–1547, 2022.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 6000–6010, Curran Associates Inc., 2017.

[15] V. Sonntag, J.-M. Le Caillec, A. Peres, and S. Devaud, "Transformer-based state estimation for tracking: Maneuvering target and multi- target capabilities," in *2024 IEEE Radar Conference (RadarConf24)*, pp. 1–6, 2024.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[17] L. N. Smith and N. Topin, "Super-convergence: very fast training of neural networks using large learning rates," in *Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications* (T. Pham, ed.), vol. 11006, p. 1100612, International Society for Optics and Photonics, SPIE, 2019.

[18] Y. Bar-Shalom and E. Tse, "Tracking in a cluttered environment with probabilistic data association," *Automatica*, vol. 11, no. 5, pp. 451–460, 1975.

[19] C. Anil, Y. Wu, A. Andreassen, A. Lewkowycz, V. Misra, V. Ramasesh, A. Slone, G. Gur-Ari, E. Dyer, and B. Neyshabur, "Exploring length generalization in large language models," in *Advances in Neural Information Processing Systems* (S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds.), vol. 35, pp. 38546–38556, Curran Associates, Inc., 2022.

[20] S. Abnar and W. Zuidema, "Quantifying attention flow in transformers," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, eds.), (Online), pp. 4190–4197, Association for Computational Linguistics, July 2020.