Multi-object tracking without dynamic models and hard association metrics

Christian Alexander Holz, Christian Bader, Matthias Drüppel

***Abstract*—In this paper, we develop Machine Learning (ML)-based methods for Multi Object Tracking (MOT) within the context of Advanced Driver Assistance Systems (ADAS). Given the increasing complexity and demand for precise and efficient object tracking systems in the automotive industry, this work focuses on the integration of ML techniques into established tracking methodologies. Key contributions encompass the creation and evaluation of three specialized neural networks: (i) the Prediction Network for predicting the trajectories of tracked objects, (ii) the Single Association Network (SANT) for associating incoming sensor objects with existing tracks sequential, (iii) and the Multi Association Network (MANTa) for associating multiple sensor objects with existing tracks within one timestep. We combine our ML methods with a traditional Kalman filter framework, offering a data driven approach to address MOT challenges while maintaining the modularity and interpretability of classical filter approaches. We asses both, the performance of all three models alone as well as their impact on the performance when they are integrated in the Kalman framework. Based on the KITTI tracking data set Car, the Root Mean Square Error (RMSE) for predictions could be reduced with our Prediction Network by half compared to a basic Kalman Filter. Replacing single components allows us to get a clear evaluation of the impact of each ML model on the overall tracking system while maintaining the modularity of the system. The results reveal a modular, robust, and maintainable tracker, underscoring the potential of ML integration in AD Tracking Systems.**

***Index Terms*—Article submission, IEEE, IEEEtran, journal, LaTeX, paper, template, typesetting.**

I. Introduction

THE ongoing evolution of Advanced Driver Assistance Systems (ADAS) has brought the need for precise and reliable Multi Object Tracking (MOT) into the spotlight [1], [2]. In complex and dynamic environments, as encountered in urban traffic, it is crucial to simultaneously and accurately capture the positions and movements of multiple objects - a key challenge in computer vision (CV).

In the commonly used Tracking-by-Detection (TbD) paradigm, a tracker fuses detected sensor objects (SO) to create consistent object tracks over time. A key challenge within this paradigm is associating the incoming measured SO with their corresponding existing object tracks or initializing new object tracks.

Tracking frameworks form the heart of ADAS systems that are used in millions of vehicles around the globe. The vast majority of these frameworks rely on classical approaches such as the Kalman filter (KF) or its variants. These classical tracking theories have the great benefit of being modular and interpretable. The task is split into clearly separated subtasks such as the prediction of currently tracked objects and the association with newly measured ones. Furthermore the math and the theory itself is often clean and comprehendible. However, in the automotive industry these tracking systems need to be applied to a variety of different car models with different sensor sets, different installation heights of the sensors, different countries and always perform for a wide range of driving scenarios. This often leads to the increasing implementation of heuristics and parameters that tweak the tracking system for specific configurations and situations. But hand-engineered parameters and heuristics are hard to maintain and develop further from a software engineering point of view. Furthermore, performance of classical approaches can reach its limits in challenging situations.

In this work, we propose a data driven approach to tracking frameworks, which would allow the same system to be fine tuned for specific configurations relying only on data, thus increasing maintainability and adaptability. We do this preserving one of the biggest strengths of classical approaches: its modularity, by replacing only single tracking components with ML models.

In comparison to the Kalman filter, our Prediction Network is capable of predicting the state of individual objects without the need for a predefined state or observation model at runtime. Our Prediction Network holds the potential, particularly in terms of adaptability to various scenarios and the ability to effectively handle nonlinearities. Many conventional tracking systems rely on static methods for data association, often based on simple heuristics or fixed thresholds. In contrast, Single Association Network (SANT) employs machine learning to automate these processes and adapt more effectively to different scenarios. As a result, within the Tracking-by-Detection (TbD) MOT framework, SANT replaces the calculation of a distance metric and the Hungarian algorithm for the corresponding assignment. Furthermore, we integrate the Prediction Network and SANT into an existing tracking system and demonstrate their performance through multiple tests and comparisons with established methods.

This work provides new insights and advancement in the development of Advanced Driver Assistance Systems (ADAS), contributing to the further evolution of technologies for autonomous driving (AD).

C. Holz is with Daimler Truck AG, Research and Advanced Development, Stuttgart, Germany

C. Bader is with Daimler Truck AG, Research and Advanced Development, Stuttgart, Germany

M. Drüppel is with the Center for Artificial Intelligence, Duale Hochschule Baden-Württemberg (DHBW), Stuttgart, Germany

II. Related work

A. Tracked object prediction

One fundamental problem in Tracking-by-Detection (TbD) frameworks is the prediction of the states of the already tracked objects. In many approaches Kalman filters and their variants have proven to be effective for state prediction [1]–[3]. However, they reach their limits in more complex scenarios, particularly in the presence of non-linear motion patterns and interactions among multiple objects. In this work, we introduce a novel MOT approach that leverages Machine Learning (ML) to overcome these challenges. We specifically focus on the development and implementation of Neural Networks (NN), which can enable more precise and flexible data-driven object tracking.

B. Association

Another fundamental problem for a TbD Tracker is the data association. For this some approaches only consider the current state of the tracks, while others integrate temporal information such as the track history. This aggregate of temporal information can be done for example using attention mechanisms as used in [4], [5] or recurrent neural networks (RNNs) as in [6], [7]. The latter is what we are also pursuing in this work. Similar to the problem statement in Mertz et al [7], the aim of our work is to develop a data-based approach that can learn to completely solve the combinatorial Non Deterministic Polynomial Time (NP) hard optimization problem of data association. Mertz et al [7] use a distance matrix based on the Euclidean distance measure as input for the developed association network, thus replacing an association algorithm such as the Hungarian Algorithm (HA) [8].

In the context of this work, we put forward the hypothesis that a Gated Recurrent Unit (GRU)-based association network can be designed and trained using an undefined distance measure. This can increase the assignment of the time-based memory units, i.e. the history, and thus boost performance. For this purpose, the Long Short-Term Memory network (LSTM) layer is implemented, which enables the memory of information using hidden units over a time interval [9]. The goal of this work was therefore to develop an association network that is intended to solve the assignment of one or more sensor objects (SO) to an existing number of object tracks without a defined distance measure.

C. Real time applications

ADAS are embedded real-time applications where it is crucial to predict the state of objects immediately after their detection. This rules out offline tracking methods as presented in [10] that process the entire video material at once in a batch process. Therefore, most recent approaches for tracking multiple objects rely on online methods that do not depend on future image information. Online methods use various features to estimate the similarity between the recognized objects and the existing tracks. This can be done on the basis of their predicted positions or even similarities in appearance.

a) Kalman Filter based Tracker: [1]–[3] propose a Kalman Filter (KF) based TbD multi-object tracker. [1] presents a MOT approach based on simple visual cues. The authors contend that many existing multi-object trackers are too complex and require a large amount of computational resources. Instead, they propose a simpler approach based on basic visual features such as colour, shape and motion. These visual cues are used to track objects at the image level and make associations between frames.

b) Recurrent Neural Network (RNN) based Tracker: How we also [6], [7], [11] present approaches for online multi-target tracking using recurrent neural networks (RNNs). The approach presented in [7] focusses on the task of data association in a TbD framework. The developed DeepDA approach represents an LSTM-based Deep Data Association Network to learn and perform the association of objects between frames. By learning association patterns from the data, the tracker can achieve robust and reliable tracking results even in highly disturbed environments. [7] use a distance matrix based on the Euclidean distance measure as input data for the developed DeepDA network, thus replacing an association algorithm such as the Hungarian Algorithm (HA). It can be assumed that the Euclidean distance measure was also used as the basis for generating the ground truth (GT) training data (distance matrices) and for the evaluation. However, this is not explicitly stated. It can therefore be argued that this preprocessing step deprives the network opportunity to follow a different association logic or to learn it data based.

c) Attention Mechanism based Tracker: Each of the papers [4], [5], [12]–[14] presents approaches for a tracker that utilise the attention mechanism [15], for example to compute soft data association [4]. The main research focus of the paper [4] is on soft data association, which enables the tracker to make probabilistic associations between objects and account for uncertainties in the associations. Soft data association in the SoDA model works by using attention mechanisms to aggregate information from all detections in a given temporal window. This allows the model to learn long-term and highly interactive relationships between detections and tracks from large datasets without using complex heuristics and hyperparameters.

III. Overview of our proposed models

Our primary contribution is the development and evaluation of three NN that we labeled: (i) the Prediction Network, (ii) the Single Association Network (SANT), and (iii) the Multi Association Network (MANTa). Figure 1 provides an overview of our approach in which the association network can be represented by (ii) or (iii) and the prediction network is represented by (i). The input for the proposed Prediction Network (i) are the sensor objects (SO) in every time step. These objects consists of a fixed-dimensional state vector that contains information such as object position, orientation and dimension. The Prediction Network predicts all vectors to the next time step where they are used as input to the Single Association Network (SANT) to build target trajectories or object tracks.

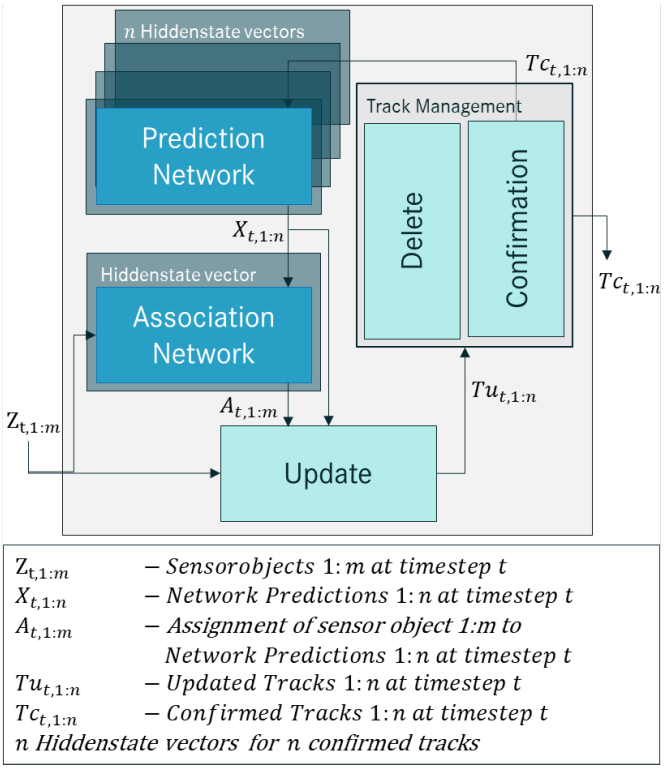

Fig. 1. Schematic representation of the two integrated networks in a tracking-by-detection (TbD) framework.

IV. Tracking with Prediction and Association Networks

We apply the tracking-by-detection (TbD) paradigm, in which a tracker fuses object fuses the object detections to generate object tracks that are consistent over time. [3], for example, provides a framework for analyzing tracking approaches that follow the TbD paradigm. This was used accordingly in this study.

To enable our models to incorporate temporal information, we use Long Short-Term Memory (LSTM) and bidirectional Long Short-Term Memory (BiLSTM) networks. Both for the prediction and the association (SANT) of the sensor objects (SO) to the existing tracks.

A. Prediction Network

Most existing tracking methods associate incoming detections pairwise with object states predicted by a simple motion model, e.g. a constant velocity model, using a Kalman filter [1]–[3]. However, recent work has demonstrated that aggregating temporal information as well as contextual information can improve the tracking of multiple objects by utilizing higher order information in addition to pairwise similarities between detections [4], [6], [7], [11]. Our approach is to use the hidden states of the LSTM layer as an object-specific information storage. The Prediction Network was designed for an open-loop application, this means that the network predicts values for a future time step based on previously received data. For use in an online MOT process, the network is therefore able to make a prediction about the most likely next state values using the state values received for a given Statevector of a Sensorobject (SO).

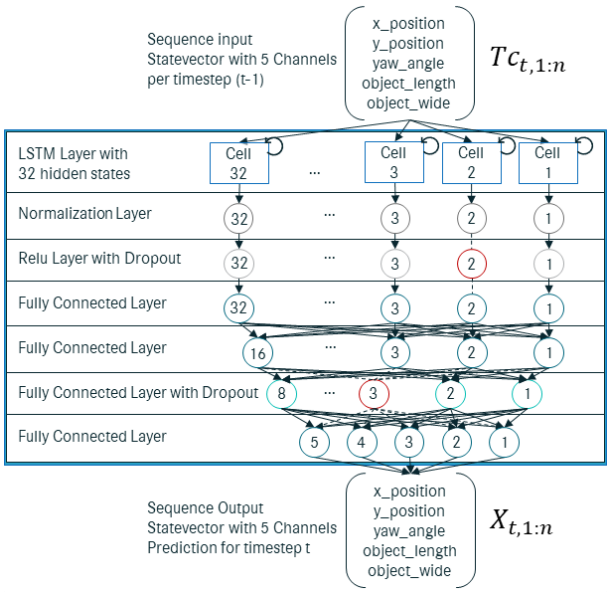

Fig. 2. Schematic representation of the generic prediction network structure.

a) Network architecture: The internal values (hidden states) of the LSTM layer are updated per time step based on the received measurement data. This updating process therefore provides a correction over the course of the sequence of each track. The number of hidden units (cells) corresponds to the amount of information that the layer remembers between time steps, as shown in Figure 2. The hidden states can contain information from all the previous time steps, regardless of the sequence length. The values of the hidden states of the LSTM layer are updated in each time step based on the received measurement data and the hidden states of the past timestep. These updates therefore results in an internal correction over the course of the sequence. In our prediction network, we have placed a batch normalisation layer directly after the LSTM layer. This means that the output of the LSTM layer is normalised before being passed on to the subsequent Relu layer. This helped speed up training and improve convergence by reducing internal covariate shifts [16]. A Relu layer performs a threshold operation to each element of the input, where any value less than zero is set to zero. At training time, a dropout layer randomly sets input elements to zero with a given probability. This operation effectively changes the underlying network architecture between iterations and helps prevent the network from overfitting [17]. The Fully Connected (FC) Layers multiplies the input by a weight matrix and then adds a bias vector. As shown in Figure 2 all neurons in a FC layer connect to all the neurons in the previous layer. This layer combines all of the local information learned by the previous layers. The last FC layer must be equal to the number of response variables in the Regression Output layer [18]. The regression layer computes the half-mean-squared-error loss. On the output of the regression layer at the end of the network (”regressionoutput”), the network calculates the loss through the mean square error (MSE) for the prediction for each state

value. The MSE indicates the average of the squared difference between the model prediction and the target value, which we use as the measure of the quality of the prediction. For a single observation, the MSE is given by:

$$MSE = \sum_{i=1}^{R} \frac{(gt_i - y_i)^2}{R} \tag{1}$$

where R is the size of the predicted state vector, gt_i is the ground truth value (Kitti car tracks) and y_i is the prediction of the network for sample i. For our sequence-to-sequence regression network the loss function of the regression layer is half the mean square error of the predictions for each time step, normalized by the sequence length S:

$$loss = \frac{1}{2S} \sum_{i=1}^{S} \sum_{j=1}^{R} (gt_{ij} - y_{ij})^2. \tag{2}$$

During training, the average loss is calculated using the observations in the mini-batch, so S equals the mini-batch length. In relation to the KITTI data set (cars and vans), a Root Mean Square Error (RMSE) of 0.025 was achieved for the position prediction of all objects in the dataset. Using a standard Kalman filter (KF), an RMSE of 0.066 was achieved on the same data set.

b) Data preprocessing: Ground truth (GT) data in the form of tracks of vehicles from the KITTI data set was used for our development. A GT Track contains the information of an object over time, starting with its appearance and ending with its exit from the sensor detection range. In order to achieve better generalization and to increase the chance that the training converges, the track state values at time t (predictors) and track state values at time t+1 (targets) were normalized following [19], such that the possible predictions and targets have a mean value of zero and a unit variance. The mean value μ and the standard deviation σ for each state variable were calculated for all tracks using the following equations:

$$\mu = \frac{1}{N} \sum_{i=1}^{N} A_i \quad \text{and} \quad \sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^{N} |A_i - \mu|^2} \tag{3}$$

where A_i the combined length of all tracks and N is the number of states. We also apply pre-padding as described in [20] where Reddy et al. show how padding influences the performance of neural networks in sequence-based tasks. Their study suggests that although both pre-padding and post-padding are feasible, the choice of padding technique can have a significant impact on the efficiency of the model, especially for LSTM networks where the sequence context is crucial. As [20] described, padding can lead to noise in the network, but one sequence adjustment per mini-batch is required for training LSTM networks. To minimise the effect, the sequences of the training data were therefore sorted by length before the mini-batch padding. Figure 3 clearly shows that the padding (turquoise area) is minimised by a sorted training data set per mini-batch. Without this modification, no convergence could be achieved in the training.

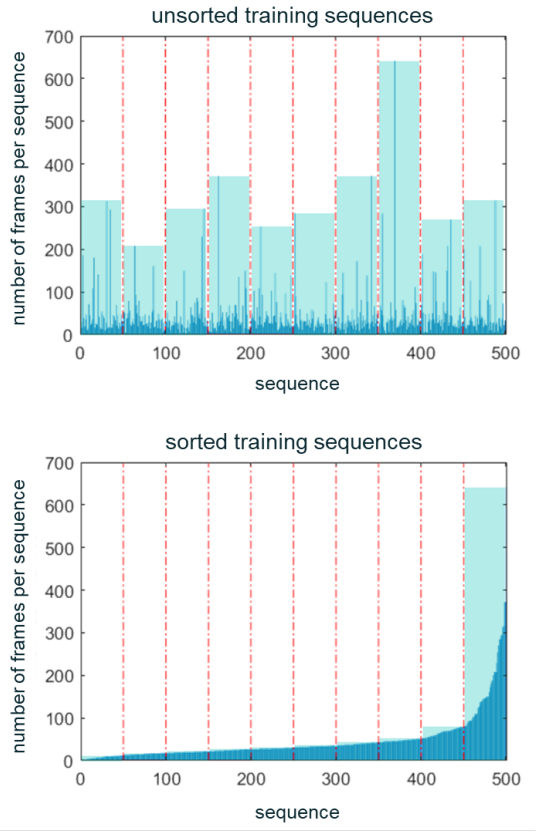

Fig. 3. Mini-batch padding effect, training data for LSTM networks.

B. Association network for a single sensor object

The association network enables a data-based approach to solve the combinatorial Non Deterministic Polynomial Time (NP) hard optimisation problem of data association. Compared to most association methods [6], [7], the single association network (SANT) is an approach that does not receive a defined distance matrix as input data, but instead receives the existing tracks and each new measured sensor object (SO).

This eliminates the need to define a distance measure, giving the network the freedom to follow its own association logic and learn it based on the training data. Therefore, the association network replaces the calculation of a distance metric and an association algorithm such as the Hungarian Algorithm (HA) [8].

a) Data preprocessing: In order to develop the single association network (SANT), the data association was considered as a time-structured problem (sequence-to-vector). The data association problem is therefore the assignment of exactly one sensor object (SO) $Z_{(t,1)}$ to a set of tracks $X_{(t,1:n)}$. The corresponding tracks were extracted from the KITTI data set of the labelled camera recordings. However, no real GT data exist for the association problem under consideration. To ensure that the assignment of a sensor object (SO) to a track of a track set has exactly one correct solution, the SO was generated in a time step from the existing track set of a time step. The data was noised artificially in order to generate realistic sensor data from the GT data. All existing GT tracks were noised one by

one for each time step. To achieve this, a maximum of 3% of the value of the current state vector was randomly subtracted or added to each value. The data set was created in 7 iterations, so that the noise intensity was increased by 0.5% per iteration.

As shown in figure 4 the data format was created accordingly to enable index-based track assignment for the Single Association Network (SANT). The size of the input matrix therefore corresponds to $m * n + 1$. With $m = 5$ (number of state values in this investigation) and $n(t)$ (current number of tracks per time step). The number of tracks currently existing in the time step can vary between 0 and a maximum of 16 objects in relation to the data set (KITTI) and the selected objects (cars and vans).

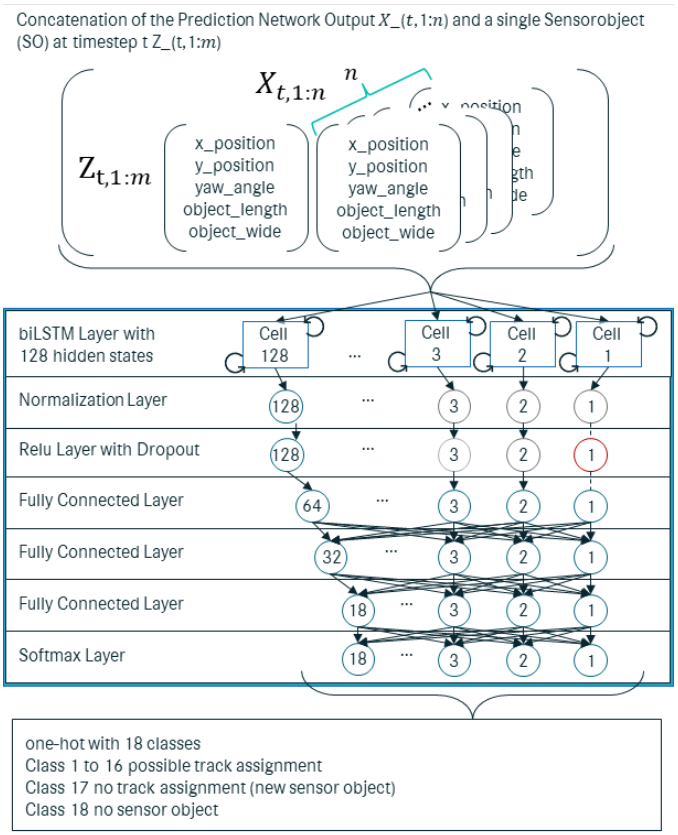

Fig. 4. Schematic representation of the generic network structure of SANT.

b) Network architecture: The network (fig. 4) is designed as a sequence-to-classification network, using the sequenceinput to represent the described input matrix per time step.

By combining the outputs of two LSTM layers that pass the information in opposite directions, [9] demonstrates the ability to capture the context from both ends of the sequence. The resulting architecture is called Bidirectional / Bidirectional Long Short-Term Memory Network (biLSTM). As part of these investigations, the best association performance was achieved with the biLSTM layer. Architectures with RNN, GNU, LSTM and biLSTM layers were tested. The network shown in Figure 4 achieved the best performance in comparison with a Training and Validation Accuracy of 95%. The output mode 'last' has been configured in the biLSTM layer. This layer is therefore able to receive a sequence as input and output a single value or value vector. This form of dimension reduction is necessary in order to carry out a corresponding classification.

The last fully connected layer (FC) specifies the number of classes via the number of output values. The classes are calculated in the Softmax layer by applying the Softmax function into a unique probability distribution. The softmax function converts a number of values z_i into a probability vector with i values. A high numerical value leads to a high probability in the resulting output vector (basis for one-hot encoding). The outstanding feature of this function is that the sum of the output values (probability values) is always less than or equal to 1:

$$g(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_i}} \tag{4}$$

By using the cross-entropy operation in the output layer, SANT could be trained to a correct assignment value (GT class index). The cross-entropy cost function (cross-entropy) calculates the cross-entropy loss between network predictions and target values for the unique assignment task (for mutually exclusive classes): One-hot coding is used to represent the class in binary form in a vector and thus generate a 1-to-n code. The following formula is used to calculate the cross-entropy loss values for each input value Y_j and associated target value T_j element by element:

$$loss_j = -(T_j ln Y_j + (1 - T_j) ln(1 - Y_j)) \tag{5}$$

To obtain a scalar $loss$, all loss values $loss_j$ are totalled and divided by the number of samples N. Optionally, the loss values of defined samples can be weighted with a weighting factor w_j:

$$loss = \frac{1}{N} \sum loss_j w_j \tag{6}$$

Eine entsprechende Nutzung des Gewichtungsfaktors w_j kann bei Datensätzen mit unausgewogenen (imbalanced) Klassenverteilung hilfreich sein.

C. Multi Association Network (MANTa)

The developed single association network (SANT) shows that a data-based association logic can be learnt from a deep learning model. The aim was also to develop a network to recognise a number of m sensor objects SO to an existing number of n tracks in one operation step. in one operation step. A multi association network (MANTa) was developed, which is able to solve the following association problems:

- **1 to n** - one SO to n tracks
- **m to 1** - m SO to one track
- **m to n** - m SO to n tracks
- **m to 0** - m SO to no tracks
- **0 to n** - no SO to n tracks
- **0 to 0** - no SO to no tracks

With the integration of a multi association network (MANTa) into a Multi Object Tracking (MOT) framework, the question can be asked whether a 0 to n and 0 to 0 assignment is a task to be solved. If no new SO is detected, the prediction of the

last operation step can be continued until the track is deleted on the basis of the decreasing probability of existence within the track management module. The association algorithm or the MANTa does not need to be called if no tracks and sensor objects are detected. Although these assignment options can therefore be resolved via the programme structure in the MOT framework, these options are also taken into account. This is intended to ensure that the network also learns to deal with SOs and tracks that are no longer available.

a) Datenvorverarbeitung: Der Datensatz für das Training und die Validierung von MANTa wurde entsprechend der beschriebenen Zielsetzung erstellt. Abbildung 5 zeigt die Inputdatenstruktur mit entsprechenden Zuordnungsaufgaben im dargestellten Zeitschritt (85) der Sequenz 20 des KITTI Datensatzes. Über alle Sequenzen wurden die jeweiligen Tracks pro Zeitschritt extrahiert. Die extrahierten Tracks wurden wie bei der SANT Entwicklung jeweils mit einem Rauschen modifiziert und im Anschluss normiert. Die so erhaltenen Werte wurden wie in Abb. 5 dargestellt als Messungen in pseudo-zufälliger Reihenfolge pro Zeitschritt einer $F * T_{max}$ Matrix zugeordnet. Wobei F die Anzahl der Feature darstellt (hier 10). Die Featureanzahl ergibt sich aus den Zustandswerten pro Track und SO Set. T_{max} steht für die maximale Anzahl an existierenden Tracks pro Zeitschritt. Für den definierten Untersuchungsfall mit Cars und Vans ergibt sich aus dem KITTI Datensatz ein $T_{max} = 16$. Im dargestellten Zeitschritt der Sequenz existieren 7 Tracks. Jeder Track erhält in diesem Zeitschritt eine neue Messung, zusätzlich wurde ein weiteres Objekt entdeckt (neues sensor object). Die GT Zuordnung ist im unteren Teil des Ausschnitts ausgegeben. Die Werte zeigen den korrekten Index, der Zuordnung für 1 bis 16 (T_{max}) bezogen auf das existierende Trackset. Der Wert 99 wurde als Klassenindex für neue bzw. nicht zuzuordnende Messungen definiert. Entsprechend ist die Messung in der zweiten Spalte ein neues Objekt und sollte keinem bestehenden Track zugeordnet werden.

Fig. 5. MANTa, data structure, input / output

b) Netzwerkentwicklung:

V. Experimental Evaluation

... KITTI-Car Benchmark.

VI. Conclusion

The conclusion goes here.

Acknowledgments

This should be a simple paragraph before the References to thank those individuals and institutions who have supported your work on this article.

Appendix
Proof of the Zonklar Equations

Proof of the First Zonklar Equation

Appendix goes here.

Proof of the Second Zonklar Equation

And here.

References

[1] Jenny Seidenschwarz, Guillem Brasó, Victor Serrano, Ismail Elezi, Laura Leal-Taixé, "Simple cues lead to a strong multi-object tracker," *Paper*, 2022.
[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, Ben Upcroft, "Simple online and realtime tracking," *Paper*, 2016.
[3] Ba-Tuong Vo, "code set for research use: Multi-sensor multi-target tracking," 2013. [Online]. Available: https://ba-tuong.vo-au.com/codes.html
[4] W.-C. H. H. K. T.-Y. L. Y. C. R. Yu, "Soda: Multi-object tracking with soft data association," *Paper*, 2020.
[5] Q. C. W. O. H. L. X. W. B. L. N. Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *Paper*, 2017.
[6] Anton Milan, Seyed Rezatofighi, Anthony Dick, Ian Reid, Konrad Schindler, "Online multi-target tracking using recurrent neural networks," *Paper*, 2016.
[7] H. L. H. Z. C. Mertz, "Deepda: Lstm-based deep data association network for multi-targets tracking in clutter," *Paper*, 2019.
[8] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, 1955.
[9] S. H. J. Schmidhuber, "Long short-term memory," *Neural Computation*, 1997.
[10] Roberto Henschel, Laura Leal-Taixé, Daniel Cremers, Bodo Rosenhahn, "Fusion of head and full-body detectors for multi-object tracking," *Paper*, 2017.
[11] Johannes Fitz, "Datenassoziation für multi-objekt-verfolgung mittels deep learning," *Paper*, 2020.
[12] Qi Chu, Wanli Ouyang, Hongsheng Li, Xiaogang Wang, Bin Liu, Nenghai Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *Paper*, 2017.

[13] Q. C. W. O. H. L. X. W. B. L. N. Yu, "Online multi-object tracking using cnn-based single object tracker with spatial-temporal attention mechanism," *ICCV*, 2017.
[14] T. M. A. K. L. L.-T. C. Feichtenhofer, "Trackformer: Multi-object tracking with transformers," *Paper*, 2022.
[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *NIPS*, 2017.
[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *ICML*, 2015.
[17] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, 2014.
[18] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," *AISTATS*, 2010.
[19] I. G. Y. B. A. Courville, *Deep Learning*. MIT Press, 2019.
[20] D. M. R. N. V. S. Reddy, "Effect of padding on lstms and cnns," *Paper*, 2019.