Track Classification for Random Finite Set Based Multi-Sensor Multi-Object Tracking

Alexander Scheible, Thomas Griebel, Martin Herrmann, Charlotte Hermann, and Michael Buchholz

This paper has been accepted for presentation and publication at the 2023 IEEE Symposium Sensor Data Fusion and International Conference on Multisensor Fusion and Integration (SDF-MFI), 27-29 November 2023, Bonn Germany. This is the accepted version of the paper, which has not been fully edited and the layout may differ from the original publication.

Citation information of the original publication:

A. Scheible, T. Griebel, M. Herrmann, C. Hermann and M. Buchholz, ”Track Classification for Random Finite Set Based Multi-Sensor Multi-Object Tracking,” 2023 IEEE Symposium Sensor Data Fusion and International Conference on Multisensor Fusion and Integration (SDF-MFI), Bonn, Germany, 2023, pp. 1-8, doi: 10.1109/SDF-MFI59545.2023.10361438.

Track Classification for Random Finite Set Based Multi-Sensor Multi-Object Tracking

Alexander Scheible, Thomas Griebel, Martin Herrmann, Charlotte Hermann, and Michael Buchholz
Institute of Measurement, Control and Microtechnology
Ulm University
89081 Ulm, Germany
{alexander.scheible, thomas.griebel, martin.herrmann, charlotte.hermann, michael.buchholz}@uni-ulm.de

***Abstract*—The state-of-the-art of random finite set (RFS) based approaches for multi-sensor multi-object setups solve the classification and track estimation jointly in a Bayesian style. This is computationally demanding and often requires additional modeling and parameter estimation. Additionally, these approaches are not designed to make use of direct class estimations, e.g., from machine learning detectors, but estimate the class based only on the kinematic features. This work applies a separated track classification, which uses direct class estimations, to RFS-based trackers. The proposed approach can be implemented for various RFS-based multi-sensor multi-object tracking algorithms without altering their structure and without additional modeling effort. For the update of the class estimation of a track, three different methods are presented. The three approaches are demonstrated and evaluated in combination with a labeled multi-Bernoulli filter on simulated and real-world data.**

I. Introduction

In multi-object tracking (MOT) setups, the additional information on the objects' classes can be used to enhance the tracking performance, e.g., in the association step [1]. Further, e.g. in the context of automated driving, subsequent modules in the processing chain require knowledge on the classes of the objects [2].

There are several ways to integrate classification methods into tracking, which can roughly be grouped into two categories: implicit and explicit methods. The first category uses solely the same kinematic features that are used for the tracking itself and thus classifies the objects implicitly from the available tracking input. The explicit methods require additional information in form of measurements or estimations on the classes of the objects as inputs.

For the implicit methods, the problem can be formulated in a fully Bayesian style. Here, the joint density of the kinematic state and of the class is estimated, which theoretically allows the modeling of arbitrary dependencies between the kinematic state and the object class. This can improve both, tracking and classification results [1, 3]. These methods can, e.g., exploit class-dependent state models with multiple-model filters [4, 5, 6, 7], or class-dependent measurement models [8]. An advantage of this approach is the closed and rigorous mathematical modeling. However, this also implies that a characteristic model for each class must be developed such that the classes can be separated only based on these class-dependent models, which can be a demanding task that is highly application specific. A good example of that is [8], which proposes a class-dependent measurement model for wideband radar observations based on a 3D scattering center model for maritime ship surveillance. The method is capable of distinguishing three classes of ships, but requires a 3D model for each class and is therefore difficult to extend.

For the implicit methods relying on class-dependent state models, it is difficult to separate classes with similar kinematic behavior. This is a severe drawback, especially in the context of automated driving, where it is important for subsequent modules to distinguish also categories that behave kinematically similar, such as a bus and a truck. Additionally, it is not straightforward to apply the approaches to different tracking algorithms, since the internal structure of the algorithm needs to be changed.

Moreover, using these implicit concepts within tracking methods makes the trackers usually computationally demanding as they often depend on sampling methods [9]. The authors of [9] tackle this computational burden by assuming that only the marginals of the objects' states and classes are of interest. This leads to the assumption that the measurements to infer the kinematics and to infer the classes are independent, which then allows to build a model upon feature-to-class and feature-to-measurement models. However, this approach also needs highly application-specific knowledge because explicit relationships between measurements as well as states on the one hand and classes on the other hand need to be modeled. In the end, the problem of increasing the number of classes still prevails.

For the explicit methods, the required class information input can come, e.g., from a machine learning detector, which not only detects the kinematic features of the objects for the actual tracking task, but also performs a class estimation for the detected objects [10, 11, 12]. In [13], the class estimation

Parts of this research have been conducted as part of the PoDIUM project and other parts as part of the EVENTS project, which both are funded by the European Union under grant agreement No. 101069547 and No. 101069614 respectively. Views and opinions expressed are however those of the authors only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the granting authority can be held responsible for them.

Parts of this work have been financially supported by the Federal Ministry of Education and Research (project AUTOtech.*agil*, FKZ 01IS22088W) and other parts by the Federal Ministry for Economic Affairs and Climate Action of Germany within the program "Highly and Fully Automated Driving in Demanding Driving Situations" (project LUKAS, grant number 19A20004F).

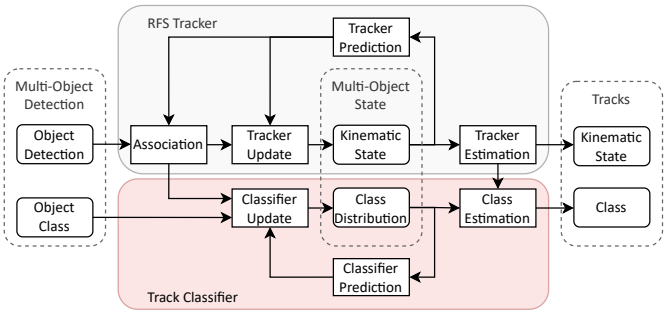

Fig. 1. The proposed track classification framework for RFS-based trackers. It uses the associations of the tracker to update the current class distribution with the object class of the detector.

of a detector is used to enhance the association step of a random matrix interacting multiple-model filter. While this is computationally substantially cheaper than the implicit approaches, it lacks the mathematically rigorous formulation of recent state-of-the-art multi-sensor MOTs that use the random finite set (RFS) framework [14]. RFS and finite set statistics (FISST) is a theory that unifies much information fusion in a single Bayesian frame [15]. Also, due to the feedback from the classifier back into the tracker, the tracker can loose its desired mathematical properties.

Therefore, we propose to strictly separate the estimation of the kinematic state and that of the class. This yields a computationally cheap classification and, since we will not have feedback from the classification into the tracker, retains the desired properties of the tracker. Additionally, as the detector usually has access to much more information on an object than it provides to the tracker, it is easier to add and distinguish more classes. Furthermore, the method will automatically benefit from the increasing performance of machine learning detectors [16].

To the best of our knowledge, this paper is the first to propose such a separate classification for RFS-based trackers. Figure 1 shows the general architecture of our method. It assumes that each detected object is annotated with a class, which ,e.g., can either be estimated by the detector directly, or by an external classifier. The object detections are then, as usual, processed by the tracker. The association of the RFS tracker between detections and tracks is reused in the classification step. There, the object class estimation of the detector is used to update the current class distribution of the internal multi-object state of the tracker. The key characteristic of our method is that it allows track classification without altering the structure of an existing tracker. In principle, it is also possible to use the kinematic information of the tracker for object classification. This, however, is not investigated in this work.

For the update of the class distribution, we present three different approaches. The first one assumes conditional independence of the object class estimations of the detector, resulting in the Bayes parallel combination rule. The other two are based on subjective logic (SL) [17], which is a framework that, among other things, explicitly models the statistical uncertainty. We will show that all three approaches benefit from their low computational complexity compared to approaches of the first category and enhance the classification performance compared to the native, single-frame, performance of the detector. The main contributions of the paper are

- a novel framework that uses direct class estimations in combination with an RFS tracker for track classification and
- a novel SL operator for the fusion of different class estimations.

After a summary of the mathematical and algorithmic background in Section II, our method will be elaborated in detail in Section III. There, we first describe the general framework and then present the three variants for updating the track class estimates. We evaluate the method and discuss all three class update variants for the labeled multi-Bernoulli (LMB) filter [18] on simulated and real-world data from an automated vehicle in Section IV. Finally, our paper closes with some conclusions and an outlook in Section V.

II. Background

This section summarizes the basics from literature required for our method and the overall framework, as illustrated in Fig. 1, like possible detectors and the RFS tracker.

A. Detector

As shown in Fig. 1, the detector delivers the required input for the tracker to update the kinematic state of the tracks. In our method, the only additional requirement on the detector is that each detection also comes with a class estimate. In general, this class estimate can either be directly produced by the detector itself, or by an additional object classifier. However, since most detectors, like [10], [12], [11], [19] and many others, jointly perform detection and classification, we simplified our figure to match this case. We interpret the classifier output as the posterior probability $P(c|d)$ of class c based on the raw data d processed by the detector.

B. Multi-Object Tracking

We consider multi-sensor MOT algorithms based on FISST theory [15]. These algorithms estimate the number and states of the objects simultaneously in a Bayesian manner, i.e., a separate prediction and update step (ignoring joint prediction and update schemes for the sake of simplicity). For this, the former step predicts the current multi-object state density over time, which is then corrected using sensor measurements [15], cf. Fig. 1. The key element for this is the definition of an RFS $X \in \mathcal{F}(\mathbb{X})$, where $\mathcal{F}(\mathbb{X})$ is the set of all finite subsets of a space $\mathbb{X}$. There are various trackers using this concept, notably the generalized labeled multi-Bernoulli (GLMB) filter, which is the first analytically closed MOT filter for labeled RFS [20], the labeled multi-Bernoulli (LMB) filter [18], which is a real-time capable approximation of the GLMB filter, and the Poisson multi-Bernoulli mixture (PMBM) filter [21] for unlabeled RFS.

The key property that we use in our classification approach is that each track in the above-mentioned filters has a unique

measurement-to-track association in the update step. In this paper, we demonstrate our classification approach on the LMB filter, which is shortly introduced in the following. Here, $\boldsymbol{\pi}$ denotes the probability measure for labeled RFS, r and p are the existence probability and the spatial distribution of a Bernoulli RFS, respectively, and $\ell \in \mathbb{L}$ denotes a label. Values with a label are indicated by bold face, and the subscript $+$ denotes predicted values [15]. If the predicted density is an LMB with parameters $\boldsymbol{\pi} = \{r_+^{(l)}, p_+^{(l)}\}_{l \in \mathbb{L}_+}$, then the updated density is a GLMB with [18]

$$\boldsymbol{\pi}(\boldsymbol{X}|Z) = \Delta(\boldsymbol{X}) \sum_{(I,\theta) \in \mathcal{F}(\mathbb{L}_+) \times \Theta} w^{(I,\theta)}(Z)\, \delta_I(\mathcal{L}(\boldsymbol{X})) \left[p^{\theta}(\cdot|Z)\right]^{\boldsymbol{X}}. \quad (1)$$

$\boldsymbol{X} \in \mathcal{F}(\mathbb{X} \times \mathbb{L}_+)$ describes the state of the tracks, $Z \in \mathcal{F}(\mathbb{Z})$ is the multi-object measurement, Θ is the space of all valid measurement-to-track associations $\theta \in \Theta$, and $w^{(I,\theta)}(Z)$ is the weight of the hypothesis $(I, \theta) \in \mathcal{F}(\mathbb{L}_+) \times \Theta$, which implicitly expresses the association uncertainty [20]. Hence, the association θ connects the states with the measurements. However, as the LMB filter approximates this GLMB distribution by an LMB distribution with parameters [18]

$$r^{(\ell)} = \sum_{(I,\theta) \in \mathcal{F}(\mathbb{L}_+) \times \Theta} w^{(I,\theta)}(Z) 1_I(\ell), \quad (2a)$$

$$p^{(\ell)} = \frac{1}{r^{(\ell)}} \sum_{(I,\theta) \in \mathcal{F}(\mathbb{L}_+) \times \Theta} w^{(I,\theta)}(Z) 1_I(\ell) p^{\theta}(\cdot, \ell), \quad (2b)$$

this connection gets lost afterward. Yet, our approach works for the LMB filter by performing the same weighted averaging.

C. Combining Classifiers

We assume the following situation: An object belongs exclusively to one class $c \in \mathbb{C}$. A classifier tries to estimate the class based on some input d. In the following, the density of a continuously random variable is denoted with f, and the probability of a discrete random variable is denoted with P.

The combination of different classifiers with potentially different input features aims to increase accuracy and is itself a vast topic [22]. Based on the interpretation of the classifier output, different fusion rules are proposed in the literature. If the output is interpreted as fuzzy membership, fuzzy rules [23] can be applied; if interpreted as belief or evidence, Dempster-Shafer techniques [24, 25] can be used.

If, otherwise, the output is interpreted as posterior probability $P(c|d)$, like we also do, [26, 27, 28] suggest using an average or, more general, some other linear combination of the individual estimations. Whether this can be theoretically justified must be checked individually.

If the different inputs d_i are conditional independent, i.e., $f(d_1, \dots, d_n|c) = \prod_{i=1}^n f(d_i|c)$, the following rule is a direct consequence of Bayes [22]:

$$P(c|d_1, \dots, d_n) \propto f(d_1, \dots, d_n|c) P(c) \quad (3a)$$

$$= P(c) \prod_{i=1}^n f(d_i|c) \quad (3b)$$

$$\propto P(c)^{1-n} \prod_{i=1}^n P(c|d_i). \quad (3c)$$

This is sensitive in the sense that if even one classifier assigns a zero probability to a class, this class cannot be estimated regardless of the other classifiers.

In practice, other rules like the sum rule [22, 29] have proven to be more robust and deliver better results. Thereby, the multiplication is replaced by a summation. This is theoretically justified if the estimated posterior probabilities do not differ much from the prior and when neglecting higher order terms [22]. Thus, if $P(c|d_i) = P(c)(1 + \varepsilon_i)$ with small ε_i, it follows from (3) that

$$P(c)^{1-n} \prod_{i=1}^n P(c|d_i) = P(c) \prod_{i=1}^n (1 + \varepsilon_i) \quad (4a)$$

$$= P(c) \left[1 + \sum_{i=1}^n \varepsilon_i + \mathcal{O}(\varepsilon^2)\right] \quad (4b)$$

$$\approx (1-n)P(c) + \sum_{i=1}^n P(c|d_i), \quad (4c)$$

where $\mathcal{O}(\varepsilon^2)$ collects higher order terms of ε in the big-O notation. This leads to the following sum rule [22]

$$P(c|d_1, \dots, d_n) \propto (1-n)P(c) + \sum_{i=1}^n P(c|d_i). \quad (5)$$

The authors of [30] suggest a different setup by applying a Dirichlet conjugate prior to the unknown parameter p of the categorical distribution of the class c. This yields the model

$$c \sim \mathrm{Cat}(p), \quad (6a)$$

$$p \sim \mathrm{Dir}(\alpha). \quad (6b)$$

They motivate this by arguing that, for every sensor and measurement, the probability of measuring a specific class is independent [30]. In their work, they use the generative density $f(d|c_i)$ as input. However, since we do not expect a detector to deliver this density, we adapt their method to the more common posterior estimates.

D. Subjective Logic

The general idea of SL is to extend probabilistic logic by explicitly including the uncertainty about probabilities and multiple subjective beliefs about the same statement [17]. The key element in SL is an opinion ω, where each opinion corresponds to a Dirichlet distribution [17]. For the sake of brevity, we avoid any required knowledge on SL. Therefore, we describe our proposed methods completely self-contained based on Dirichlet distributions. However, it is straightforward to transfer them into the SL opinion notation.

III. Track Classification Method

This section describes in detail our proposed track classifier framework summarized in Fig. 1. Belonging to the explicit methods, our approach makes use of the class estimation of the detector to perform the track classification. There are two main points that need to be addressed:

- How to associate the object class estimation of the detector to the tracks?
- How to update the track class with the object class?

A straightforward answer to the first question is to reuse the associations of the filter. In an implementation, this can be achieved by augmenting the traditional state and measurement space with the classification information. The MOT does not use the augmented state, but whenever a measurement is used to update a state, the class estimation of that state is updated by that of the measurement. Theoretically, this can be justified if the object detection measurements and the object class measurements of the detector are independent. Additionally, the independence of the kinematic and classification state must be assumed.

The remainder of this section addresses the second question, i.e., we propose different discounting approaches and fusion operators based on Bayes' rule and SL for the prediction and for the update step.

A. Prediction

In the prediction step of the MOT, we perform a discounting of the class estimation with parameter $\delta \in [0, 1]$. With $\delta < 1$, the method discounts knowledge from previous time steps. This has a regularizing effect and puts more weight on the current estimation. It can be especially advantageous if the class of an object is not assumed to be fixed in time. With the extreme value $\delta = 0$, the method discards all old knowledge and effectively uses only the current measurement for the estimation.

1) Bayes Method: The Bayes method implements the discounting by a weighted average between the estimated distribution and a uniform distribution or, more general, some other categorical distribution. For the uniform distribution, this yields

$$P(c_{+,i}) = \delta P(c_i) + (1 - \delta)\frac{1}{|\mathbb{C}|}. \tag{7}$$

More generally, this could be seen as a Markov process, where D is an appropriate Markov matrix, i.e.,

$$P(c_+) = DP(c). \tag{8}$$

2) Subjective Logic-based Methods: The two SL-based methods implement the discounting by the well-known trust-discounting operator [17] with the same parameter δ.

B. Update

As we use the MOT association to map the class estimates to tracks, we can focus the description on the update of one track with one measurement only. In the multi-sensor case it is assumed that the update from different sensors can be applied one after the other. Note that this does not limit the MOT, which can handle the multiple measurements differently.

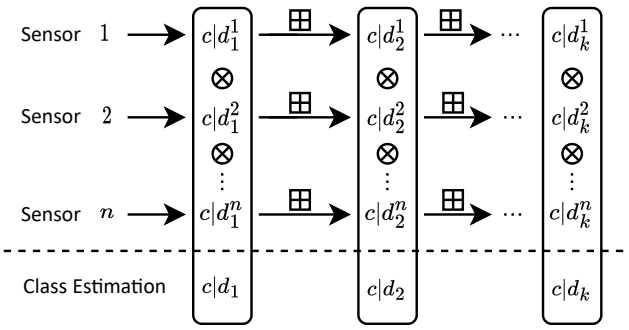

Fig. 2. The classification approach with Bayes method. $\boxplus$ denotes the sum rule and $\otimes$ the product rule. d_i^j denotes the data at time i of sensor j and d_i denotes the data of all sensors at time i.

1) Bayes Method: This method is based on [22]. In the single-sensor case, the same sensor estimates the class of the track at different times. This means that the sensor measures the same features, maybe from different perspectives, of the same object. As the features and also the detection and classification approach stay the same, there should be a high temporal dependency between the class estimations. Thus, the estimations should not change much over time, so applying the sum rule (5) is preferred over the product rule for this case.

In a synchronous multi-sensor setup, different sensors estimate the class at the same time. If the sensors are independent, e.g., by measuring different features with different sensor technologies, the independence of the sensors and, thus, of the class estimations justifies the application of the product rule (3).

This yields the procedure shown in Fig. 2, where class estimations of different sensors are fused with the product rule, and estimations of the same sensor at different times are fused with the sum rule. Finally, to get an estimation $\hat{c}$ of the class of a track, we choose the most likely class

$$\hat{c} = \arg\max_{c_i} \; P(c_i|d_1, \ldots, d_k). \tag{9}$$

2) Subjective Logic Moment Matching Method: We adapt the model of [30] to handle the posterior estimates of the detectors. We apply a Dirichlet prior with parameter α to the categorical distribution of the class with parameter p

$$c \sim \mathrm{Cat}(p), \tag{10a}$$

$$p \sim \mathrm{Dir}(\alpha). \tag{10b}$$

If $P(c_i|d) = 1$ for some i and $P(c_j|d) = 0$ for all other $j \neq i$, the posterior of p given class c_i is given by the well known formula [30]

$$f(p|c_i) = \frac{P(c_i|p)f_\alpha(p)}{\int_\Delta P(c_i|p)f_\alpha(p)dp} \tag{11a}$$

$$= \frac{p_i f_\alpha(p)}{\int_\Delta p_i f_\alpha(p)dp} = f_{\alpha+e_i}(p), \tag{11b}$$

where e_i is the i-th standard basis vector, Δ is the support of the Dirichlet distribution, and here and in the following f_α

denotes the density of a Dirichlet distribution with parameter α. The calculation uses the fact that $p_i f_\alpha(p) = \frac{\alpha_i}{S} f_{\alpha+e_i}(p)$ with $S = \sum_i \alpha_i$, which can be shown by a direct comparison of the two densities.

More general, let $P(c_i|d) = l_i \in [0, 1]$. Then

$$f(p|d) = \sum_i f(p|c_i) P(c_i|d) \tag{12a}$$

$$= \sum_i l_i f_{\alpha+e_i}(p). \tag{12b}$$

This is a mixture of Dirichlets, where each component is the result of the update (11) with a certain class and with the weight given by the probability of that class.

The problem is the growing number of mixtures. After n updates, we have $\binom{n+|\mathbb{C}|-1}{|\mathbb{C}|-1} = \mathcal{O}(n^{|\mathbb{C}|-1})$ mixture components. Thus, a practical implementation requires a suitable pruning or merging strategy. We propose to approximate the mixture by a single Dirichlet distribution using the moment matching (MM) approach from [30] as a computationally cheap and analytically closed solution. The parameter $\tilde{\alpha}$ of the single Dirichlet is given by [30]

$$\tilde{\alpha}_i = m_i \frac{\sum_k (m_k - v_k) m_k (1 - m_k)}{\sum_k (v_k - m_k)^2 m_k (1 - m_k)}, \tag{13}$$

where m_k and v_k are the first and the second non-central moments. In doing so, the expected value is preserved and the covariance is approximated in the least squares sense. The values for m_k and v_k are computed using (12) and result in

$$m_k = \mathrm{E}[p_k|d] = \int p_k \sum_i l_i f_{\alpha+e_i}(p) dp \tag{14a}$$

$$= \sum_i l_i \frac{\alpha_k + \delta_{ik}}{S+1} = \frac{\alpha_k + l_k}{1+S}, \tag{14b}$$

$$v_k = \mathrm{E}[p_k^2|d] = \int p_k^2 \sum_i l_i f_{\alpha+e_i}(p) dp \tag{15a}$$

$$= \sum_i l_i \frac{\alpha_k + \delta_{ik}}{S+1} \frac{\alpha_k + \delta_{ik} + 1}{S+2} \tag{15b}$$

$$= \frac{(1+\alpha_k)(\alpha_k + 2l_k)}{(1+S)(2+S)}, \tag{15c}$$

where δ_{ij} is the Kronecker-delta. This approximation can be applied at any time, e.g., after every update step or when the number of components exceeds a given threshold.

To estimate the class, first, the parameter p gets marginalized out, and then the most likely class is chosen. If the posterior is given by a Dirichlet with parameter α, this yields

$$\hat{c} = \arg\max_{c_k} \; P(c_k|\alpha) = \arg\max_{c_k} \; \frac{\alpha_k}{S}. \tag{16a}$$

In this method, we do not need to differentiate between single and multi-sensor setups, but regardless of their origin, measurements at the same time step are fused using the procedure described above.

Because of the connection between Dirichlet distributions and SL, we call this the SL MM method. Note, however, that the update rule (12) together with the merging (13) define a newly developed SL operator for fusing probabilistic evidence.

3) Subjective Logic Cumulative Belief Fusion Method: Another possibility for the update in an SL sense can be based on the quite commonly used cumulative belief fusion (CBF) operator [17]. For this, the object class estimation is interpreted as an SL opinion, i.e., the class estimation $P(c|d)$ is transferred to a Dirichlet distribution with parameter $\alpha_Z = \alpha_0 + P(c|d)$, where α_0 is a prior. Then, the parameter α_k after the update of α_{k-1} is given by $\alpha_k = \alpha_{k-1} + \alpha_Z$. In this method, we again do not need to differentiate between single and multi-sensor setups. We refer to this method as SL CBF.

IV. Experimental Results

In this section, we evaluate and compare our proposed methods in simulation and on real-world data. First, we describe the different experimental setups, then we present the results.

A. Experimental Setups

For the scenarios, we consider five different classes, i.e., Pedestrian, Car, Truck, Bike, and Unknown. In the simulation, the ground truth class of an object is fixed in time and randomly selected at the birth of an object. In the real world experiment, the ground truth of the objects neither changes over time. The approximation of the mixture in the SL MM method is performed after every update step. An LMB filter [18] is used as an example for RFS filters to retrieve the measurement-to-track associations for the classifier in the multi-object scenarios. For the single-object evaluations, however, no actual tracking is required for evaluating the classifier, because the association is known beforehand. The association of tracks to the ground truth is done with the association computed by the generalized optimal sub-pattern assignment metric (GOSPA) metric [31]. As we do not evaluate the tracking performance, the GOSPA itself is not shown. The classification performance is evaluated with the weighted average F1-score. To compare the results of the different setups, the performance is not plotted over time, but over the track age measured in update steps k and averaged over all tracks with the same age.

1) Real-World: For the real-world experiment, we use one of our automated vehicles at the Institute of Measurement, Control and Microtechnology at Ulm University, which is described in [32].The tracking uses one Velodyne VLP 32 lidar sensor with the detector described in [33] and three Continental ARS 408 radar sensors, where each sensor has its own detector based on [34]. Figure 3 shows the path of the ego vehicle in black as well as the class and the path of the tracked objects. The ground truth class of each track is labeled manually based on camera data. It is sufficient to label only tracks, as we are only interested in the classification performance and do not evaluate the tracker itself.

2) Multi-Object Simulation: We use the software in the loop (SIL) framework [36] to simulate a scenario similar to the real-world setup. The ego-vehicle in the simulation follows a fixed path in the map of Fig. 3 and tracks other vehicles. The other vehicles are spawned at random time intervals at fixed locations

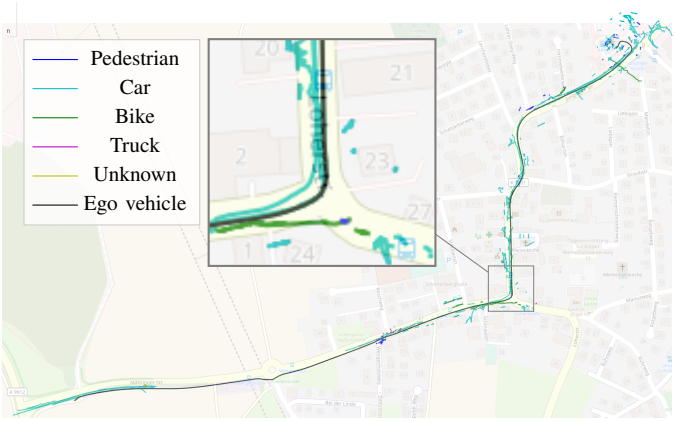

Fig. 3. Tracks with ground truth class. In total, 367 tracks corresponding to real objects are computed. 76% of these tracks belong to cars, 3% to trucks, 16% to pedestrians, and 5% to bikes. Map from [35].

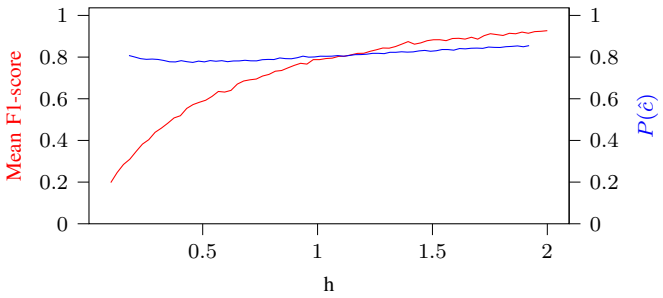

Fig. 4. Classification result and the probability of the estimated class with different parameters of the Dirichlet distribution that models the detector with $l = 0.1$ fixed and for varying h.

and follow a path that intersects the ego-vehicle's field of view. The ego-vehicle uses the same sensors for tracking as in the real experiment. Each sensor has its own detector that correctly classifies an object to 50%, otherwise a randomly chosen class is selected. The selected class has always a fixed probability of 80%. The remaining 20% are distributed equally among the remaining classes. The experimental results with this setup are averaged over 150 Monte-Carlo (MC) runs. This yields around 3000 samples per time step k, as each MC run is composed of multiple objects and tracks.

3) Single-Object Simulation: As mentioned before, in this setup, no actual tracking is required to evaluate our classifier. The class estimation of the detector is modeled by a Dirichlet distribution with parameter α^D. If the true object class is c_i, then $\alpha_i^D = h$ and $\alpha_j^D = l$ for all other $j \neq i$ with $h > l$. One object class estimation is then sampled from that distribution. An often observed characteristic of machine learning-based detectors is that a high probability is estimated even for falsely classified objects. This behavior can be captured with values $h, l < 1$, which result in a density where most of the weight is in the corners of the support of the Dirichlet distribution. Figure 4 shows the average behavior of the detector model from 1000 MC runs. Depending on the value of l and h, the performance varies between extremely bad and nearly perfect. However, the probability of the estimated class $P(\hat{c})$ is almost

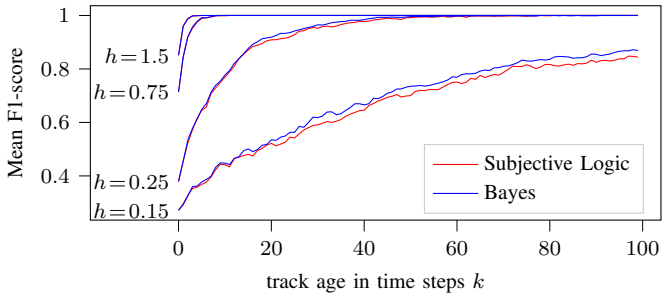

Fig. 5. Classification result with varying detector quality for a single-sensor setup with detector parameters $l = 0.1$ and varying h.

constant at 80%. The experimental results with this setup are averaged over 1000 MC runs per time step k.

B. Results from Single-Object Simulations

In this section, we evaluate three different experiments with the setup described in Section IV-A3.

1) Varying Detector Quality: Figure 5 shows the mean F1-score in a single-sensor setup with detectors of different classification quality. In the single-sensor case, we apply the sum rule to fuse the Bayes estimations, as does the SL CBF operator, which also sums up the different evidence. Thus, in a single-sensor setup, the Bayes and SL CBF methods result in exactly the same estimation. Therefore, they are shown together as one curve.

The quality of the detectors ranges from a very bad F1-score of about 0.2 up to a very good one of about 0.9. The results show that the fusion is, in all cases, beneficial compared to the native single-frame performance of the detector. The improvement is not linear: while the classification with detectors with a single time step F1-score of 0.4 or better achieve a nearly perfect score after at least 40 time steps, the classification with the worst detector in the figure would need roughly 400 time steps to achieve a nearly perfect score. Further, the difference between the various proposed methods vanishes for high values of h, i.e., for high-quality detectors.

Remarkable is the slightly worse behavior of the SL MM method, especially with worse detectors. This behavior is not caused by the approximation of the mixture, but is a direct consequence of the model, which represents time-varying classes. In the internal model, at every time step, the class is drawn randomly from the categorical distribution (10a). While this reduces the performance in ideal settings, it enhances the method's robustness. Figure 6 shows exemplarily the effect of switching detector performance. At time step 50, h raises from 0.12 to 0.2. While the Bayes and the SL CBF methods perform better for any fixed h, the SL MM method outperforms the other two in the switching situation.

2) Varying Number of Sensors: Figure 7 shows the performance using a varying number $|S|$ of independent sensors and detectors. An increasing number of sensors increases the classification performance. The SL MM and the SL CBF methods handle measurements of different detectors and of different time steps k equally. This means that n times the

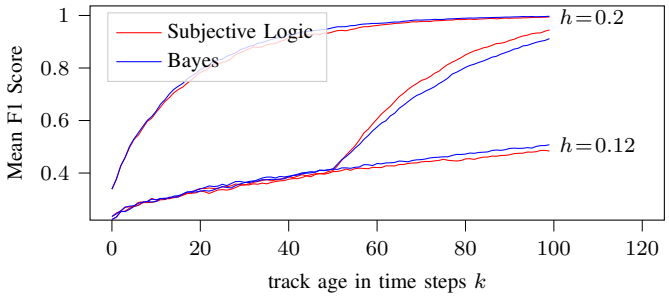

Fig. 6. Classification result with a single-sensor setup with and without switching detector performance by a switching h parameter at time step 50 from 0.12 to 0.2. $l = 0.1$ constant.

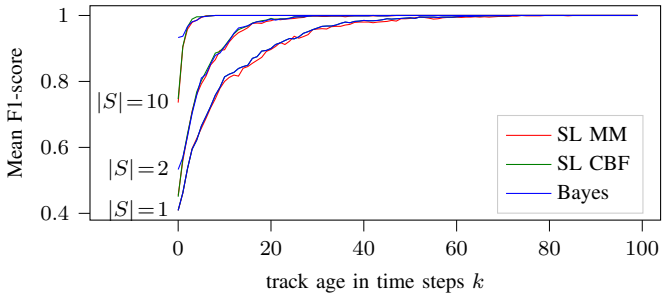

Fig. 7. Classification result with a varying number of independent sensors $|S|$, where each sensor has its own detector with parameters $l = 0.1$ and $h = 0.25$. The results for the SL CBF method are mostly covered by that of the Bayes method.

number of sensors takes $1/n$ times the time steps to achieve the same performance.

C. Results from Multi-Object Simulations

The general characteristics of the different proposed methods have already been demonstrated using the single-object setup, and no additional effects have been observed for the multi-object setup. Therefore, in this section, we present only one experiment with the setup described in Section IV-A2, of which Fig. 8 shows the results. As the SL CBF method has almost the same behavior as the Bayes method, the plot focuses on the SL MM and the Bayes method. Similar to the single-object case, the first few updates increase the classification performance rapidly. Also, the Bayes method performs in the beginning slightly better than the SL MM. Because of wrongly associated measurements, the classification does not reach a perfect score, and the Bayes method reaches a plateau after roughly 50 updates. The SL MM method handles the wrongly assigned measurements better. Its F1-score improves steadily and becomes higher than that of the Bayes method after approx. 100 updates. This again shows the greater robustness of the SL MM method, which can be particularly advantageous for MOT with high association ambiguity.

D. Results from Real-World Data Evaluation

This section presents the results of the real-world data experiment described in Section IV-A1. Figure 9 shows the mean F1-score and the score for the two largest classes (cars and pedestrians). The performance for cars is significantly better

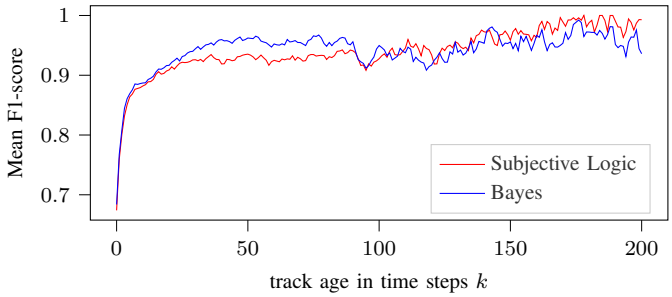

Fig. 8. Classification results for the multi-sensor setup.

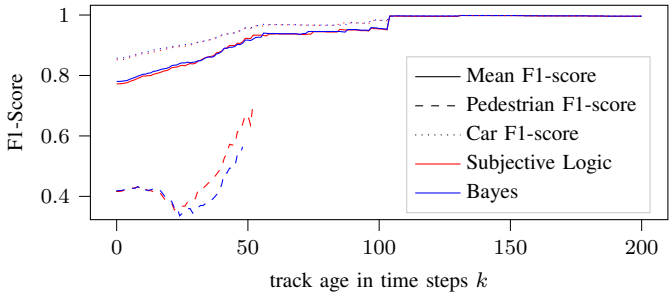

Fig. 9. F1-score of the classification in the real-world scenario.

than for the pedestrians, because the detector is optimized for cars and the driving task. For cars, the two methods perform almost equally. This is consistent with the result in Fig. 5 for high-quality detectors. For pedestrians, the SL MM method is better. This can possibly be explained by the greater robustness of the method against wrongly assigned measurements as shown in Fig. 8. We see the real world scenario as a proof-of-concept. As the performance of the track classification is directly coupled to that of the detectors, it will automatically benefit from improvements in the detectors.

E. Computational Effort

To evaluate the computational costs of our methods, we have measured the processing times of the tracker with and without our classifier running within the tracking module. The classification only increases the average processing time of the tracking module by about 12% for the Bayes method and 16% for the SL MM method. This is independent of the scenario. To get some absolute numbers, in this specific scenario with an average number of about 12 detections, the classification increases the average runtime for one prediction and update cycle from 0.74ms to 0.83ms for the Bayes method and 0.87ms for the SL MM method. The tests have been performed on a AMD Ryzen 9 7950X CPU with a single-threaded tracking implementation. Thus, in both cases, the tracking module with classifier is still real-time capable.

V. Conclusion

This work has presented a novel framework for track classification for RFS trackers based on object class estimations of an object detector. Our framework can be used for trackers that have a unique measurement-to-track association in the

update, e.g. RFS trackers, and the algorithm of the underlying tracker does not need to be changed. For our framework, we have presented three different methods for updating the class estimation, one of which additionally defines a novel fusion operator in the SL framework. All methods clearly improve the classification performance compared to that of the detector, while the computational costs are low and scale only linearly with the number of classes.

REFERENCES

[1] Y. Bar-Shalom, T. Kirubarajan, and C. Gokberk, "Tracking with classification-aided multiframe data association," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 41, no. 3, pp. 868–878, 2005.

[2] R. Matthaei *et al.*, "Autonomous driving," in *Handbook of Driver Assistance Systems: Basic Information, Components and Systems for Active Safety and Comfort*, 2016, pp. 1519–1556.

[3] M. Li, Z. Jing, P. Dong, and H. Pan, "Multi-target joint detection, tracking and classification using generalized labeled multi-bernoulli filter with bayes risk," in *International Conference on Information Fusion*, 2016, pp. 680–687.

[4] B. T. Vo and B. N. Vo, "Tracking, identification, and classification with random finite sets," *Defense, Security, and Sensing*, vol. 8745, p. 87450D, 2013.

[5] W. Yang, Y. Fu, and X. Li, "Joint target tracking and classification via RFS-based multiple model filtering," *Information Fusion*, vol. 18, pp. 101–106, 2014.

[6] W. Yang, Z. Wang, Y. Fu, X. Pan, and X. Li, "Joint detection, tracking and classification of a manoeuvring target in the finite set statistics framework," *IET Signal Processing*, vol. 9, no. 1, pp. 10–20, 2015.

[7] G. Li, P. Wei, G. Battistelli, L. Chisci, and L. Gao, "Multi-sensor joint target detection, tracking and classification via bernoulli filter," 2021. [Online]. Available: http://arxiv.org/abs/2109.11259.

[8] R. Zhan, L. Wang, and J. Zhang, "Joint tracking and classification of multiple targets with scattering center model and CBMeMBer filter," *Sensors*, vol. 20, no. 6, p. 1679, 2020.

[9] W. Mei, G.-l. Shan, and X. Li, "Simultaneous tracking and classification: A modularized scheme," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 2, pp. 581–599, 2007.

[10] R. Girshick, "Fast r-CNN," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[11] W. Liu *et al.*, "SSD: Single shot MultiBox detector," in *Computer Vision – ECCV 2016*, 2016, pp. 21–37.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 779–788.

[13] S. Haag, B. Duraisamy, W. Koch, and J. Dickmann, "Classification assisted tracking for autonomous driving domain," in *IEEE Sensor Data Fusion: Trends, Solutions, Applications*, 2018, pp. 1–8.

[14] B.-N. Vo, B.-T. Vo, and M. Beard, "Multi-sensor multi-object tracking with the generalized labeled multi-bernoulli filter," *Transactions on Signal Processing*, vol. 67, no. 23, pp. 5952–5967, 2019.

[15] R. P. S. Mahler, *Statistical multisource-multitarget information fusion*. Boston: Artech House, 2007.

[16] S. Shi *et al.*, "PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3d object detection," *International Journal of Computer Vision*, vol. 131, no. 2, pp. 531–551, 2023.

[17] A. Jøsang, *Subjective Logic*. Cham: Springer International Publishing, 2016.

[18] S. Reuter, B.-T. Vo, B.-N. Vo, and K. Dietmayer, "The labeled multi-bernoulli filter," *IEEE Transactions on Signal Processing*, vol. 62, no. 12, pp. 3246–3260, 2014.

[19] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *Computer Vision – ECCV 2020*, vol. 12346, 2020, pp. 213–229.

[20] B.-T. Vo and B.-N. Vo, "Labeled random finite sets and multi-object conjugate priors," *IEEE Transactions on Signal Processing*, vol. 61, no. 13, pp. 3460–3475, 2013.

[21] J. L. Williams, "Marginal multi-bernoulli filters: RFS derivation of MHT, JIPDA and association-based MeMBer," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 51, no. 3, pp. 1664–1687, 2015.

[22] J. Kittler, M. Hatef, R. Duin, and J. Matas, "On combining classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 226–239, 1998.

[23] Sung-Bae Cho and J. Kim, "Combining multiple neural networks by fuzzy integral for robust classification," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 25, no. 2, pp. 380–384, 1995.

[24] J. Franke and E. Mandler, "A comparison of two approaches for combining the votes of cooperating classifiers," in *Proceedings., 11th IAPR International Conference on Pattern Recognition*, 1992, pp. 611–614.

[25] G. Rogova, "Combining the results of several neural network classifiers," in *Classic Works of the Dempster-Shafer Theory of Belief Functions*, 2008, pp. 683–692.

[26] S. Hashem and B. Schmeiser, "Improving model accuracy using optimal linear combinations of trained neural networks," *IEEE Transactions on Neural Networks*, vol. 6, no. 3, pp. 792–794, 1995.

[27] L. Xu, A. Krzyzak, and C. Suen, "Methods of combining multiple classifiers and their applications to handwriting recognition," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 22, no. 3, pp. 418–435, 1992.

[28] J. Kittler, "Multiple expert fusion," in *Classification in the Information Age*, 1999, pp. 21–28.

[29] D. Tax, M. Breukelen, R. Duin, and J. Kittler, "Combining multiple classifiers by averaging or by multiplying?" *Pattern Recognition*, vol. 33, pp. 1475–1485, 2001.

[30] L. Kaplan, M. Şensoy, S. Chakraborty, and G. De Mel, "Partial observable update for subjective logic and its application for trust estimation," *Information Fusion*, vol. 26, pp. 66–83, 2015.

[31] A. S. Rahmathullah, A. F. Garcia-Fernandez, and L. Svensson, "Generalized optimal sub-pattern assignment metric," in *IEEE International Conference on Information Fusion*, 2017, pp. 1–8.

[32] M. Buchholz *et al.*, "Handling occlusions in automated driving using a multiaccess edge computing server-based environment model from infrastructure sensors," *IEEE Intelligent Transportation Systems Magazine*, vol. 14, no. 3, pp. 106–120, 2022.

[33] M. Herzog and K. Dietmayer, "Training a fast object detector for LiDAR range images using labeled data from sensors with higher resolution," in *IEEE Intelligent Transportation Systems Conference*, 2019, pp. 2707–2713.

[34] A. Danzer, T. Griebel, M. Bach, and K. Dietmayer, "2D car detection in radar data with PointNets," in *IEEE Intelligent Transportation Systems Conference*, 2019, pp. 61–66.

[35] OpenStreetMap contributors, *Planet dump retrieved from https://planet.osm.org*, https://www.openstreetmap.org, 2023.

[36] J. Strohbeck, J. Müller, A. Holzbock, and M. Buchholz, "Deepsil: A software-in-the-loop framework for evaluating motion planning schemes using multiple trajectory prediction networks," in *International Conference on Intelligent Robots and Systems*, 2021, pp. 7075–7081.