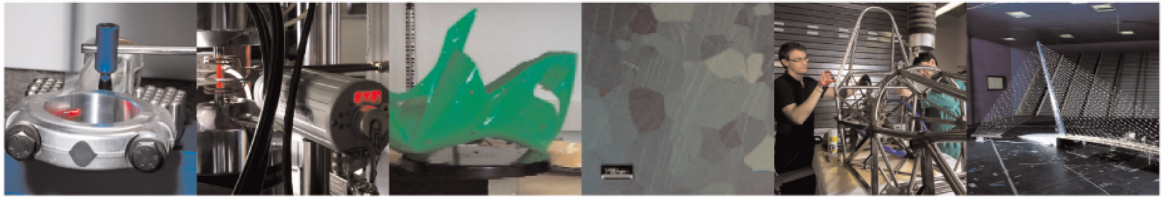




POLITECNICO
MILANO 1863

DIPARTIMENTO DI MECCANICA



Extended Object Tracking in Curvilinear Road Coordinates for Autonomous Driving

Pragyan Dahal, Simone Mentasti, Stefano Arrigoni, Francesco Braghin, Matteo Matteucci, and Federico Cheli

This is a post-peer-review, pre-copyedit version of an article published in *IEEE Transactions on Intelligent Vehicles*. The final authenticated version is available online at:
<http://dx.doi.org/10.1109/TIV.2022.3171593>

© 2022 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

This content is provided under [CC BY-NC-ND 4.0](https://creativecommons.org/licenses/by-nc-nd/4.0/) license



Extended Object Tracking in Curvilinear Road Coordinates for Autonomous Driving

Pragyan Dahal
Dept. of Mechanical Engineering
Politecnico di Milano
Milano, Italy
pragyan.dahal@polimi.it

Simone Mentasti
Dept. of Electronics, Information
and Bioengineering
Politecnico di Milano
Milano, Italy
simone.mentasti@polimi.it

Stefano Arrigoni
Dept. of Mechanical Engineering
Politecnico di Milano
Milano, Italy
stefano.arrigoni@polimi.it

Francesco Braghin
Dept. of Mechanical Engineering
Politecnico di Milano
Milano, Italy
francesco.braghin@polimi.it

Matteo Matteucci
Dept. of Electronics, Information
and Bioengineering
Politecnico di Milano
Milano, Italy
matteo.matteucci@polimi.it

Federico Cheli
Dept. of Mechanical Engineering
Politecnico di Milano
Milano, Italy
federico.cheli@polimi.it

Abstract—In literature, Extended Object Tracking (EOT) algorithms developed for autonomous driving predominantly provide obstacles state estimation in cartesian coordinates in the Vehicle Reference Frame. However, in many scenarios, state representation in road-aligned curvilinear coordinates is preferred when implementing autonomous driving subsystems like cruise control, lane-keeping assist, platooning, etc. This paper proposes a Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter with an Unscented Kalman Filter (UKF) estimator that provides obstacle state estimates in curvilinear road coordinates. We employ a hybrid sensor fusion architecture between Lidar and Radar sensors to obtain rich measurement point representations for EOT. The measurement model for the UKF estimator is developed with the integration of coordinate conversion from curvilinear road coordinates to cartesian coordinates by using cubic hermit spline road model. The proposed algorithm is validated through Matlab Driving Scenario Designer simulation and experimental data collected at Monza Eni Circuit. *The Experimental Dataset will be made publicly available upon the paper acceptance*

Index Terms—Extended Object Tracking, Curvilinear Road Coordinates, Sensor Fusion, Lidar, Radar, GM-PHD

I. INTRODUCTION

Correct perception of the ego vehicle’s surrounding environment is central in implementing autonomous driving. However, the dynamic and uncertain nature of the environment makes this task very challenging. As the ego vehicle perceives the environment through sensors like Camera, Lidar, and Radar, the data collected from

Supported by project TEINVEIN: TEcnologie INnovative per i VEicoli Intelligenti, CUP (Codice Unico Progetto - Unique Project Code): E96D17000110009 - Call “Accordi per la Ricerca e l’Innovazione”, cofunded by POR FESR 2014-2020 (Programma Operativo Regionale, Fondo Europeo di Sviluppo Regionale - Regional Operational Programme, European Regional Development Fund).



Fig. 1: Image of the experimental vehicle (Ego Vehicle) instrumented with Lidar, Radar and Camera employed for the dataset acquisition and algorithm validation

these sensors need to be processed to an understandable format to make driving decisions. The driving software’s detection and tracking blocks consistently provide the state of various objects of interest (e.g., vulnerable road users, road traffic participants, etc.) to the decision stack. Our work focus on obstacle tracking with two of the most used sensors in autonomous driving, Lidar, and Radar.

Research on object tracking has led to multi-object tracking algorithms with point object representation, called point object tracking (POT). Early research with this representation merely provided information about the obstacle’s kinematic state while disregarding its extent. Recent algorithms, ([1], [2] [3]) with detect and then track approaches can also be categorized into this segment. The POT representation assumes that an object

could generate at most one measurement at one time instance. One substantial drawback with this approach is the propagation of detection error to the tracking algorithm. While learning-based detectors, [4] do provide sufficiently accurate detections for driving tasks, they require the use of high-resolution lidar sensors which can provide distinguishable object features. However, working with low-resolution lidar, the detections from the learning-based algorithms are not accurate due to the lack of these features. One alternative to the learning-based detections is occupancy grids-based detections [5]. The obstacles detected with this approach do not have exact object shape information and are noisy in yaw angle estimation due to the grid discretization. Another promising alternative is developing a tracking algorithm that eliminates the detection step or employs minimal pre-processing. Extended object tracking algorithms are developed with an assumption that an object can generate multiple measurements at a single time instance [6]. This allows for the development of tracking recursions where minimally pre-processed sensor data can be used. These measurements provide tentative information on the spatial occupancy of the object, therefore enabling the state of the obstacle to be modeled with the kinematic and extent information.

Most of the works in extended object tracking are focused on providing the state of the obstacles in cartesian coordinates in the Vehicle Reference Frame (VRF) of the ego vehicle [6]. While obstacle state information in cartesian coordinates is sufficient for some tasks, for applications that require information sharing, such as V2V and V2X, integration of the road model into the estimation process and state representation in curvilinear coordinates is advantageous. In particular, for information sharing, which enables easier lane level localization for participating agents [7]. In addition to this, road model integration and curvilinear state representation expedite the process of dynamic path planning for the ego vehicle motion as well [8]. In literature, EOT algorithms are developed using measurements coming from standalone sensors. In [9], authors use a high-resolution Radar sensor to develop a measurement model for a Bayesian estimation process. An extent rich and highly accurate 2D PointCloud obtained from a Lidar sensor is also employed to perform this task in [10]. Radar measurements, which provide information on object velocity, are observed to be extremely noisy. In contrast, lidar points with highly accurate positions do not provide information on the object's speed. Fusion between these two sensors opens the door to extract a highly enriched representation of the surrounding obtained by combining complementary information. The common fusion approaches in literature can be categorized into either model based, [11] or data driven [12].

This work proposes an EOT algorithm that provides the obstacle state in curvilinear road coordinates by

integrating a Cubic Hermite spline road model into the estimation routine. We use a fused representation of the measurement points obtained from Lidar and object detections obtained from Radar sensors using model based fusion. The proposed algorithm is validated through simulation and experimental data while also comparing with other state of the art algorithms.

The remainder of the paper is structured in the following fashion. Related works on EOT and road model integration into tracking routine are discussed in Section II. In the Section III, the hybrid sensor fusion architecture employed in this work is presented. Section IV presents the tracking problem. We present the pipeline for the EOT procedure in detail in Section VI. Algorithm validation through simulation data and experimental data recorded by vehicle shown in Fig. 1 is presented in Section VII. Section VIII concludes the paper.

II. RELATED WORKS

A. Extended Object Tracking

An extensive overview on EOT is presented in [6]. Authors have grouped EOT into three broad categories based on the different approaches used to develop measurement models. The first modeling approach assumes that measurements primarily originate from reflection points rigidly fixed to the extended objects. Authors in [13] develop such EOT algorithm for rectangular objects using radar resolution model. This work is centered around the assumption that Radar detections from a vehicle originate from particular reflection points, for example, wheel housings, headlamps, etc. In the second approach, the measurement model is based on an inhomogeneous Poisson Point Process (PPP). The detections are assumed to be spread spatially around the target, and the PPP is used to model their probability. Random Matrix models are the predominant variant of this approach; in [14], authors develop a Gamma Gaussian Inverse Wishart Poisson Multi Bernoulli (GGIW PMB) EOT filter using this model. They assume that the measurements are Gaussian distributed around the target center while the number of measures is Poisson distributed. Based on this assumption regarding the measurement model, a Gamma-Gaussian-Inverse Wishart conjugate prior is used to develop the Bayesian recursion for filtering. In [15], a Random Hyper Surface Model, another variant of the spatial model, is proposed for estimating kinematic and shape estimates of ellipses and star convex shaped objects. The measurement model is developed by assuming that the measurement points are generated from sources located on the scaled boundary of the predefined shape. This scaling factor is assumed to be a random variable. Filtering recursion is performed without actually estimating the location of these assumed measurement sources. Finally, The third approach includes techniques that model the physics behind the

sensor operation into measurement models; [16] use direct scattering technique to perform EOT.

While some works have been developed with the predefined assumption of the object shapes, like ellipses in [17], rectangles in [10], a parametric approach that defines the boundary of an object with star convex modeling is also gaining momentum. Authors in [15] use Fourier Coefficients to parametrize the shape of the radial function of the contour while authors in [18] use Gaussian Processes (GP). In [18], authors develop an Extended Kalman Filter (EKF) based estimator by approximating Gaussian Process into a state-space model, which in turn is augmented with the kinematic state of the track. Similarly, EKF based tracker for detections obtained from Radar sensors is developed in [9]. Doppler rate information obtained from the high-resolution Radar sensor is exploited to better estimate the translational and rotational velocities of the track.

In recent developments, machine learning-based approaches are also proposed in the literature to develop measurement models. In [19], Variational Gaussian Mixture modeling is used to form a Bayesian Framework for EOT for detections obtained from Radar sensors. [20] proposes a hierarchical truncated Gaussian model learned through raw data collected from a Radar sensor.

B. Road Model Integration into Tracking Recursion

A model of the geometric representation of the road can be computed online using images acquired by camera sensors [21], [22]. Deep learning techniques are implemented to perform image segmentation and provide road identification. However, results obtained by such online frameworks are not always robust enough to be integrated into tasks further down the pipeline due to information blackout created by occlusions or in cases of extreme weather scenarios. When the algorithm cannot rely on the information acquired through image data, localization and obstacle tracking are performed by integrating predefined HD map of the road into these routines [23], [24], [25].

Integration of the road model into the tracking routine firstly provides the possibility of filtering out sensor measurements obtained from objects outside road bounds. This also enables to restrict the tracking region to be constrained within the road bounds, which is sufficient for autonomous highway applications like platooning, lane-keeping, etc. [25]. For the development of connected architectures, authors in [24] demonstrate that the integration of the road model expedites the lane level localization of the participating vehicles. Furthermore, in [22], authors develop a framework for lane situation assessment by integrating the road model in the point object-based estimation routine. In their work, a track-level decentralized fusion of object detections from Radar sensors is performed using the nearest neighbor

approach. Track estimates provided in cartesian coordinates are later converted into road-aligned curvilinear coordinates to perform lane situation assessment using a constant curvature conversion model based on osculating circle assumption. The authors generate a cubic hermit spline road model from lane detections obtained from the camera sensor to perform this conversion. In [26] the previous work is extended by performing fusion between Lidar, Radar, and Camera detections to generate a consistent point object measurement list. These measurements are converted into road curvilinear coordinates used to develop a tracking and behavior reasoning framework for the obstacles. This enables authors to integrate road geometry constraints into a unified interactive multi-model (IMM) tracking and behavior reasoning module.

Authors in [8] demonstrate the efficiency introduced in trajectory planning for ego vehicle motion when the road model is integrated into the planning process. Ego vehicle and road participants are localized in the road curvilinear coordinates, which makes it possible to remove the curvature effects of the road in the trajectory planning problem and hence generate requested maneuvers in the straight-road like scenarios.

Finally, in [27], authors integrate road model and road constraints with learning-based techniques to perform trajectory prediction of the surrounding road vehicles. The past and present motion of the road vehicles represented as a bounding box is obtained by using Global Navigation Satellite Systems (GNSS) and Lidar sensors. However, in a real-world driving scenario, where we assume no exchange of information between the ego vehicle and road participants, the kinematic state and extent of the obstacles can only be obtained by tracking them using exteroceptive sensors. Model Predictive Controller developed in [28] assumes elliptical-shaped obstacles whose states are expressed in the curvilinear coordinates and derive the constraints for the optimization problem. The algorithm is developed assuming that the obstacle state is deterministic in the curvilinear coordinate, which does not represent the real-world driving scenario, where obstacles are generally perceived using various exteroceptive sensors. Integration of an extended object tracking algorithm in curvilinear coordinates into the pipeline provides the obstacle state for the algorithm with reasonable closeness to attainable accuracy.

These works demonstrate the benefits of road model integration in the ego vehicle state estimation and obstacle tracking. However, sufficient attention has not been given to the accurate representation of the obstacles in the curvilinear coordinates, which is a crucial step in tasks such as road situation assessment [22], behavior prediction [26], obstacle trajectory prediction [27], ego vehicle trajectory generation, and controller implementation [8], [28]. Similarly, the literature addresses the EOT problem for a single sensor only, either for Lidar or

Radar, while no attempt has been made to exploit the fused representation for EOT.

In this work, we propose an EOT algorithm that provides an extent rich representation of the obstacles by implementing a curvilinear coordinate-based rectangular measurement model in tracking recursion and using fused measurement points from Lidar and Radar sensors. This work presents the first approach to perform EOT in curvilinear coordinates while using fused measurements from Lidar and Radar sensors for the purpose. The core contributions of the paper are:

- Extends the POT algorithm developed in [25] to EOT in curvilinear road coordinates
- Implements the integration of road model in EOT recursion for measurement filtering and birth intensity definition
- Analyzes sensor fusion between lidar points and radar detections for EOT.
- Demonstrates that the obstacle state representation in curvilinear coordinates increases accuracy of state estimation, especially for yaw angle estimation.

III. SENSOR DATA ACQUISITION AND PREPROCESSING

The experimental vehicle used to validate the proposed algorithm is equipped with a Velodyne VLP-16 Lidar installed at the vehicle’s roof, which provides PointCloud at 10Hz. In addition, two Continental ARS 408-21 Radars are installed at the front and rear of the car, and a camera is facing forward [29]. The detections obtained from the Radar sensors are expressed in Vehicle Reference Frame (VRF) centered at the vehicle and are obtained at 14Hz . Radars already provide a cluster of detection supposedly generated from an obstacle. Hence, very limited preprocessing is required to express them in the cartesian coordinate VRF. Interested readers are referred to our previous work, [25] for a detailed explanation of the Radar preprocessing routine.

Raw Lidar measurements are obtained as 3D Point-Cloud referenced to the Lidar sensor. Then, a series of preprocessing operations are performed to this 3D Point-Cloud data to move it to the desired 2D representation to perform EOT. Indeed, in [10], the EOT algorithm is developed for the measurements obtained from a 2D Lidar sensor. Measurement in our work is also illustrated in a similar fashion, i.e, projected into ground plane, to perform rectangle fitting for developing the measurement model in the filtering recursion. The preprocessing pipeline for the Lidar data is illustrated in the Fig. 2. In the first step of the pipeline, the Lidar PointCloud is transformed to VRF from the Sensor Reference Frame (SRF). Then, a plane fitting algorithm based on RANdom SAMple Consensus (RANSAC) is used to remove the points corresponding to the ground plane, as shown in [30]. Consecutively, a preliminary filtering step is implemented by exploiting the knowledge of the road bounds to remove all points outside of it. At this stage

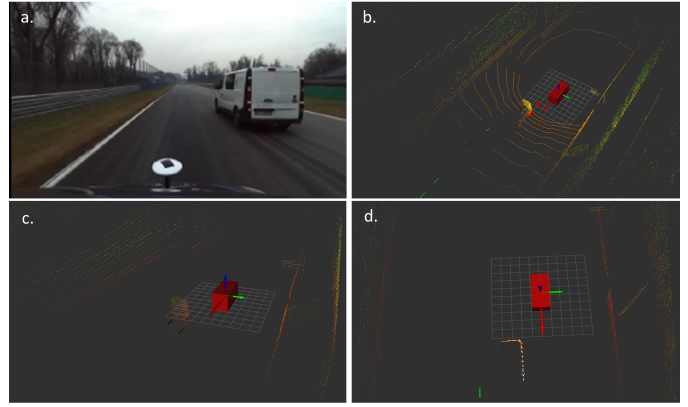


Fig. 2: Preprocessing of the Lidar PointCloud for the Experimental Scenario:

- Snapshot of the experimental environment
- 3D PointCloud data obtained from the scenario, ego vehicle represented by the red bounding box
- filtered 3D PointCloud as points corresponding to the ground plane are removed
- Pointcloud from object projected to the ground plane

of the PointCloud preprocessing, only the points supposedly originated from the dynamic obstacles within the road bounds are left. Any obstacle within the road bound is assumed to be dynamic and is under consideration for tracking in our application. The next step involves projecting the PointCloud into the road plane using the normal direction obtained in the previous RANSAC implementation step. The output of this step is a set of observed 2D points in VRF as illustrated by Fig. 2.d .

As assumed by the Poisson Point Process (PPP) spatial measurement model $p(\mathbf{Z}_k|\mathbf{X}_k)$ [10], only measurements generated from the surface of the object are required. However, the output of the last preprocessing step can possibly contain measurement points corresponding to the inside of the obstacle as well. Although not visible for our experimental setup due to our tracked vehicle’s nearly perfect rectangular nature, this scenario was fairly common in our simulation setup. Hence, in this step, measurements supposedly generated from the sources inside the vehicles are removed using a bearing clustering approach. In this approach, the bearing angles for all the measurement points are computed with reference to the VRF origin, and those points which fall under the threshold of some angle are clustered together. From each cluster, the closest point to the ego vehicle is chosen while all others are discarded.

A. Sensor Fusion and Measurement Clustering

EOT algorithms are developed assuming that multiple measurement points can generate from a single object. Hence, correct clustering of these points based on their source is crucial in developing these filters. However, considering all possible clusters or partitions of these

points in the update step of the filter can be computationally intractable [31]. Therefore, a clustering approach is used to compute measurement clusters that are believed to stem from a single object with high confidence. Lidar is used as the primary sensor, meaning the sensor fusion module provides output only when Lidar measurements are available. This fusion architecture aims to exploit the precise object shape information obtained from the Lidar sensor and complement it with velocity information obtained from the radar sensor. Position values obtained from radar detections were observed to be extremely noisy and are hence disregarded. Meaning measurements clusters are generated only when Lidar measurements are obtained. Based on the availability of the measurement, multiple scenarios can be anticipated:

1) *Only Lidar Measurements are available:* If only Lidar measurements are available at given time instance k , state estimates from last time instance, $k - 1$ are used to perform measurement clustering. Since the state estimates are in curvilinear coordinates (s, n) and measurements are in cartesian coordinates in VRF (\mathbf{E}), a conversion is required here. First, the state positional values of the tracked objects are converted to VRF (\mathbf{E}). Measurement points within the object's extent with some defined threshold are put together into a single cluster. Those points which are not selected into any cluster corresponding to the object state are further clustered using distance-based technique implemented in [31]. The measurement RFS, \mathbf{Z}_k is represented as union of the M_c^k cluster RFSs as expressed in Equation (1)

$$\mathbf{Z}_k = \bigcup_{c=1}^{M_c^k} \mathbf{C}_k^c, \mathbf{ID}_k = \{Id_c\}_{c=1}^{M_c^k} \quad (1)$$

where, M_c^k is the number of measurement clusters. In addition to this, each cluster set is assigned with an ID $,Id_c$ to identify the nature of its origin and type of sensors used to generate it. The measurement points in cluster \mathbf{C}_k^c only have positional values and are expressed as:

$$z_{k,j}^E = [x_j^E, y_j^E]_k \quad (2)$$

where, x_j^E and y_j^E are the relative position between the ego vehicle and an obstacle.

2) *Lidar and Radar Measurements are available:* When Radar measurements are also available in addition to Lidar measurements, multi-stage clustering is employed, and the Radar velocity components are integrated into the measurement points. First, the clustering discussed in the earlier section is performed, generating cluster sets from Lidar measurement. Then cluster to Radar detection association is performed to assign velocity values to measurement points inside the clusters. A greedy association based on the nearest-neighbor approach is done between Radar detections and cluster centroid. However, only those Lidar clusters that come within a

certain distance of radar detections are assigned velocity values; the remaining ones are left unchanged. This clustering approach can provide two kinds of measurement clusters, those with measurement points having positional values only, $z_{k,j}^E = [x_j^E, y_j^E]_k$ and those with positional and velocity values represented in Equation (3).

$$z_{k,j}^E = [x_j^E, y_j^E, V_{x,j}^E, V_{y,j}^E]_k \quad (3)$$

Relevant Id are also assigned to these clusters based on the fusion performed. Velocities, $V_{x,j}^E$ and $V_{y,j}^E$ are computed as components of absolute velocity in X and Y coordinates of VRF (\mathbf{E}).

IV. PROBLEM FORMULATION

A. Measurement and Track State Representation

The proposed algorithm aims at providing a state estimate of the obstacles around the ego vehicle in curvilinear road coordinates given the set of measurements, expressed as Random Finite Sets (RFS), \mathbf{Z}_k . The measurement points are divided into clusters by implementing the clustering algorithm discussed in Section III and represented by Equation (1). Hence, the input to the filtering algorithm would be M_c^k number of measurement clusters at any given time instance. If N_k is the random unknown number of the objects of interest in the FoV of the ego vehicle at time instance k , the Random Finite Set (RFS) representing the object states is:

$$\mathbf{X}_k = \{x_{1,k}, x_{2,k}, \dots, x_{N_k,k}\} \quad (4)$$

where, $x_{i,k}$ is the state of the i^{th} object and is made up of kinematic and shape values corresponding to the object:

$$x_{i,k} = [s_i, n_i, v_i, \zeta_i, \dot{\zeta}_i, L_i, W_i]_k^T \quad (5)$$

where, s_i is the relative distance between the ego vehicle and the track i in curvilinear axis, while n_i is the minimum lateral distance between the road centerline and the track. v_i is the track absolute velocity. ζ_i and $\dot{\zeta}_i$ are the track relative yaw and yaw rate respectively to the Road Reference Frame (\mathbf{R}_0). L_i and W_i are the length and width of the track. In Fig. 3, \vec{s}_e and \vec{n}_e are abscissa and ordinate of the Road Reference Frame (\mathbf{R}_e) centered at the nearest point corresponding to ego vehicle in road centerline. Similarly, \vec{s}_o and \vec{n}_o are abscissa and ordinate of the Road Reference Frame (\mathbf{R}_o) centered at the nearest point at the road centerline corresponding to the tracked object under consideration.

B. Multi-Object Bayesian Filtering Recursion

A multi object Bayesian Filtering framework is developed to infer the tracked object state recursively through the posterior probability distribution function (pdf)

$p(\mathbf{X}_k | \mathbf{Z}_{1:k})$ given the set of measurements till time instance k . The filtering recursion is developed with two

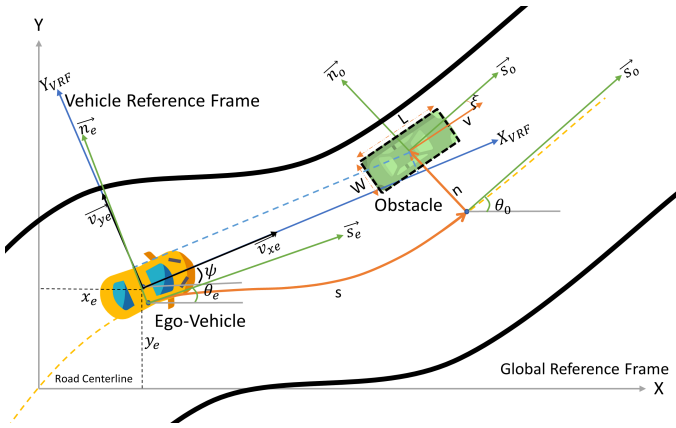


Fig. 3: Representation of the Ego Vehicle and Obstacle reference frames for the localization task on the road with the curvilinear road coordinates

different steps, namely: Chapman Kolmogorov Prediction and Bayes Update. Given the prior multi-object distribution $p(\mathbf{X}_{k-1}|\mathbf{Z}_{1:k-1})$ and motion model of the system $p(\mathbf{X}_k|\mathbf{X}_{k-1})$, predicted distribution $p(\mathbf{X}_k|\mathbf{Z}_{1:k-1})$ is computed using Chapman Kolmogorov Prediction formulation. To infer \mathbf{X}_k , at time instance k , we employ a Gaussian Mean Probability Hypothesis Density (GM-PHD) filter with unscented transformations to deal with the models nonlinearities. The detailed development of the filtering recursion is discussed in the Section VI.

V. ROAD MODEL AND COORDINATE CONVERSION

The proposed tracking algorithm is developed assuming that the road model is accurately known. A cubic hermit spline mathematical model is used to describe the road; readers are referred to [28] for details on this model. For every point in the road, polynomial parameters describing the road center-line position, its heading, and curvature 30m ahead are provided. The heading angle (θ) and road curvature (κ) are described using polynomial and are expressed as:

$$\begin{aligned} \theta(s) &= a_\theta s^3 + b_\theta s^2 + c_\theta s + d_\theta, \\ \kappa(s) &= a_\kappa s^3 + b_\kappa s^2 + c_\kappa s + d_\kappa \end{aligned} \quad (6)$$

where, s is a distance between the ego vehicle and object along the road centerline, both localized in the road reference frame. The parameters of these polynomials can be accessed by localizing the ego vehicle in the global road curvilinear coordinates. The origin of this coordinate system aligns with the origin of the Global Reference Frame, which is fixed at the starting point of the experiment in the Monza Circuit.

Coordinate conversion from curvilinear (s, n) to cartesian (x, y) coordinates, which is the first step of the proposed measurement model, can be found in our previous work, [25]. Pseudocode for this conversion is given in Algorithm 1. This conversion model is based on a

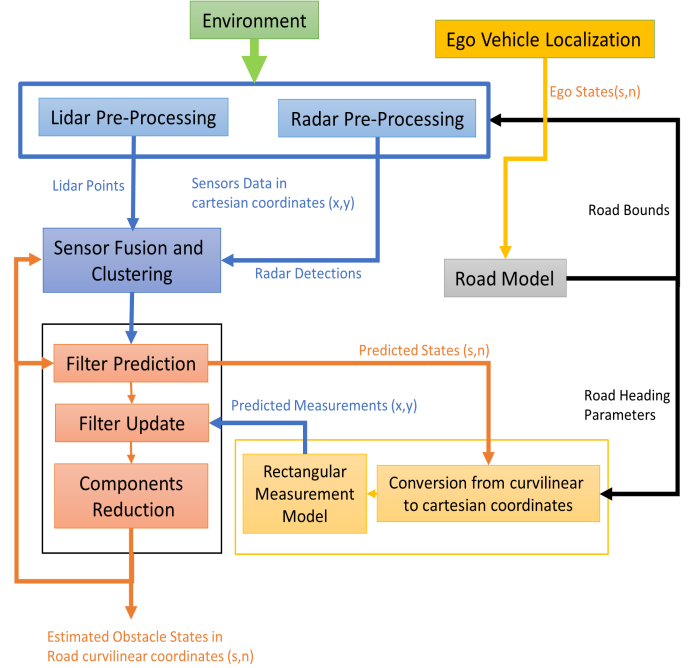


Fig. 4: Schema of the obstacle detection and tracking framework presented in this work. The input of the system are lidar and radar data, plus the Ego Vehicle localization. The output is a list of estimated obstacle states in curvilinear coordinates.

piece-wise Euler Integration method, which significantly differs from the model implemented in [22] and [26]. Authors in [22] implement a constant curvature-based conversion model, which assumes the road curvature to be constant in the conversion region. However, the proposed model depends only on the road heading angle, hence eliminating the errors propagated through the constant curvature assumption.

VI. GM-PHD FILTER FOR EXTENDED OBJECT TRACKING

In a Probability Hypothesis Density (PHD) filter, the recursive algorithm is developed assuming the prior, $p(\mathbf{X}_{k-1}|\mathbf{Z}_{k-1})$, predicted $p(\mathbf{X}_k|\mathbf{Z}_{k-1})$ and posterior $p(\mathbf{X}_k|\mathbf{Z}_k)$ distributions to be a Poisson multi-object distribution. The filtering recursion is developed by propagating the first order statistical moment of the distribution, i.e. PHD. A PHD $D(x)$ when represented as weighted Gaussian Mixture leads to Gaussian Mixture Probability Hypothesis Density (GMPHD) filter and can be parameterized as:

$$D(x) = \sum_{i=1}^N w^i \mathcal{N}(x : \mu^i, P^i) \quad (7)$$

Hence, PHD at different steps of filtering recursion is parametrized with a weight (w^i) assigned to a Gaussian

Distribution, which in turn is parametrized with its mean (μ^i) and covariance (P^i). The GMPHD filter discussed in [31] is used for performing the EOT. Readers are referred to this work for various assumptions and derivations of the filtering algorithm. In [10] and [31], authors use EKF as a Bayesian Estimator for performing object tracking; however, due to the highly non-linear measurement model involving multiple steps of conversion in our work, we implement UKF as our estimator. The general framework for filtering recursion is presented in the Fig. 4.

A. Filter Prediction

The predicted RFS, $\mathbf{X}_k|\mathbf{Z}_{k-1}$ is computed as an union of the surviving RFS from time instance $k-1$, $\mathbf{X}_{k-1}^S|\mathbf{Z}_{k-1}$ and the new birth set at time instance k , \mathbf{B}_k . The predicted distribution in the filtering recursion remains Poisson multi-object distribution. Its PHD, $D_{k|k-1}(x)$ can be expressed as:

$$D_{k|k-1}(x) = D_{k|k-1}^S(x) + \lambda_{b,k}(x) \quad (8)$$

where, $\lambda_{b,k}(x)$ is intensity of birth corresponding to birth model and $D_{k|k-1}^S$ is the intensity function corresponding to the surviving objects. The birth intensity is also approximated as Gaussian Mixture. Calculation of the parameters of $D_{k|k-1}^S$: $w_{k|k-1}^i$, $\mu_{k|k-1}^i$ and $P_{k|k-1}^i$ is based on UKF transformation through a constant turn rate motion model.

1) *Prediction of Surviving Objects*: To predict the motion of the surviving objects through time instances, we implement a constant turn rate motion model in curvilinear road coordinates. Tests were carried out with constant velocity motion model as well, but we validate the algorithm with constant turn rate model to integrate possible lane change scenarios. The center of rotation of the obstacle is assumed to be in the rectangle centroid of the tracked object. The motion model is represented as:

$$\begin{bmatrix} s \\ n \\ v \\ \zeta \\ \dot{\zeta} \end{bmatrix}_k = \begin{bmatrix} s_{k-1} + \frac{2}{\xi_{k-1}} v_k \sin\left(\frac{\xi_{k-1}\delta t}{2}\right) \cos(\zeta_{k-1} + \frac{\xi_{k-1}\delta t}{2}) \\ n_{k-1} + \frac{2}{\xi_{k-1}} v_k \sin\left(\frac{\xi_{k-1}\delta t}{2}\right) \sin(\zeta_{k-1} + \frac{\xi_{k-1}\delta t}{2}) \\ v_{t-1} \\ \zeta_{k-1} + \dot{\zeta}_{k-1}\delta t \\ \dot{\zeta}_{k-1} \end{bmatrix} \quad (9)$$

The length (L_{k-1}) and Width (W_{k-1}) are kept constant through prediction step. The process noise (ω_k), assumed to be Gaussian and additive, is also incorporated in the motion model. The motion model can hence be represented by the Equation (10).

$$x_k = f(x_{k-1}) + g(\omega_k) \quad (10)$$

where, $f(x_{k-1})$ represents the motion model given by Equation (9) and $g(\omega_k)$ process noise. $\mu_{k|k-1}^i$ and $P_{k|k-1}^i$, which are the mean and covariance of the weighted

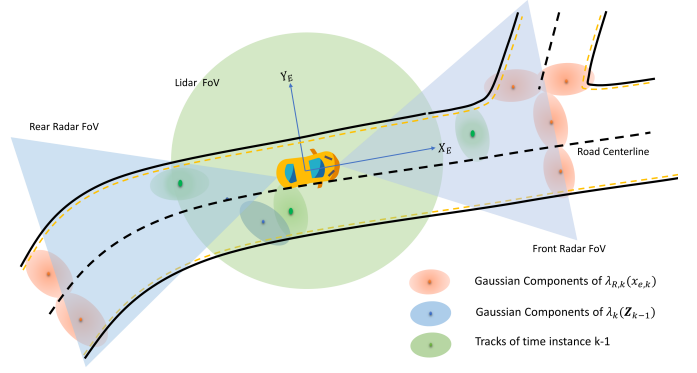


Fig. 5: Birth Components based on sensor FoV and Road Geometry and measurements of previous time instance.

Gaussian components are computed using the unscented transformation. Furthermore, the weight of each Gaussian component, $w_{k|k-1}^i$ is scaled by the Probability of Survival P_S , which is assumed to be constant and state independent value in our work.

2) *Addition of Birth Components*: We exploit information of road network and bounds to define the birth intensity, $\lambda_{b,k}(x)$. In addition to this, those measurement points which are very far from the preexisting tracks in terms of Mahalanobis Distance are also used to initialize new births components. Fig. 5 illustrates the Gaussian components used to define these birth process.

$$\lambda_{b,k}(x) = \lambda_{R,k}(x_{e,k}) + \lambda_k(\mathbf{Z}_{k-1}) \quad (11)$$

Ego vehicle localization in the road, $x_{e,k}$, the road geometry and the FoV definition of sensors are used to define the road based Gaussian components, $\lambda_{R,k}(x_{e,k})$. Disregarded measurement clusters from last time instance \mathbf{Z}_{k-1} are used to define intensity corresponding to measurements, $\lambda_k(\mathbf{Z}_{k-1})$. State representation in curvilinear road coordinates further facilitates this definition of the birth components, for example, when the road bifurcates in the scenario illustrated in Fig. 5.

B. Filter Update

Prediction step outputs $p(\mathbf{X}_k|\mathbf{Z}_{k-1})$, which is multi-object Poisson distribution of Poisson RFS $\mathbf{X}_k|\mathbf{Z}_{k-1}$. The PHD of this distribution, calculated by using equation (8) is also approximated as a Gaussian mixture. A measurement model is developed to update these intensity parameters, which outputs predicted measurement points in cartesian coordinates of VRF from the state in curvilinear road coordinate.

1) *Measurement Model*: A single object measurement model is discussed in this section, which is later integrated into the tracking iteration for the multi-object tracking process. Let $\mathbf{C}_k = \{z_l\}_{l=1}^{|\mathbf{C}_k|}$ be the set of measurements generated from an object $x_{i,k}$ clustered together. A measurement likelihood $p(\mathbf{C}_k|x_{i,k})$ needs to be calculated

to define the measurement update step. Measurement points within this cluster are assumed to be independent of each other, hence the measurement likelihood can be computed as:

$$p(\mathbf{C}_k|x_{i,k}) = \prod_{l=1}^{|\mathbf{C}_k|} p(z_l|x_{i,k}) \quad (12)$$

A measurement point z_l can be assumed to be generated from a measurement generating point y_l with some noise e , i.e. $z_l = y_l + e$. Removing the point indices for clearer representation, the likelihood of point z is expressed as:

$$p(z|x) = \int p(z|y)p(y|x)dy \quad (13)$$

and approximated by Gaussian mixture as:

$$p(z|x) = \sum_{i=1}^N w^i \mathcal{N}(z : y^i(x), R^i), \sum_{i=1}^N w^i = 1 \quad (14)$$

where, N is the number of Gaussian components used to approximate the likelihood. This value is equal to the numbers of measurement points in the given cluster, M_k^c under consideration for track state update. It is also assumed that each measurement generating point, y generates a single measurement point only. Development of this measurement model requires the computation of the measurement generating points \mathbf{Y}_k from the known object state $x_{i,k}$ and obtained measurement cluster \mathbf{C}_k . Since the object state is in curvilinear coordinate(s, n), the first step of this process would be to compute the state in cartesian coordinate (x, y) in VRF (E).

Coordinate Conversion: This conversion process is illustrated in the Fig. 6 and pseudocode in Table 1. The pseudocode algorithm provides the conversion for the positional values of the state(s, n to x, y), the velocity value remains the same while the yaw and yaw rate conversions are given by equations (15).

$$\begin{aligned} \psi &= \zeta + \theta_o - \psi_e \\ \dot{\psi} &= \dot{\zeta} + \kappa_o \cdot v \cdot \cos(\zeta) - \dot{\psi}_e \end{aligned} \quad (15)$$

where, κ_o is the curvature of the road component corresponding to the obstacle position. ψ_e and $\dot{\psi}_e$ are yaw and yaw rate of ego vehicle. The output of this conversion step is the object state in VRF (E), $x_{i,k} = [x, y, v, \psi, \dot{\psi}, L, W]_{i,k}^T$. It is important to notice that the yaw and yaw rate are relative values in VRF.

Computing Measurement Generating Points: In the second stage, the measurement generating points are computed from the object state by closely following the model developed by [10] with modifications to include velocity components. The process is discussed here for completeness. In [10], authors only use measurements obtained from the Lidar sensor to develop the measurement model, however, this work further extends it by using the measurement points obtained from sensor fusion module from Lidar and Radar sensors. To develop

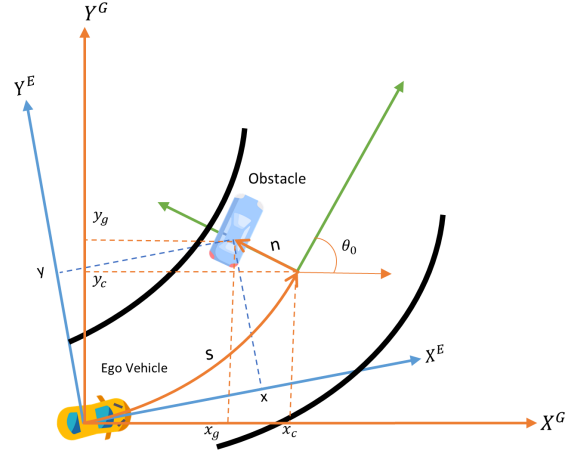


Fig. 6: Representation of the curvilinear and cartesian coordinate system in Vehicle Reference Frame

Algorithm 1 Curvilinear To Cartesian :Euler Model

Input: (x, θ_r, x_e , EgoPosition)

Computations:

$\bar{\theta} = \theta_r(\text{EgoPosition})$

Discretize $s_a \leftarrow 0 : \delta s : s$

$N \leftarrow \text{length}(s_a)$

$\theta_a \leftarrow \bar{\theta}(1)s_a^3 + \bar{\theta}(2)s_a^2 + \bar{\theta}(3)s_a + \bar{\theta}(4)$

$\theta_o \leftarrow \theta_a(\text{end})$

Position Coordinate Conversion

for $n=2$ to N **do**

$x^n \leftarrow x^{n-1} + \delta s \cos(\theta_a(n))$

$y^n \leftarrow y^{n-1} + \delta s \sin(\theta_a(n))$

end for

$x_c \leftarrow x^N$

$y_c \leftarrow y^N$

$x_g \leftarrow x_c + n \cos(\frac{\pi}{2} - \theta_o)$

$y_g \leftarrow y_c + n \sin(\frac{\pi}{2} - \theta_o)$

$\zeta \leftarrow x_e(2)$

$\psi \leftarrow x_e(3) \leftarrow \text{Ego Vehicle Yaw Angle}$

$x \leftarrow \cos(\psi)x_s - \sin(\psi)y_s$

$y \leftarrow \sin(\psi)x_s + \cos(\psi)y_s + x_e(1) \cos(\zeta)$

Output: (x, y)

this model, the Lidar sensor is taken as a primary sensor, i.e., measurement points are used to update the object state only when measurements from Lidar are available. The measurement points in the cluster \mathbf{C}_k are sorted in counter-clockwise direction based on their bearing angle, and predicted measurement generating points are similarly arranged, eliminating the need to perform the association between these points. The likelihood then becomes,

$$p(\mathbf{C}|x) = \prod_{l=1}^{|\mathbf{C}|} \mathcal{N}(z_l; y_l(x), \sigma_r^2 I_2) = \mathcal{N}(z_c; y_c(x), S_c) \quad (16)$$

where, z_C is measurement points vertically concatenated and $y_C(x)$ is predicted measurement generating points vertically concatenated as well. S_C is the measurement innovation matrix. The model is developed with an assumption that the sensor can see a maximum of two sides of the rectangular object at a time instance. Based on this assumption, two models are developed.

Single Sided Measurements: For the state x_i , let $\alpha_1, \alpha_2, \alpha_3$ and α_4 be the angles made by normal of the sides of the rectangle. Let β represent the angle made by the vector from the first to last measurement point in the cluster. The measurements are associated to i^{th} side of the rectangle, which satisfy the condition:

$$i_{min} = \operatorname{argmin}_i |\alpha_i - \beta + \pi/2| \quad (17)$$

The measurement generating points are computed to be spread along the associated side based on the extent of the spread of the measurement points in C_k .

Double Sided Measurements: If the measurements in C_k are deemed to be two sided measurements, a corner index is computed to separate the measurements into two single sided measurements. The corner index, n is computed by least squares fitting lines. By doing so, C_k is split into two sub sets, $C_{k,1} = \{z_l\}_{l=1}^n$ and $C_{k,2} = \{z_l\}_{l=n+1}^{|C_k|}$. Now, the single sided model is applied to each sub set to obtain the predicted measurement generating points.

2) *State Update:* The posterior PHD is computed as:

$$D_{k|k}(x) = D_{k|k}^{ND}(x) + \sum_{C_k \in \mathbf{Z}_k} D_{k|k}^D(x, C_k) \quad (18)$$

where, the PHD, $D_{k|k}^{ND}(x)$ is computed for the undetected objects, while $D_{k|k}^D(x, C_k)$ is computed for detected objects using the measurement clusters which are the output of the sensor fusion module. The PHD corresponding to the detected objects is given by:

$$D_{k|k}^D(x, C_k) = \sum_{i=1}^{N_{k|k-1}} w_{k|k}^{i, C_k} \mathcal{N}(x : \mu_{k|k}^{i, C_k}, P_{k|k}^{i, C_k}) \quad (19)$$

To compute the posterior parameters, $w_{k|k}^i$, $\mu_{k|k}^i$ and $P_{k|k}^i$ for the cluster C_k , update recursion of UKF is used. What follows is the update of the Gaussian components of the predicted density with the a measurement cluster C_k ; it should be noted that the measurement points at time instance k , \mathbf{Z}_k would be divided into multiple cluster as an output of the clustering algorithm discussed in Section III.

The predicted measurement mean ($y_{k|k-1}^{i,C}$) and Innovation covariance matrix ($S_{k|k-1}^{i,C}$) for Gaussian component i and cluster C_k can be calculated by using the Equations (20) and (21).

$$y_{k|k-1}^{i,C} = \sum_{\alpha=1}^{2n+1} w_{\alpha} \zeta_{\alpha,k|k-1}^{i,C} \quad (20)$$

$$S_{k|k-1}^{i,C} = \sum_{\alpha=1}^{2n+1} w_{\alpha} [\zeta_{\alpha,k|k-1}^{i,C} - y_{k|k-1}^{i,C}] [\zeta_{\alpha,k|k-1}^{i,C} - y_{k|k-1}^{i,C}]^T + R_C \quad (21)$$

where, R_C is the measurement noise covariance matrix obtained as $R_C = \text{blkdiag}(R_k, R_k, \dots, R_k)$. Definition of R_k depends on the type of the measurement cluster C_k , whether it has standalone Lidar points or the fused points form Lidar and Radar sensors. The noise for Lidar only points consider the noise between the measurement generating points and the measurement in spatial space while fused ones also consider the radar noise for the velocity components. $\zeta_{\alpha,k|k-1}^{i,C}$ are the predicted measurement sigma points computed by passing state sigma points, $\chi_{\alpha,k|k-1}^i$ through the measurement model discussed earlier. Finally, the cross correlation matrix between the predicted state and predicted measurement is calculated in Equation (22).

$$T_{k|k-1}^{i,C} = \sum_{\alpha=1}^{2n+1} w_{\alpha} [\chi_{\alpha,k|k-1}^i - \mu_{k|k-1}^i] [\zeta_{\alpha,k|k-1}^{i,C} - z_{k|k-1}^{i,C}]^T \quad (22)$$

Next, the Kalman gain can be calculated using this cross correlation matrix as:

$$K_{k|k-1}^{i,C} = T_{k|k-1}^{i,C} (S_{k|k-1}^{i,C})^{-1} \quad (23)$$

and the predicted mean of Gaussian component i is updated by measurement cluster C_k as:

$$\mu_{k|k}^{i,C} = \mu_{k|k-1}^i + K_{k|k-1}^{i,C} (z_k^C - y_{k|k-1}^{i,C}) \quad (24)$$

while, the covariance matrix is updated as:

$$P_{k|k}^{i,C} = P_{k|k-1}^i - (K_{k|k-1}^{i,C}) S_{k|k-1}^{i,C} (K_{k|k-1}^{i,C})^T \quad (25)$$

the weight of the each Gaussian component is also updated as:

$$w_{k|k}^{i,C} = \frac{P^D w_{k|k-1}^i \Gamma^i L^{i,C}}{\delta_{|C|,1} + \sum_{i=1}^{N_{k|k-1}} \Gamma^i P^D w_{k|k-1}^i L^{i,C}} \quad (26)$$

where, $L^{i,C}$ is computed using Equation (27):

$$L^{i,C} = \mathcal{N}(z_k^C : y_{k|k-1}^{i,C}, S_{k|k-1}^{i,C}) \prod_{z_k \in C_k} \frac{1}{\lambda_k C_k(z_k)} \quad (27)$$

and Γ^i is computed as:

$$\Gamma^i = e^{-\gamma^i} (\gamma^i)^{|C|} \quad (28)$$

Standard mixture reduction techniques like: pruning, merging and capping are employed to ensure the tractability of the algorithm.

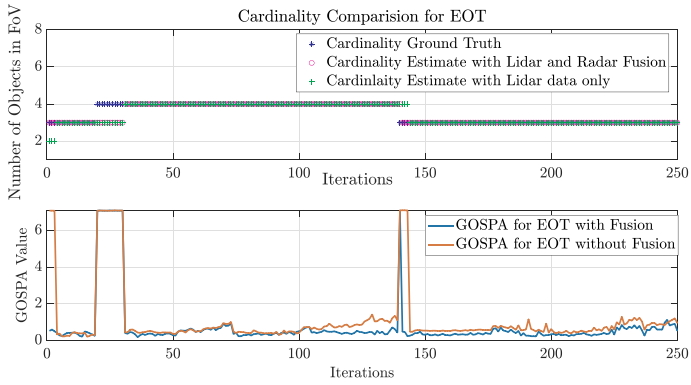


Fig. 7: Cardinality and GOSPA values computed for estimated states for multi-object tracking scenario

VII. ALGORITHM VALIDATION

The proposed EOT algorithm provides estimates of the kinematic and extent state of objects within road bounds in curvilinear road coordinates. The state estimates are computed from the measurements obtained from Lidar and Radar sensors. Precomputed HD road map is integrated into the EOT GM-PHD filtering routine. The algorithm is validated through simulation and experimental data.

A. Simulation Validation

A multi-object scenario is developed in Matlab Driving Scenario Designer application to validate the algorithm. Here, we present one of the scenarios with high road curvature and obstacles performing various maneuvers. The simulated scenario is made available in GIF format here. This visualization contains all the sensor data along with the obtained state estimate in curvilinear coordinate.

Fig. 7 illustrates how the algorithm can consistently provide obstacle state estimates with reasonable accuracy. The scenario is simulated with two different setups, one using both Lidar and Radar sensor while another with Lidar data only. Results in Fig. 7 show that EOT employing sensor fusion between lidar and radar sensor performs better in relation to the localization error component of the Generalized Optimal sub-pattern assignment (GOSPA) metric, [32]. The peaks in the GOSPA values are observed due to occlusion in the driving scenario and error in ground truth cardinality estimation for defined FoV in the simulation environment.

Furthermore, we observed that the obstacle state representation in curvilinear coordinates enables for better accuracy in obstacle state estimation. In Fig. 8, the GOSPA values are computed for two different approaches for single object tracking scenario, in first approach, the obstacles states are represented in cartesian coordinates while in second approach states are represented in the curvilinear coordinates. For comparison with the ground truth, the states computed in curvilinear

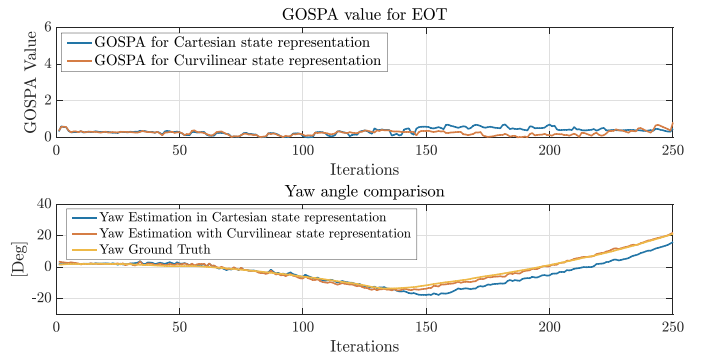


Fig. 8: Top: GOSPA values computed with state representation in cartesian and curvilinear coordinates Bottom: Yaw angles estimates obtained with state representation in cartesian and curvilinear coordinates.

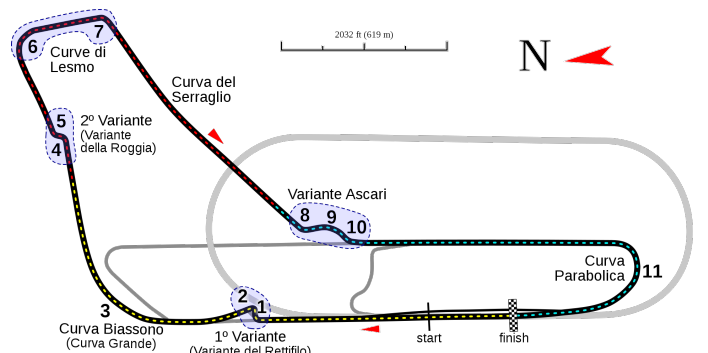


Fig. 9: Experimental Test site, Monza Circuit

coordinates are converted to cartesian coordinates. It can be observed that the state estimation in curvilinear coordinates allows for better accuracy in terms of object localization and especially computation of the yaw angle. (GIF for the scenario available here.).

B. Experimental Validation

For safety reasons and to acquire an accurate ground truth, the algorithm is validated using data collected during experimental campaigns at Monza Eni Circuit, as shown in Fig. 9. The experiment consists of the ego prototype and another tracked vehicle installed with RTK corrected GPS (Real Time Kinematic Global Positional System). The tracked vehicle used in the experiments is FIAT Talento, with size $4.8m \times 2m \times 2m$. The algorithm is analyzed by dividing a single lap run at the Monza circuit into various segments based on mutual maneuvers between the ego vehicle and the obstacle and the degree of the road curvature. Given that the output of the Lidar preprocessing is at 10Hz, the object state estimation routine runs at 10Hz on a soft real-time-based Robot Operating System (ROS) system [33].

The scenario under investigation here is taken from the “Variante Del Roggia” segment of the track. We choose this scenario because of its significantly varying

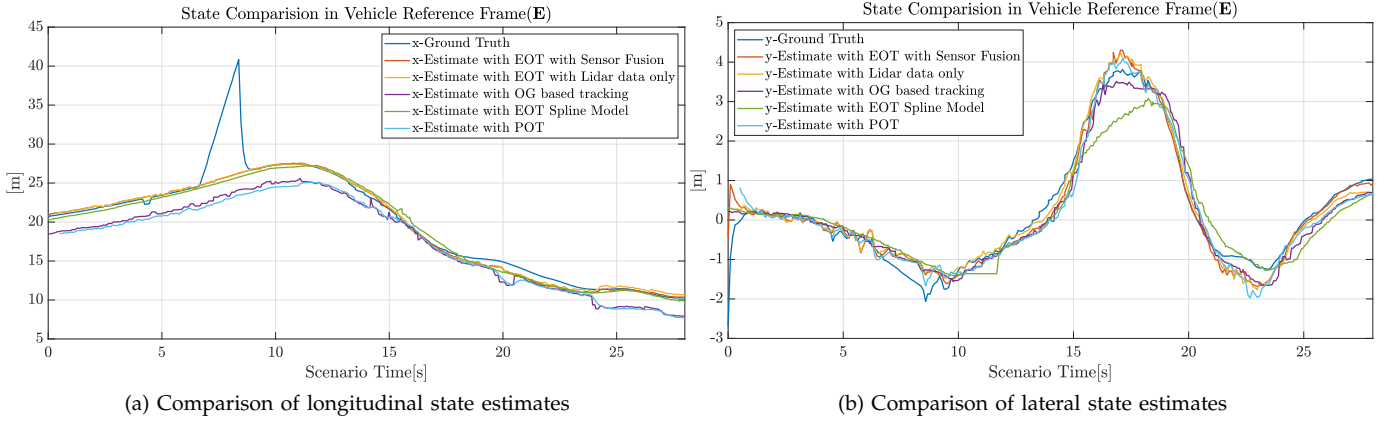


Fig. 10: Positional state estimates obtained from different tracking approaches with ground truth for Scenario 1

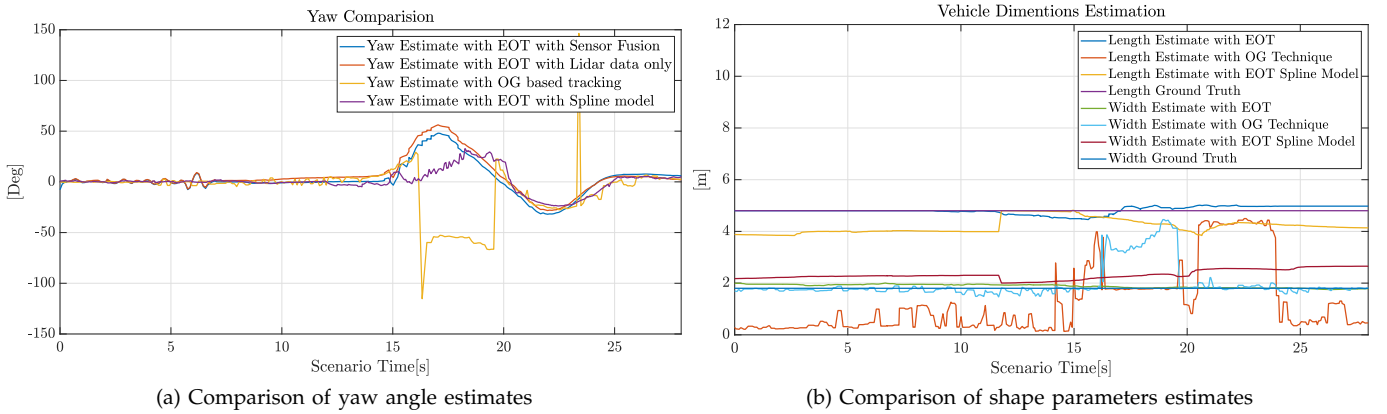


Fig. 11: Yaw angle and shape parameters estimates obtained from different tracking approaches with ground truth for Scenario 1

road curvature, as can be observed in Fig. 9. The results of the experimental scenario are made available in GIF format here for further clarity. Additionally, results corresponding to the scenario at "Variante Del Rettifilo" are attached here as well. In this scenario under analysis, an object-following maneuver is developed while the ego vehicle sees single and double sides of the object, necessitating the switch between single side and double sides measurement models. The velocities of the vehicles are around $18m/s$ during this experiment.

We performed a comparative study with various recent algorithms to validate our proposed approach. In our previous work, [25], a Point Object Tracking (POT) implementation of Global Nearest Neighbour (GNN) filter with UKF estimator, object states are computed in curvilinear coordinates. However, with POT assumption, no object's shape and yaw angle estimates are performed. The algorithm is able to provide object estimates at $20Hz$. State estimates using Occupancy Grid (OG) based tracking technique, [5] provides object shape as well as yaw angle estimate and operates at $10Hz$. A

GMPHD filtering recursion with UKF estimator using spline measurement model proposed by [34] is also developed for validation of the proposed approach. States estimates are obtained at a frequency of $6Hz$. Furthermore, the results obtained with two different setups of the proposed algorithm, first using Lidar data only as input and second with fused data from both Lidar and Radar sensors are compared.

In Fig. 10, the results of this scenario are illustrated. These comparisons are made in VRF (E) based on cartesian coordinates because of the unavailability of ground truth in curvilinear coordinates. Estimated object states obtained from our proposed algorithm and [25] are converted to cartesian coordinates for comparing with the ground truth. The ground truth values in the cartesian coordinates are obtained from an RTK-GPS sensor installed in the tracked vehicle. The proposed algorithm outperforms the other algorithms in terms of positional accuracy of the estimated state. A root mean square error of $0.401m$ and $0.51m$ were observed respectively for the setup with and without sensor fusion for the proposed

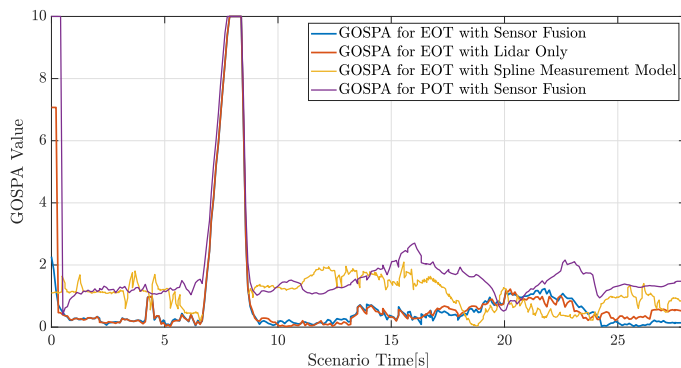


Fig. 12: Comparison of GOSPA metric for different algorithms

algorithm.

Fig. 11a illustrates the results obtained from different algorithms for yaw angle estimates. Due to the unavailability of the ground truth values for the yaw angle, results from different algorithms are compared among themselves. Visual assessment of the scene showed that the proposed algorithm can estimate the yaw angle with reasonable accuracy. The fluctuation of the yaw estimate from the OG-based technique is due to the nature of the detection obtained from the OG-based detector. As illustrated here, the algorithm cannot differentiate between the leading side of the bounding box and can provide wrong estimates. For the Spline measurement model-based EOT tracker, yaw angle estimates were observed to be erroneous due to the rapidly changing curvature of the road and only a single side of the tracked object being observed during this maneuver. Due to the integration of the road model and object state representation in curvilinear coordinates, estimation from the proposed algorithm is significantly better than the presented alternatives. Fig. 11b validates the extent estimates obtained from the proposed algorithm. The width estimated from the proposed EOT algorithm is confirmed with the outcome of the OG-based approach and the ground truth (approximately $2m$). Spline measurement model-based EOT tracker over estimates width at multiple instances. The length estimates from the OG-based approach, which can compute the length observed by the Lidar sensor only, strongly underestimates the length at multiple time instances. The proposed EOT algorithm's length estimates closely match the ground truth (approximately $4.8m$) and outperform the spline measurement model-based EOT algorithm estimates.

The proposed approach is presented as a filtering algorithm and hence is validated using the GOSPA metric. The GOSPA values obtained for the various algorithms are illustrated in the Fig. 12. We observe that the use of sensor fusion for EOT provides better state estimates, and state representation in curvilinear coordinates reduces the localization error of the tracked

object. The algorithm can also provide state estimates with reasonable GOSPA values except for the instances in between 7s to 9s. The GOSPA performs poorly due to sudden drift in ground truth values obtained from the track GPS sensor. This drift can also be observed in the Fig. 10a.

VIII. CONCLUSIONS AND FUTURE WORKS

The key contribution of this work is to provide state estimation of dynamic objects within road bounds in curvilinear road coordinates. Extended object representation of these objects enables estimation of the kinematic and extent state. A GM-PHD Filter for EOT with a UKF estimator is used for this estimation process. A hybrid sensor fusion architecture consisting of Lidar and Radar is employed to obtain information regarding these objects. The algorithm is validated through simulation, and experimental data collected from the test runs at the Monza Eni. Circuit. A comparative study with the Occupancy-Grid-based tracking technique, Point Object Tracking in curvilinear coordinates, and EOT with spline-based measurement model is explored to validate the object state estimates obtained using the EOT algorithm. We also demonstrate that the state representation in curvilinear coordinates increases the accuracy in yaw angle estimate in high road curvature scenarios.

Future work will focus on generalizing the proposed approach, removing constraints on the object's shape. Tracking objects other than vehicles requires a different approach than predefined rectangular shapes to compute measurement generating points. Gaussian Processes, Splines models, etc., can be explored to develop star convex-shaped object models that provide free-form object shape.

REFERENCES

- [1] X. Weng, J. Wang, D. Held, and K. Kitani, "3d multi-object tracking: A baseline and new evaluation metrics," 2020.
- [2] H. kuang Chiu, A. Prioletti, J. Li, and J. Bohg, "Probabilistic 3d multi-object tracking for autonomous driving," *ArXiv*, vol. abs/2001.05673, 2020.
- [3] N. Benbarka, J. Schröder, and A. Zell, "Score refinement for confidence-based 3d multi-object tracking," 2021.
- [4] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," 2019.
- [5] S. Mentasti, M. Matteucci, S. Arrigoni, and F. Cheli, "Two algorithms for vehicular obstacle detection in sparse pointcloud," 2021.
- [6] K. Granstrom, M. Baum, and S. Reuter, "Extended object tracking: Introduction, overview and applications," 2017.
- [7] R. Lot and F. Biral, "A curvilinear abscissa approach for the lap time optimization of racing vehicles," vol. 19, 08 2014.
- [8] J. Hudecek and L. Eckstein, "Improving and simplifying the generation of reference trajectories by usage of road-aligned coordinate systems," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014, pp. 504–509.
- [9] K. Thormann, M. Baum, and J. Honer, "Extended target tracking using gaussian processes with high-resolution automotive radar," in *2018 21st International Conference on Information Fusion (FUSION)*, 2018, pp. 1764–1770.

- [10] K. Granström, S. Reuter, D. Meissner, and A. Scheel, "A multiple model phd approach to tracking of cars under an assumed rectangular shape," in *17th International Conference on Information Fusion (FUSION)*, 2014, pp. 1–8.
- [11] M. P. Muresan, I. Giosan, and S. Nedeveschi, "Stabilization and validation of 3d object position using multimodal sensor fusion and semantic segmentation," *Sensors*, vol. 20, no. 4, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/4/1110>
- [12] J. Nie, J. Yan, H. Yin, L. Ren, and Q. Meng, "A multimodality fusion deep neural network and safety test strategy for intelligent vehicles," *IEEE Transactions on Intelligent Vehicles*, vol. 6, no. 2, pp. 310–322, 2021.
- [13] L. Hammarstrand, L. Svensson, F. Sandblom, and J. Sorstedt, "Extended object tracking using a radar resolution model," *IEEE Transactions on Aerospace and Electronic Systems - IEEE TRANS AEROSP ELECTRON SY*, vol. 48, pp. 2371–2386, 07 2012.
- [14] K. Granström, M. Fatemi, and L. Svensson, "Gamma gaussian inverse-wishart poisson multi-bernoulli filter for extended target tracking," in *2016 19th International Conference on Information Fusion (FUSION)*, 2016, pp. 893–900.
- [15] M. Baum and U. D. Hanebeck, "Extended object tracking with random hypersurface models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 1, pp. 149–159, 2014.
- [16] C. Knill, A. Scheel, and K. Dietmayer, "A direct scattering model for tracking vehicles with high-resolution radars," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 298–303.
- [17] M. Schuster, J. Reuter, and G. Wanielik, "Probabilistic data association for tracking extended targets under clutter using random matrices," in *2015 18th International Conference on Information Fusion (Fusion)*, 2015, pp. 961–968.
- [18] N. Wahlström and E. Ozkan, "Extended target tracking using gaussian processes," *IEEE Transactions on Signal Processing*, vol. 63, p. 4165–4178, 08 2015.
- [19] A. Scheel and K. Dietmayer, "Tracking multiple vehicles using a variational radar model," pp. 3721–3736, Oct 2019.
- [20] Y. Xia, P. Wang, K. Berntorp, P. Boufounos, P. Orlik, L. Svensson, and K. Granström, "Extended object tracking with automotive radar using learned structural measurement model," in *2020 IEEE Radar Conference (RadarConf20)*, 2020, pp. 1–6.
- [21] P. Cudrano, S. Mentasti, M. Matteucci, M. Bersani, S. Arrigoni, and F. Cheli, "Advances in centerline estimation for autonomous lateral control," 02 2020.
- [22] J. Kim, K. Jo, W. Lim, M. Lee, and M. Sunwoo, "Curvilinear-coordinate-based object and situation assessment for highly automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, pp. 1–17, 06 2015.
- [23] E. Héry, S. Masi, P. Xu, and P. Bonnifait, "Map-based curvilinear coordinates for autonomous vehicles," 10 2017, pp. 1–7.
- [24] E. Héry, P. Xu, and P. Bonnifait, "Along-track localization for cooperative autonomous vehicles," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 511–516.
- [25] M. Bersani, S. Mentasti, P. Dahal, S. Arrigoni, M. Vignati, F. Cheli, and M. Matteucci, "An integrated algorithm for ego-vehicle and obstacles state estimation for autonomous driving," *Robotics and Autonomous Systems*, vol. 139, p. 103662, 2021.
- [26] K. Jo, M. Lee, J. Kim, and M. Sunwoo, "Tracking and behavior reasoning of moving vehicles based on roadway geometry constraints," *IEEE Transactions on Intelligent Transportation Systems*, vol. PP, pp. 1–17, 09 2016.
- [27] Y. Yoon, T. Kim, H. Lee, and J. Park, "Road-aware trajectory prediction for autonomous driving on highways," *Sensors*, vol. 20, no. 17, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/20/17/4703>
- [28] S. Arrigoni, F. Braghin, and F. Cheli, "Mpc path-planner for autonomous driving solved by genetic algorithm technique," 02 2021.
- [29] S. Arrigoni, S. Mentasti, F. Cheli, M. Matteucci, and F. Braghin, "Design of a prototypical platform for autonomous and connected vehicles," in *2021 AEIT International Conference on Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, 2021, pp. 1–6.
- [30] D. Zermas, I. Izzat, and N. Papanikolopoulos, "Fast segmentation of 3d point clouds: A paradigm on lidar data for autonomous vehicle applications," in *IEEE International Conference on Robotics and Automation*, 2017.
- [31] K. Granstrom, C. Lundquist, and O. Orguner, "Extended target tracking using a gaussian-mixture phd filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 3268–3286, 2012.
- [32] A. S. Rahmathullah, A. F. Garcia-Fernandez, and L. Svensson, "Generalized optimal sub-pattern assignment metric," *2017 20th International Conference on Information Fusion (Fusion)*, Jul 2017. [Online]. Available: <http://dx.doi.org/10.23919/ICIF.2017.8009645>
- [33] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng *et al.*, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.
- [34] H. Kaulbersch, J. Honer, and M. Baum, "A cartesian b-spline vehicle model for extended object tracking," in *2018 21st International Conference on Information Fusion (FUSION)*, 2018, pp. 1–5.