Kalman filtering aspects in camera and deep learning based tracking for traffic monitoring

Nils Kornfeld
Institute of Transportation Systems
German Aerospace Center (DLR)
Braunschweig, Germany
nils.kornfeld@dlr.de

Andreas Leich
Institute of Transportation Systems
German Aerospace Center (DLR)
Berlin, Germany
andreas.leich@dlr.de

Michael Roth
Institute of Transportation Systems
German Aerospace Center (DLR)
Braunschweig, Germany
m.roth@dlr.de

***Abstract*—In multiple object tracking applications for traffic monitoring the underlying algorithms often use rectangular, axis-aligned bounding boxes from deep-learning based object detection systems as a measurement input. Often the association of the measurements to trajectories is performed in the image domain, where after for every bounding box an already associated pseudo-measurement in a world coordinate system is estimated, which is finally used as a measurement input to a Kalman Filter. In contrast to this approach this article examines a multiple object tracking system with a measurement model which maps the estimated state of objects in world coordinates to the aforementioned rectangular bounding boxes in an image coordinate system. In addition the choice of the state vector elements modelled to represent the vehicles is shown and discussed. The approach presented in this article allows for association founded in physical reality, the estimation of the spacial dimensions of tracked objects and avoids shortcomings of a two-staged approach with association in the image coordinate frame.**

***Index Terms*—Kalman filters, state estimation, deep learning, traffic monitoring, measurement models**

Fig. 1. Example of a camera system operated by DLR. Detail of a camera mast in Braunschweig at the research intersection. The installations carry further hardware for e.g. communication.

I. Introduction

The DLR Institute of Transportation Systems operates stationary camera networks in urban and highway environments in order to perform research on the interaction between different road users, the assessment of critical situations [1], and connected autonomous vehicles (CAV) [2]. In both use cases, correct 3D position estimates for the visible 2D objects in the camera are crucial. The accuracy requirements for these tasks vary depending on the use case. An examples of a camera system operated by DLR is given in Fig. 1. More information on these systems can be found in [3]

The assessment of whether or not a situation is critical is often based on the time to collision (TTC) surrogate safety measure. In [4] it is shown, that the required accuracy of position estimation should be in the range of $0.1 \ldots 1.0\,\mathrm{m}$ if the false positive rate of the detector should not exceed the true positive rate for a TTC threshold of $0.5\,\mathrm{s}$. Knowing precisely where a road user is in a local Cartesian coordinate system like Universal Transverse Mercator (UTM) helps automating the perception of a traffic scene.

There are several sensors that can be considered for traffic monitoring. Cameras are ubiquitous and low in cost. Camera data is rich in information and easy to interpret by humans. In order to convert the unstructured information into measurements that can be further processed, deep neural networks (NN) for object detection are a timely choice. In the simplest case NN detectors draw rectangles around objects in the image plane. To estimate vehicle trajectories in a local Cartesian coordinate frame multiple target tracking methods are applied. These commonly incorporate Kalman filters for state estimation.

The scene depicted in Fig. 2 was recorded during a measurement campaign at Test Bed Lower Saxony, an open research and development platform constructed by DLR. The car moving towards the camera in this scene recorded a reference trajectory in a world coordinate system which is used in the evaluation part of this article. The reference trajectory was composed of Global Navigation Satellite System (GNSS) data and corrected with Real Time Kinematic (RTK) [5].

In this article we present a multiple object tracking system for traffic monitoring that uses a physically plausible motion model for filtering and association, which also estimates the physical extent of vehicles. The system is robust to changing camera perspectives, allows for multiple sensors and correctly computes the measurement emission of vehicles only partially visible in the image. As an input only axis-aligned bounding boxes on monocular camera images, generated by NN detec-

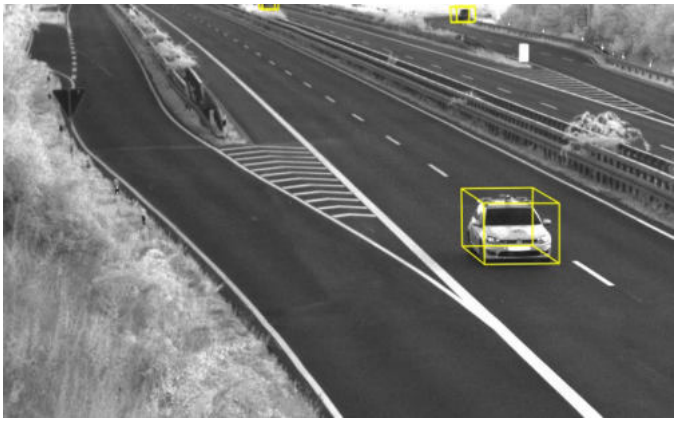

Fig. 2. An example perspective recorded at Test Bed Lower Saxony. An RTK-corrected reference trajectory recorded by the car moving towards the camera is used for the evaluation in this article.

tors are used. We also present an approach to optimize the camera calibration needed for this tracking system.

Sec. II provides pointers to the relevant tracking and deep learning literature. Sec. III presents a typical tracking pipeline for traffic monitoring. The choice of the state vector and the motion model are discussed in Sec. IV. The measurement model equations are presented in Sec. V. Challenges and real world implications are discussed in Sec. VI. Following the experiments in Sec. VII some concluding remarks are given.

II. Background and related work

There are different communities that work on tracking problems. Early work has focused on radar for air traffic control problems. Distant aircraft are modeled as point objects. Radar detectors provide range and bearing measurements. False alarms and missed detections are common. A comprehensive source is [6]. The individual tracks are commonly processed using Kalman filter (KF) variants [7], [8], often EKF and UKF [9]. More recently, extended target tracking [10] has been investigated. Here, each object can cause multiple detections in the measurement domain. An elegant aspect is that much of the underlying theory of the above references is agnostic to the employed sensors or dynamic models. This facilitates its transfer between domains.

The system presented in this article is methodically closer to point tracking applications, because every object generates at most one measurement, although the object extents are considered in the tracking process.

In the computer vision community, tracking is often understood as following a given image patch (template) over consecutive video frames in the image plane. In general, this involves evaluating correlation in the sense of calculating the similarity between the image patch in a reference image at time t_k (template) and the moved image patch at t_{k+1}. Early approaches involved gradient descent on a cost function over some parameter space, e.g. the sum of squared pixel grey-value differences [11], [12]. Besides linear translation, also zoom, rotation and affine transformation between the template and current image patches can be used for modelling motion. The demand for an image representation that is robust to changes in gray level and deformations in general led to approaches such as the Histogram of oriented Gradients representation (HoG) [13]. Kernel methods, such as support vector machines (SVM) [14] allow to learn a representation of the tracked image patch that is stable under arbitrary deformation of the object. Therefore, popular visual tracking algorithms do correlation based tracking with kernelized filters [15]. Kernelized methods are capable not only of successfully tracking image patches that change their appearance, but also of re-identifying image patches after being fully occluded.

While the HoG representation is a human engineered solution, finding the feature mapping is part of the learning process in deep learning approaches. Since the resurgence of Artificial Neural Networks in image processing due to the success of AlexNet [16] in the ImageNet large scale visual recognition challenge [17] a lot of tracking work in the computer vision community has shifted to deep-learning based tracking systems. These systems can either be end-to-end deep learning architectures like GOTURN [18], [19] or conduct only the object detection task with a deep NN and use a more traditional tracking pipeline with KF variants as introduced in the first paragraph of this section like SORT [20] and DeepSORT [21]. Every one of these methods from the computer vision community described so far conducts the tracking of an image patch in the image plane. The resulting trajectories have to undergo an additional coordinate transform from the image coordinate system to a local Cartesian coordinate frame. A prerequisite for this transformation is sufficient information about the inner and exterior orientation of the camera used to acquire the images [19], [22]. Further, some estimation procedure for the pose of the vehicle in the local Cartesian coordinate frame is needed. Fitting the convex hull of a projected cuboid to the semantic mask of the vehicle has been reported to yield acceptable accuracy in this regard [23]–[25]. In principle, deep learning based object pose estimation in 6 degrees of freedom [26] can tackle this problem.

Once object and track hypotheses exist, linear programming techniques have been demonstrated to solve the K-Shortest paths optimization problem of assigning object ids to tracks and track fragments [27] in multiple target tracking. Employing this technique can be seen as a post processing approach. It proved to reduce ID-switches and track fragmentation on real-world tracking benchmarks significantly [28].

Multiple object tracking systems for traffic monitoring applications can be built as a two-staged approach as in [29]. The first stage of the system applies a tracker that solely works in the image space like SORT [20], [21]. In this initial stage the detections are each associated to an individual object ID, based on an intersection over union in a Kalman Filter framework [20], [21]. Because of this approach, the association is not based on a motion model based in physical reality like in more traditional KF based tracking methods. Another problem of applying SORT for the association of measurements to vehicles is that it uses assumptions that were tailored to the

tracking of pedestrians like a static aspect ratio for each object [20], which does not necessarily apply for vehicles typically present in traffic monitoring scenarios.

After the association the detections are projected to an estimated ground plane. Because the central point of a bounding box in the image is not necessarily projected to the center of a vehicle on the ground plane, some heuristics need to be applied to get a suitable estimate of the vehicle's location [29], [30]. The extensions of objects can not be determined by such a system and need to be supplied by prior knowledge about the average spatial dimensions of objects within the corresponding object class.

The approaches of this article treat the deep learning based object detection as the measurement in a KF based multiple target tracking framework, with the position of each tracked target defined in a Cartesian world coordinate frame. An implementation of such a system is *Urban Traffic Surveillance* (UTS) [30]. In contrast to UTS, our choice of the measurement function does not need any special treatment for objects only partially visible in the image. Because our measurement function is parameterized with the camera calibration data, using a moving camera as well as multiple sensors with different calibration data are possible. For measurement association we use a squared Mahalanobis distance metric to facilitate a statistically sound association with a gating threshold motivated by confidence intervals around the measurements [31].

III. Tracking pipeline

Multiple target tracking problems are best addressed in a modular fashion. In the simplified pipeline of Fig. 3 each block represents a software module that communicates with the other blocks via messages in a predefined protocol. In the block diagram only one camera is depicted, but the content of this paper is also applicable to a tracking pipeline with multiple cameras. The multiple target tracking system described here is based on [32]. The system described here is simplified and adapted to camera sensors instead of Radar.

The camera captures one video image where a deep-learning based object detection system recognizes objects. For the scope of this paper the specific architecture of the object detector is not relevant. For an overview on object detection algorithms refer to [33]. The output of the object detection is a set of rectangular, axis-aligned bounding boxes around the objects visible in the video frame. Each bounding box is treated as a single measurement.

In an association step each measurement is either assigned to a previously initiated track or used to initiate a new tentative track. Tentative tracks that do not match an incoming measurement are deleted. New tracks are initialized if two consecutive measurements are associated to a tentative track. The centers of the detections are projected on the estimated ground plane. These two points are sufficient to get an initial rough estimate of the vehicles heading angle ϕ_w and speed v_w. The initial estimate of the extensions of an object are read from a previously prepared lookup table, which contains average extents of vehicles based on their class (car, van, bus, lorry, ...). As argued in Sec. II the projected bounding box centers are not necessarily good estimates for the actual vehicle centers. Thus this preliminary state vector is optimized with the Nelder-Mead method [34] to best fit the current measurement. Assigned pairs and confirmed tentative tracks are forwarded to the filtering block. An optimized version [35] of the Kuhn-Munkres algorithm can be used for the association.

The filtering block contains a filter for each tracked object and estimates the object state x by recursively updating the parameters of a probability density function (PDF). The most common choice is a Gaussian distribution $\mathcal{N}(\hat{x}, \mathbf{P})$. The filtering algorithm is a KF variant that can handle the noise assumptions and nonlinearities, including the measurement function for NN detections of Sec. V. The state of each tracked object is represented by a mean state vector $\hat{x}_k$ and a state covariance $\mathbf{P}_k$ at timestep k. The specific configuration of $\hat{x}_k$ depends on the motion model used to predict the state at the next timestep $\hat{x}_{k+1}$. A reasonable state representation for a vehicle in a traffic context is shown in Sec. IV. The state vector resides in a local Cartesian coordinate frame, while the measurements are defined in pixel coordinates in the input image coordinate system. To enable the filtering algorithm to correct the tracked state, a measurement function which describes the image acquisition and object detection process must be known.

IV. State and Motion Model

For tracking vehicles in traffic scenarios it is sensible to model the motion of the objects of interest in a manner that is grounded in physical reality. Because the motion takes place in world coordinates, the state representation is also located in the world coordinate frame, with a position in Cartesian coordinates. In this case UTM-coordinates are used. In contrast to the position the velocity is represented in polar coordinates. Such a motion model is referred to as a coordinated turn model. This model describes the motion of road vehicles more accurately than a constant velocity model by facilitating the modeling of steering angle and thrust [9]. The choice of a suitable state vector is not only dependent on the motion or state transition model but also on the measurement model. The state vector needs to incorporate all information necessary to compute an emitted measurement from the state by the use of the measurement model, which is discussed in detail in Seq. V. Because the measurements are dependent on the extensions of an object in world coordinates, these need to be represented as well. The state vector $\vec{x}$ consists of a point location in 3D Cartesian coordinates, the speed v_w, a heading angle ϕ_w, the derivatives of speed and heading angle, as well as width, length and height of the vehicle w_v, l_v and h_v. The indices w and v denote that the respective values are either relative to the world coordinate frame or they are attributes of the vehicle which is represented by this vector.

$$\vec{x} = \begin{pmatrix} x_w & y_w & v_w & \phi_w & \dot{v}_w & \dot{\phi}_w & w_v & l_v & h_v \end{pmatrix}^\mathsf{T} \quad (1)$$

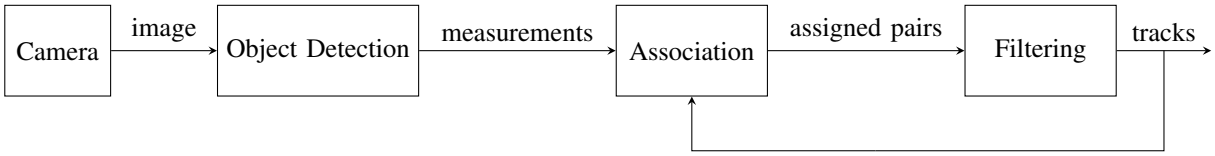

Fig. 3. A simplified multiple target tracking pipeline. Track management and initialization have been omitted in the diagram.

Through the use of the heading angle $0 \leq \phi_w < 2\pi$ and the vehicle's speed $v_w \geq 0$ the assumption that vehicles move only forward is modeled implicitly. Because of the representation of the motion in polar coordinates, the state transition function becomes nonlinear. Thus a Kalman Filter variant that can handle nonlinear functions needs to be used. In the implementation presented here, an Unscented Kalman Filter is used [36].

For a linear Kalmin Filter the motion model discussed leads to the state transition function in eq. 2. This function computes an estimation of the expected value of the state $\hat{x}$ at timestep k, after the time interval Δt has passed since timestep k. In the UKF, this function is used in the unscented transform.

$$\hat{x}_{k|k-1} = f(\hat{x}_{k-1}) = \begin{pmatrix} x_w + v_w \cos(\phi_w)\Delta t \\ y_w + v_w \sin(\phi_w)\Delta t \\ v_w + \dot{v}_w \Delta t \\ \phi_w + \dot{\phi}_w \Delta t \\ \dot{v}_w \\ \dot{\phi}_w \\ w_v \\ l_v \\ h_v \end{pmatrix} \tag{2}$$

The values of the acceleration $\dot{v}_w$, the angular velocity $\dot{\phi}_w$ and the object dimensions l_v, h_v and w_v are untouched by the state transition function, but are still modeled to be affected by state transition noise.

In contrast to the kinematic elements of the state vector, the static elements like the object dimensions are updated during the measurement update.

V. Camera Measurement

In a scenario where images are captured by a camera, a simple conceivable measurement function is the projection of points in world coordinates to a point measurement on the camera sensor. Such a measurement model is consistent with the point models in early tracking literature mentioned in Sec. II. The measurement model described here is based on the pinhole camera model. For points in the image plane $\vec{p}$, the equation for the projection can be written as

$$\vec{p} = \mathbf{K}\mathbf{R}_\mathbf{t}\vec{P}, \tag{3}$$

with the camera matrix $\mathbf{K}$, the rotation and translation matrix $\mathbf{R}_\mathbf{t}$, which transforms from the world coordinate frame to the camera coordinate frame and the point in world coordinates $\vec{P}$.

In the context of this article states and measurements are not only points in 3D or 2D coordinates respectively. The state vector in eq. (1) is described in Sec. IV.

The measurement is defined by a vector $\vec{y}$. It represents an axis-aligned rectangle in the acquired camera image as shown in eq. (4).

$$\vec{y} = \begin{pmatrix} x_f & y_f & w_b & h_b \end{pmatrix}^\intercal \tag{4}$$

The measurement model used in the Kalman Filter is represented by the function h, which relates the state vector $\vec{x}$ to the observations in noise $\vec{v}$ as shown in eq. (5).

$$\vec{y} = h(\vec{x}) + \vec{v}. \tag{5}$$

To be able to use the pinhole camera projection described by eq. (3), a matrix consisting of the world coordinates of eight corner points of a cuboid around the vehicle described by $\vec{x}$ is computed.

As an example on how to compute the single elements of $\mathbf{X}$, the equations for the rear left corner point of the cuboid are shown in eq. (6). After the conversion to the cuboid corners, the resulting points are independent from the velocity components of $\vec{x}$.

$$\begin{aligned} x_{rl} &= x_w - \frac{l_v}{2}\cos(\phi) - \frac{w_v}{2}\sin(\phi) \\ y_{rl} &= y_w - \frac{l_v}{2}\sin(\phi) + \frac{w_v}{2}\cos(\phi) \end{aligned} \tag{6}$$

With the projected cuboid points in the image $\mathbf{X}_f$, equation (3) can be rewritten in matrix form, which leads to eq. (7).

$$\mathbf{X}_f = \mathbf{K}v(\mathbf{M}\begin{bmatrix} \mathbf{R} & \vec{t} \\ 0 & 1 \end{bmatrix}\mathbf{X}) \tag{7}$$

The index f indicates that the points in the matrix are located in the video **f**rame coordinate system. The function v scales the extent of a projected object to its size in the image plane by dividing by the z-element, as shown in eq. (9) [19]. This function can additionally be used to model lense distortion. $\left[\begin{smallmatrix} \mathbf{R} & \vec{t} \\ 0 & 1 \end{smallmatrix}\right]$ is the matrix, which performs the rotation and translation from the world to the camera frame in homogeneous coordinates. $\mathbf{M}$ is a matrix, which converts the result from homogeneous to Cartesian coordinates.

To be able to write the following equations in a more compact way, we introduce a matrix containing the cuboid points in the camera coordinate frame $\mathbf{X}_c$.

$$\mathbf{X}_c = \mathbf{M}\begin{bmatrix} \mathbf{R} & \vec{t} \\ 0 & 1 \end{bmatrix}\mathbf{X} \tag{8}$$

The index c indicates that the points in the matrix are located in the **c**amera coordinate system.

$$v(\mathbf{X}_c)^\intercal = v \begin{pmatrix} x_{c1} & y_{c1} & z_{c1} \\ x_{c2} & y_{c2} & z_{c2} \\ \vdots & \vdots & \vdots \\ x_{c8} & y_{c8} & z_{c8} \end{pmatrix}^\intercal = \begin{bmatrix} \frac{x_{c1}}{z_{c1}} & \frac{y_{c1}}{z_{c1}} & 1 \\ \frac{x_{c2}}{z_{c2}} & \frac{y_{c2}}{z_{c2}} & 1 \\ \vdots & \vdots & \vdots \\ \frac{x_{c8}}{z_{c8}} & \frac{y_{c8}}{z_{c8}} & 1 \end{bmatrix}^\intercal \tag{9}$$

Because the third coordinate is always 1 after this transformation, with $\frac{z_{ci}}{z_{ci}} = 1$, $\mathbf{X}_c$ also contains homogeneous coordinates. $\mathbf{K}$ is the intrinsic camera matrix.

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

The focal lengths f_x and f_y and the principal point coordinates c_x and c_y are measured in pixel units (px). To convert the focal lengths from mm to px, the scaling factors s_x and s_y have to be known, which represent the scale for pixels per millimeter on the image sensor: $f_x = f \cdot s_x$.

$$\mathbf{X}_f = \mathbf{K} v(\mathbf{M} \begin{bmatrix} \mathbf{R} & \vec{t} \\ 0 & 1 \end{bmatrix} \mathbf{X}) \tag{11}$$

The nonlinear function w maps the 2D projections of the corner points of the 3D cuboid to a rectangular bounding box containing all corner points.

$$w(\mathbf{X}_f) = \begin{pmatrix} x_{min} + \frac{x_{max} - x_{min}}{2} \\ y_{min} + \frac{y_{max} - y_{min}}{2} \\ x_{max} - x_{min} \\ y_{max} - y_{min} \end{pmatrix} = h(\vec{x}) \tag{12}$$

$$\begin{aligned} x_{max} &= \min\left(w_f, \max\left(x_i\right)\right) \\ y_{max} &= \min\left(h_f, \max\left(y_i\right)\right) \\ x_{min} &= \max\left(0, \min\left(x_i\right)\right) \\ y_{min} &= \max\left(0, \min\left(y_i\right)\right) \end{aligned} \tag{13}$$

Because the width w_f and height h_f of the video frame are used in eq. (13) this measurement model only generates valid measurements within the boundaries of the image, even if some points of the object in world coordinates lie outside the image. The incorporation of this into the measurement equation leads to a correct description of the detection of vehicles only partially within the image.

Within the whole transformation from the state vector to the measurement vector, there are three nonlinear transformations: $u(\cdot)$, $v(\cdot)$ and $w(\cdot)$. $u(\cdot)$ is the mapping from the state vector for a constant velocity model of a car to the corner points of its surrounding cuboid. The nonlinear function $v(\cdot)$ represents nonlinearities in the camera projection like distortion parameters. The third nonlinear function $w(\cdot)$ exists because of the choice of the measurements as axis-aligned rectangles around the projected corner points.
In the context of tracking vehicles in traffic scenarios all three nonlinearities exist in the measurement model. Thus we arrive at the measurement equation in (14).

$$h(\vec{x}) = w(\mathbf{K} v(\mathbf{M} \begin{bmatrix} \mathbf{R} & \vec{t} \\ 0 & 1 \end{bmatrix} u(\vec{x}))) \tag{14}$$

Fig. 4. Correct estimation of the bounding box around a car only partially visible in the scene.

VI. Challenges

While the preceding section introduced the measurement model of a camera that is recording a traffic scene, the following paragraphs discuss typical phenomena and characteristic errors that can be observed in tracking with such a measurement setup. In this section a single camera with a neural network for object detection is considered.

First, effects of the camera position and perspective are considered. For the application of tracking a vehicle in an estimated ground plain, a bird's eye view on the traffic would be suited better than the perspectives considered in this article. With the structural constraints of the environment, this cannot be realized for stationary cameras in most cases. Instead the traffic is observed from an elevated position. The perspective lets closer objects occupy more space in the image plane. Distant objects appear smaller. Hence, the sensor resolution and error characteristics depend on the object position. From a KF perspective this can be seen as state-dependent measurement noise.

Especially urban traffic scenarios can be densely populated, with relevant objects on roads and side walks etc. Because of the camera perspective full or partial occlusion will always happen, either by dynamic or static objects. Occlusion by static objects can be handled by prior annotation of occluded areas in the image, as suggested in [30]. Another possible approach to solve the occlusion problem is the use of multiple cameras.

Two-staged tracking systems like [29] have problems with correctly estimating the location of objects that are only partially inside the image boundaries. This issue can be solved by using a suitable measurement function as described in Sec. V. A vehicle only partially visible is shown in Fig. 4.

There are dynamic effects related to the cameras mounted on masts and beams. Wind or vehicles can cause vibrations. The former induces camera motion that can be interpreted as a dynamic calibration error. From a KF perspective the white noise assumption is violated. The temporal correlation in measurements can degrade the tracking performance. Traffic induced vibrations typically exhibit higher frequencies. Hence,

they can be seen as a temporal change in the measurement noise intensity, for example, whenever heavy vehicles pass.

Camera images are sensitive to lighting conditions. The contrast in less illuminated areas has an effect on the visibility of objects. Reflections and direct sunlight can cause overexposure and bright areas in the images. Both effects relate to a degradation of the detection performance in the entire camera image or regions of it.

Regardless of the specific processing, typically a deep neural network, a number of challenges can be observed. There are false alarms, i.e. reported bounding boxes which do not actually contain objects. There are missed detections for actual objects which are not reported by the object detector. The detection performance depends on various parameters including the aforementioned lighting conditions and varying distances.

As described in Sec. V bounding boxes are in the best case tight rectangles around an object of interest in the image plane. With objects that enter and leave the observed region, it inevitably happens that detections are drawn around parts of a vehicle. Similar effects can be observed in the case of occlusion. The bounding boxes then represent only the visible part. The relation to the state vector in a KF framework is no longer obvious, specifically the components that relate to the object extent. In order to alleviate this, it makes sense to use metrics for the residual computation that are robust with respect to occlusion.

Beyond the scope of rectangles in camera images, the detection of an object can be complemented with further information on the object class, color, size, etc. provided by a NN. These advanced features can be employed in the measurement association of a tracker, for instance. Furthermore, velocity information can be extracted from the optical flow in consecutive images.

VII. Experiments and Results

To evaluate the performance of the multiple object tracking system discussed in this article, it is applied to the tracking of a vehicle, which recorded an RTK-corrected GNSS reference trajectory and compared against this trajectory. To show the advantage in comparison to the two-staged tracking system presented in [29], this system is applied as well and shown in this section. Both methods are compared, using the same camera calibration parameters.

To get a sufficiently accurate camera calibration, a multi-stage calibration process is applied. At first, a rough estimate of the intrinsic calibration, represented by the matrix $\mathbf{K}$ is constructed from the information in the camera datasheet. A reference video image from the camera is compared to a geotiff of the area and twenty point correspondences are manually annotated. With these point correspondences, an initial estimate of the exterior orientation is computed. Using this initial camera calibration to plot the reference trajectory to the video showed that this estimate is not good enough to yield optimal results from the algorithms. So, in the next stage of the calibration process, the cuboid points around the reference trajectory are manually annotated in each video frame. These point correspondences are then used to solve the Perspective-n-Point problem. To get a globally optimal solution and to be able to also use the out of ground plane points of the cuboids the SQPnP algorithm is used [37].

		Longitudinal		Lateral	
		Front	Rear	Front	Rear
2-stage	max	2.30 m	1.61 m	2.00 m	1.36 m
	mean	1.98 m	1.28 m	0.32 m	0.17 m
	min	0.10 m	0.14 m	0.14 m	0.17 m
our tracker	max	0.49 m	1.55 m	0.19 m	0.21 m
	mean	**0.24 m**	**0.73 m**	**0.07 m**	**0.05 m**
	min	0.00 m	0.02 m	0.00 m	0.00 m

TABLE I
ERRORS OF LONGITUDINAL AND LATERAL DIFFERENCES BETWEEN THE REFERENCE TRAJECTORY AND THE TRACKERS.

A summary of the results of the experiments is presented in table I. The mean error of our tracker is in every case significantly lower than the mean error of the 2-stage tracker.

To further look into the behaviour of the two different tracking systems, the evolution of the errors needs to be evaluated. The time series of the longitudinal errors of the 2-stage tracker is presented in red in Fig. 5. The error slightly increases after the track initialization, drops when the vehicle is close to the camera and then increases drastically towards the end of the trajectory.

Fig. 5. Longitudinal errors of the different tracking systems

The lateral errors of the 2-stage tracker are presented in Fig. 6. The error increases slightly throughout the whole track and then drastically increases at the end as well.

The initial increase in the errors can be explained by the heuristic nature of the estimation of the actual position of the vehicle. The drastic increase towards the end of the trajectory is due to the inability of this tracking system to correctly model vehicles, which are only partially visible in the image like in Fig. 4.

The time series of the longitudinal errors of the tracking system explored in this paper is shown in blue in Fig. 5. The longitudinal errors decrease after the initialization of the

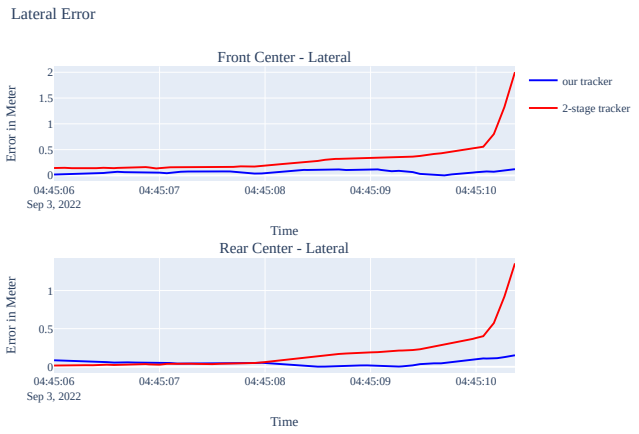

Fig. 6. Lateral errors of the different tracking systems

trajectory during the filtering and increase again towards the end of the tracking. The decrease in the errors is the expected outcome of the filtering. The increase towards the end of the tracking is due to the fact that the vehicle front is not visible anymore and the dynamics are from that point on no longer updated with measurements. The longitudinal errors of the vehicle rear are on average higher then the errors of the front, because to the tracker the influence of the vehicle length versus the influence of the height is not easily discernible. This problem may be solved by the use of additional cameras as well as by a different camera perspective.

The lateral errors are shown in Fig. 6. The errors are throughout the whole tracking very low compared to the noise with which this process is modeled. So, the evolution over time is not as expressive as in the other plots. However a significant increase in the errors is clearly visible towards the end of the trajectory. This increase is probably due to remaining issues with the camera calibration.

VIII. Conclusion and Future Work

This article has illustrated the application of Kalman filter based multiple target tracking pipelines with deep learning object detection systems for measurements. Real world challenges have been elaborated. The measurement model has been examined in detail. The experimental results show that the measurement model in combination with the applied state transition model poses a viable option to use in traffic monitoring applications. The proposed system shows a significantly better tracking performance than the 2-stage tracking system it is compared against. The choice of the measurement function also enables the system to be used with multiple and non-stationary sensors.

Future work includes the incorporation of the above insights within our tracking pipeline. This tracking pipeline will be evaluated on more real-world data, which is to be collected. For better comparability the creation and release of a benchmark dataset is planned.

Whereas this paper only treated bounding boxes, future work should also address further NN results like semantic masks. Other examples are fingerprint or identifier information for the detected objects or optical flow from consecutive images. Especially the latter provides another nonlinear measurement for tracking in the traffic monitoring context.

References

[1] H. Saul, M. Junghans, M. Dotzauer, and K. Gimm, "Online risk estimation of critical and non-critical interactions between right-turning motorists and crossing cyclists by a decision tree," *Accident Analysis & Prevention*, vol. 163, p. 106449, 2021.

[2] J. Schindler, R. Markowski, D. Wesemeyer, B. Coll-Perales, C. Böker, and S. Khan, "Infrastructure Supported Automated Driving in Transition Areas–a Prototypic Implementation," in *2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS)*, IEEE, 2020.

[3] S. Knake-Langhorst and K. Gimm, "AIM Research Intersection: Instrument for traffic detection and behavior assessment for a complex urban intersection," *Journal of large-scale research facilities JLSRF*, vol. 2, 04 2016.

[4] A. Leich, A. Kendziorra, H. Saul, and R. Hoffmann, "Calculation of error rates for detection of critical situations in road traffic," in *95th TRB Annual Meeting–Compendium of Papers*, vol. 95, Transportation Research Board, 2016.

[5] L. Dai, S. Han, J. Wang, and C. Rizos, "A Study of GPS/Glonass Multiple Reference Station Techniques for Precise Real-Time Carrier Phase-Based Positioning," 07 2001.

[6] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and Data Fusion: A Handbook of Algorithms*. Storrs, Conn: Yaakov Bar-Shalom, Apr. 2011.

[7] F. Gustafsson, *Statistical Sensor Fusion*. Lund, Sweden: Studentlitteratur, 2010.

[8] S. Särkkä, *Bayesian filtering and smoothing*. Cambridge, UK: Cambridge University Press, 2013.

[9] M. Roth, G. Hendeby, and F. Gustafsson, "EKF/UKF maneuvering target tracking using coordinated turn models with polar/Cartesian velocity," in *17th International Conference on Information Fusion (FUSION)*, July 2014.

[10] K. Granstrom, M. Baum, and S. Reuter, "Extended Object Tracking: Introduction, Overview and Applications," *Journal of Advances in Information Fusion*, vol. 12, pp. 139–174, Dec. 2016.

[11] S. Baker and I. Matthews, "Lucas-Kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[12] C. Tomasi and T. Kanade, "Detection and Tracking of Point Features," tech. rep., International Journal of Computer Vision, 1991.

[13] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol. 1, pp. 886–893, IEEE, 2005.

[14] B. Schölkopf, A. J. Smola, F. Bach, *et al.*, *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 3, pp. 583–596, 2014.

[16] A. Krizhevsky, "One weird trick for parallelizing convolutional neural networks," *CoRR*, 2014.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[18] D. Held, S. Thrun, and S. Savarese, "Learning to Track at 100 FPS with Deep Regression Networks," in *European Conference on Computer Vision (ECCV)*, 2016.

[19] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[20] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*, pp. 3464–3468, IEEE, 2016.

[21] N. Wojke, A. Bewley, and D. Paulus, "Simple Online and Realtime Tracking with a Deep Association Metric," in *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

[22] Z. Zhang, "A Flexible New Technique for Camera Calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 1330–1334, 2000.

[23] K. Kozempel, H. Saul, M. Haberjahn, and C. Kaschwich, "A comparison of trajectories and vehicle dynamics acquired by high precision GPS and contemporary methods of digital image processing," in *20th international conference on urban transport and the environment*, vol. 138, pp. 381–391, 2014.

[24] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[25] A. Clausse, S. Benslimane, and A. de La Fortelle, "Large-scale extraction of accurate vehicle trajectories for driving behavior learning," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 2391–2396, IEEE, 2019.

[26] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," *arXiv preprint arXiv:1809.10790*, 2018.

[27] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple object tracking using k-shortest paths optimization," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 9, pp. 1806–1819, 2011.

[28] T. Janz, A. Leich, M. Junghans, K. Gimm, S. Yang, and M. Baum, "Post-processing of multi-target trajectories for traffic safety analysis," in *2018 21st International Conference on Information Fusion (FUSION)*, IEEE, 2018.

[29] F. Becker, "Extending Simple Online and Realtime Tracking with Sparse Optical Flow," Februar 2021.

[30] H. Bradler, A. Kretz, and R. Mester, "Urban Traffic Surveillance (UTS): A fully probabilistic 3D tracking approach based on 2D detections," in *IEEE Intelligent Vehicles Symposium, IV 2021, Nagoya, Japan, July 11-17, 2021*, pp. 1198–1205, IEEE, 2021.

[31] S. Blackman and R. Populi, "Design and Analysis of Modern Tracking Systems," 1999.

[32] S. S. Blackman, "Multiple hypothesis tracking for multiple target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, pp. 5–18, 2004.

[33] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object Detection in 20 Years: A Survey," *CoRR*, 2019.

[34] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, pp. 308–313, 01 1965.

[35] J. Edmonds and R. M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems," *Journal of the ACM (JACM)*, vol. 19, no. 2, pp. 248–264, 1972.

[36] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *Signal Processing, Sensor Fusion, and Target Recognition VI* (I. Kadar, ed.), vol. 3068, pp. 182 – 193, International Society for Optics and Photonics, SPIE, 1997.

[37] G. Terzakis and M. Lourakis, "A consistently fast and globally optimal solution to the perspective-n-point problem," in *Computer Vision – ECCV 2020* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), (Cham), pp. 478–494, Springer International Publishing, 2020.