

Micro- and Macroscopic Road Traffic Analysis using Drone Image Data

Friedrich Kruber ✉ 

Technische Hochschule Ingolstadt, Esplanade 10, Ingolstadt, Germany

Eduardo Sánchez Morales ✉ 


Technische Hochschule Ingolstadt, Esplanade 10, Ingolstadt, Germany

Robin Egolf ✉ 

Technische Hochschule Ingolstadt, Esplanade 10, Ingolstadt, Germany

Jonas Wurst ✉ 

Technische Hochschule Ingolstadt, Esplanade 10, Ingolstadt, Germany

Samarjit Chakraborty ✉ 

University of North Carolina at Chapel Hill (UNC), Department of Computer Science, NC 27599, USA

Michael Botsch ✉ 

Technische Hochschule Ingolstadt, Esplanade 10, Ingolstadt, Germany

Abstract

The current development in the drone technology, alongside with machine learning based image processing, open new possibilities for various applications. Thus, the market volume is expected to grow rapidly over the next years. The goal of this paper is to demonstrate the capabilities and limitations of drone based image data processing for the purpose of road traffic analysis.

In the first part a method for generating microscopic traffic data is proposed. More precisely, the state of vehicles and the resulting trajectories are estimated. The method is validated by conducting experiments with reference sensors and proofs to achieve precise vehicle state estimation results. It is also shown, how the computational effort can be reduced by incorporating the tracking information

into a neural network. A discussion on current limitations supplements the findings. By collecting a large number of vehicle trajectories, macroscopic statistics, such as traffic flow and density can be obtained from the data. In the second part, a publicly available drone based data set is analyzed to evaluate the suitability for macroscopic traffic modeling. The results show that the method is well suited for gaining detailed information about macroscopic statistics, such as traffic flow dependent time headway or lane change occurrences.

In conclusion, this paper presents methods to exploit the remarkable opportunities of drone based image processing for joint macro- and microscopic traffic analysis.

2012 ACM Subject Classification Computing methodologies → Machine learning

Keywords and Phrases traffic data analysis, trajectory data, drone image data

Digital Object Identifier 10.4230/LITES.8.1.2

Supplementary Material *Software (Source Code)*: <https://github.com/fkthi/OpenTrafficMonitoringPlus>

Acknowledgements The authors acknowledge the financial support by the Federal Ministry of Education and Research of Germany (BMBF) in the framework of FH-Impuls (project number 03FH7I02IA).

Received 2021-05-11 **Accepted** 2022-01-29 **Published** 2022-11-16

Editor Samarjit Chakraborty and Qing Rao

Special Issue Special Issue on Embedded Systems for Computer Vision

1 Introduction

Drones are an excellent example of the ongoing technological development of embedded systems. The commercial drone market is already a multi-billion dollar business. Growth rates of 15 percent per year are forecast for the coming years [13]. The capability of drones to carry payloads make

02:2 Drone Image Data For Joint Micro- and Macroscopic Road Traffic Analysis

them versatile tools. They are used to detect biological microorganisms or chemical substances, to inspect technical equipment such as wind turbines and gas pipelines, or to monitor crop growth and measuring biomass, to name a few examples [56]. Most commonly, drones are equipped with cameras. The development of drones as embedded vision systems runs parallel to breakthroughs in computer vision using deep learning methods. This increases the potential for automated analysis of image data.

The present work is specifically dedicated to the application for traffic surveillance and analysis. Traffic is traditionally analyzed either on a global, macroscopic view, or in contrast, on a vehicle based, microscopic view. Section 2 summarizes typical sensor setups and study procedures from established methods.

Section 3 proposes methods for some of the core elements for the data acquisition via drones: image registration, object detection, coordinate transformation and object tracking. The requirements for object detection are high, since every pixel translates into several centimeters on the ground. For example, it is crucial to know which lane a vehicle is located in. Thus, the tolerance is accordingly in the range of a few decimeters or pixels, respectively. In order to map the images to the real world, the pixelized information has to be transformed to real world coordinates. Since a hovering drone is exposed to wind, slight movements have to be compensated with image registration techniques. By tracking vehicles over time, state variables, such as position over time, speed, acceleration and orientation can be obtained. To validate the proposed method of estimating the vehicle's state, experiments with reference sensors are conducted in this work. Lastly in Section 3, a novel approach to reduce the computational load of the object detection is proposed. For this, Kalman Filter based predictions are fed into a neural network as region proposals, which allows to deactivate one part of the neural network temporarily and increases the average throughput. The code for the complete methodology is available at <https://github.com/fkthi/OpenTrafficMonitoringPlus>.

By collecting a large number of vehicle trajectories, macroscopic traffic statistics can be derived as well. In Section 4, a publicly available dataset, acquired by means of drones, is used to validate the approach for macroscopic traffic statistics. The synchronization of the micro- and macroscopic domain can improve traffic modeling, but is difficult to accomplish with established approaches. Now, information from both domains can be captured synchronous by a single camera, mounted on a drone. Questions like how likely are lane changes performed in relation to the traffic flow, how do vehicles distribute on multi-lane roads, or how do drivers adapt distances related to the traffic situation, can be answered with bird's-eye view images from hovering drones.

Possible applications are accordingly diverse. For example, the data can be used to understand risk factors for traffic accidents by analyzing the behavior of traffic participants in intersections. Macroscopic traffic simulations benefit from real traffic measurements too. They are applied in many fields, for example to predict the effects of road network modifications, to optimize traffic signal coordination for green wave traffic, or to improve emergency vehicles travel times [10]. Roadside unit sensors are expensive to plan, install and maintain, while drone recording campaigns can be carried out at every place with low effort, so that investigations can also be conducted off the main roads and at very short notice. Such data supports engineers in making traffic in various situations more predictable and controllable as well.

Other applications target the automotive industry. Trajectory prediction and path planning are two main challenges for automated driving. Here, data is typically collected with test vehicles, equipped with reference sensors. However, the field of view from the vehicle perspective is limited because of occlusions and the range of vision, which negatively affects the understanding of other traffic participants decisions and actions. Images from aerial campaigns do not suffer from occlusion and observe a large area on ground. They allow data collection for many vehicles in parallel and capture how individual objects interact with each other.

2 Related work

This section starts with an overview on typical measurement principles for macroscopic data, followed by the microscopic data collection. The third subsection summarizes the current state of traffic surveillance from aerial imagery and provides an overview of public available data sets.

2.1 Macroscopic data

Traffic flow theory has been intensively researched for decades, starting with Greenshields publication and the first fundamental diagram in 1933 [18]. The works of [20, 23] provide a broad overview of the literature research of traffic stream characteristics and comparisons of several approaches on how to obtain data. Table 1 summarizes the measurement principles, typical sensors and the obtained measurements. It should be noted that, while some variables cannot be measured with certain methods, they can still be estimated to some extent. For example, single loop detectors cannot estimate speed, but if several of them are placed within distances of some hundred meters, average speed can still be estimated. Around 90 % of fixed sensors are induction loops [9], which are buried in the pavement. Radars and cameras are increasingly becoming established and are constantly being further developed. Floating car data, acquired by cell phones, enables end-user services for estimated travel times and alternative routes. Electronic toll collection is automated with RFID tag equipped vehicles. By measuring the time between consecutive toll readers, the travel times can be estimated as well [9]. The type of variables that can be captured by a fixed camera depends on the mounting location. For example, if they are located on a high building to perceive a large area, they offer the same possibilities as a drone. A major advantage of image-based methods is that they can deliver both, point-based and distance-based variables. Looking at an individual image frame, the distance-based traffic density can be estimated. However, if one considers a sequence of video frames, the number of vehicles crossing a specific location over time yields the traffic flow.

The costs for fixed sensor units are estimated up to USD 20,000, depending on existing structures [15]. Once installed, they deliver a constant data stream and they are less affected by weather conditions compared to drones. In contrast to that, drone data can only be obtained within certain time windows and the data acquisition causes variable costs. Batteries are a bottleneck, but this disadvantage can nowadays be compensated by tethered drone systems, which allow flight durations of several hours. Wind and water resistance, alongside with low-light capabilities of cameras, are still weak points. In return, drone based data acquisition has three key advantages: 1) data can be recorded without the necessity of additional infrastructure and 2) both, point-based and distance-based variables can be obtained, and 3) it enables joint micro- and macroscopic traffic analysis.

2.2 Microscopic data

In this work, microscopic data is defined as the estimation of vehicle state variables, such as speed, accelerations and rotation rates. By observing several nearby located vehicles, their interactions can be inferred as well. In contrast to that, the aggregation of data from many vehicles over time or space defines the macroscopic view. While the macroscopic analysis is rather of interest for transportation planners, the microscopic data is targeted towards the automotive industry. Typical data recording procedures from the ego vehicle perspective are:

- Field Operational Tests (FOT),
- Natural Driving Studies (NDS).

■ **Table 1** Measurement principles, available sensors and corresponding traffic variables for macroscopic traffic stream characteristics.

Method	Sub-Method	Measurements						
		Speed	Flow	Density	Occupancy	Travel Time	Headway	Vehicle class
Fixed sensor	single loop		x		x			
	dual loop	x	x		x		x	x
	radar	x	x		x	x	x	x
	camera	x	x	x	x	x	x	x
Drone	hovering	x	x	x	x	x	x	x
Floating cars	GPS / Cell Phones	x				x		
	Transponder (RFID)					x		

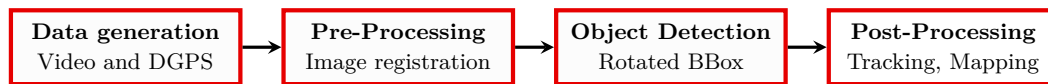
The two procedures differ according to their target [35]. In FOT, the drivers get instructions in order to validate or assess a certain functionality. As an example, the drivers are asked to evaluate a lane keeping assist. In contrast to that, NDS are free of instructions in order to analyze unbiased driving behavior. For example, NDS can be used to examine how fast drivers approach an uncontrolled intersection, or how often they perform lanes changes. Car manufacturers proceed similarly. Test drives can be performed to validate a specific functionality according to given specifications. Alternatively, long term tests can be carried out without specific instructions in order to reveal any sort of unexpected malfunction, especially in the context of system integration. In both cases, FOT and NDS, vehicles can be either equipped with series production sensors, or additionally with reference sensors, such as Lidars and additional cameras.

The scope of in-vehicle testing spans all aspects of driving: perception, planning, and execution. In contrast to that, recordings from external sensors capture only the output of these tasks. Nevertheless, such data is valuable for several purposes, such as developing trajectory planning algorithms, where the drone based recordings function as ground truth data. Another application is the development of behavior models of traffic participants.

2.3 Traffic data acquired with drones

The interest in traffic data acquisition with drones is underlined by the increasing number of publications over the past years. Here, related works are divided into object detection methods from aerial imagery, followed by publicly available data sets.

Some works propose architectures for object detection from satellite or airplane images [34, 4, 41], while the present work pursues a precise state estimation from flight altitudes of up to 100 m. Other works focus on detection and tracking from drones or infrastructure cameras [19, 53, 32]. DroNet [32] is a lean implementation of the YOLO network [45], where the number of filters in each layer is reduced. DroNet outputs several frames-per-second (fps) with onboard hardware, but at the cost of lower detection performance and image resolution. The network struggles with variations in flight height and vehicle sizes. It outputs horizontal bounding boxes, which are not favorable for estimating variables such as the vehicle orientation or yaw rate. DroNet is rather implemented for vehicle detection with a moving drone, than for vehicle state estimation. The R^3 network [34] enables the detection of rotated bounding boxes from high altitude images. R^3 is a bounding box detector, while the method in this work uses Mask R-CNN [22], which is an instance segmentation network. Except for [53], no other work focuses on the vehicle tracking and state estimation.



■ **Figure 1** The overall process: From data recording to tracking.

The work in [19] approximates the most to the present work. A test vehicle was equipped with a GPS logger that receives positions and speed. The images were geo-referenced to obtain a fixed frame. The main differences with the present work can be stated as follows: The detection algorithm in [19] compares the differences between two frames, hence identifying moving objects by localizing altered pixel values. This type of detector is prone to errors, e.g., during vehicle standstills, changing light conditions or due to the movement of vegetation, as stated by the authors. The output is a non-rotated bounding box, which fails to estimate the vehicles shape and thereby worsens the state estimation. The images in [19] were processed with a Gaussian blur filter, which is claimed to eliminate high frequency noise. Applying such a filter blurs the edges and is contra productive when applying a neural network detector. Finally, relief displacement was not taken into account, which causes an increasing error with growing distance to the principal point, see Section 3. The authors state a normalized root mean square error of 0.55 m at a flight altitude of 60 m. By the same measure, the error obtained in the present work is much lower with 0.18 m at a flight altitude of 75 m and identical image resolution. Finally, the reference sensor used in [19] has an accuracy of 0.2 m and 0.03 m/s in the position and velocity respectively, while the one used in this work has a position and velocity accuracy of 0.01 m and 0.01 m/s accordingly. A better reference sensor accuracy allows a pixel-accurate comparison, which makes the experiments more relevant.

Regarding the public data sets, two different types are available. The first type provides pre-processed trajectory data [28, 58, 47, 12, 6]. With more trajectory data sets available now, not only vehicles are tracked, but also bicycles and pedestrians among other classes. Also, the types of locations broaden, from highway to urban infrastructure, such as crossings and roundabouts. The dataset in [6] extends the perceptive field throughout a complete city district with a swarm of several drones flying simultaneously.

The other type of data sets provide labeled training data to improve computer vision methods and tracking algorithms. Large data sets can be found in [42, 59]. Despite their size, these are less useful for traffic monitoring and vehicle state estimation, since the drone does not hover but flies without providing the flight meta data of the drone. Additionally, it was found that [59] appears to lack partly label quality, which is a key factor to obtain good results. Other data sets, such as [40, 11] provide additionally flight meta data including the measurements from GPS and a Inertial Measurement Unit (IMU). In these works, the drones fly at low altitudes, thus capturing a smaller area on ground. The scope of these works is rather on visual odometry, Simultaneous Localization and Mapping (SLAM) or autonomous aerial surveillance.

3 Drone image based vehicle state estimation

This section builds on our previous work in [30, 50] and describes a method to acquire a highly accurate vehicle state based on computer vision detection. An overview of the main steps is depicted in Figure 1. In the present work, the method is extended by feeding Kalman Filter [24] based proposals into the neural network in order to reduce the computational load.

The section starts with a description of the coordinate systems, followed by the method description and experiments.

3.1 Coordinate systems

The coordinate systems used in this work are described in what follows. The vehicles move on the Local Tangent Plane (LTP), where x_L points east, y_L north and z_L upwards, with an arbitrary origin o_L on the surface of the earth. The Local Car Plane (LCP) is defined according to the ISO 8855 norm, where x_C points to the hood, y_C to the left seat, z_C upwards, with the origin o_C at the center of sprung mass of the vehicle. For simplification, it is assumed that

- 1) the $x_C y_C$ -plane is parallel to the $x_L y_L$ -plane,
- 2) the centre of sprung mass and geometrical centre of the vehicle are identical, and
- 3) the sensor in the vehicle measures in the LCP.

Quantities expressed in the LTP and LCP are given in the International System of Units (SI). The Pixel Coordinate Frame (PCF) is a vertical image projection of the LTP, where x_P and y_P represent the axes, with the origin o_P in one corner of the image. It is assumed that the camera is pointing vertically downwards. Quantities expressed in PCF are given in pixels (px).

In the following, vectors are represented in boldface and matrices in boldface, capital letters.

3.2 Data generation setup

The data set was recorded on a test track. This gives certain degrees of freedom regarding arbitrary vehicle trajectories within the image frame and allows the experiments to be repeated with the same setup. The test vehicle was equipped with an Inertial Navigation System (INS)¹, which serves as a reference sensor for the position, velocity, acceleration, orientation and rotation rate. This INS is equipped with servoaccelerometers, optical gyroscopes and receives Real-Time Kinematic (RTK) correction data. The Correvit sensor² estimates the linear velocities (v_x, v_y) along the longitudinal and lateral axes of the vehicle by means of an optical grid [21]. According to [16] the sideslip angle of the vehicle is defined as

$$\beta = \arctan\left(\frac{v_y}{v_x}\right). \quad (1)$$

Therefore, the Correvit can serve as the reference sensor for the estimated sideslip angle.

Table 2 depicts the flight altitudes and Ground Sampling Distance (GSD) for the drone³ in use, Section 3.5.1 details the computation.

■ **Table 2** Total count of frames and GSD per altitude.

Flight altitude	50 m	75 m	100 m
Number of frames	14 532	15 217	24 106
GSD [cm/px]	3.5	5.2	6.9

Generally, for a vertical photograph, the GSD S is a function of the camera's focal length f and flight altitude H above ground:

$$S = \frac{f}{H}. \quad (2)$$

¹ GeneSys ADMA-G-PRO+

² Kistler Correvit S-Motion

³ DJI Phantom 4 Pro V2



■ **Figure 2** Registered image (left) and the raw drone image (right) of a Ground Control Point (GCP) from 1.5 m height. The borders of the left image are clipped due to translation and rotation. The red box depicts the GCP location at the first video frame, which was shot around 30 s beforehand.

Varying altitudes brings flexibility in the trade-off between GSD and the captured area on the ground. For each altitude, from 50 m to 100 m, several videos were recorded. Note, that minor, unavoidable altitude differences during hovering are compensated by the image registration, see Section 3.3. The camera frame-rate f_f was set to $f_f = 50$ fps and exposure time was kept constant at $\frac{1}{400}$ s for all recordings.

3.3 Pre-processing

The pre-processing consists generally of two parts: The camera calibration and the image registration. According to the manufacturer of the drone, the camera is shipped calibrated, so this step is skipped. The image registration is performed to overlay the sequential video frames over the first one to ensure a fixed image frame. The registration implemented in this work is composed of a correction of the orientation, translation, and scaling of the image. Figure 2 depicts an example of the registration result. This process involves three steps in order to find correspondences between two images: a feature detector, a descriptor and finally the feature matching. The goal of the detector is to find identical points of interest under varying viewing conditions. The descriptor is a feature vector, which describes the local area around the point of interest. To match the points between two images, the distances between the feature vectors are computed. If the distance fulfills a certain criterion, e. g., a nearest neighbor ratio matching strategy, a matching point on two images is found. The matches are then fed into the MLESAC algorithm [54] to eliminate outliers. Lastly, a randomly selected subset of the remaining matching points is used for the image scaling, rotation and translation. The scenery recorded should offer some distinguishable, static features to ensure a robust image registration. Since all consecutive video frames are compared to the very first frame, a confusion between static and moving objects can be avoided. In this work, two competitive algorithms are applied and compared according to their execution time: SURF [7] and ORB [49], based on their openCV implementation. Additional information can be obtained from the survey [27].

3.4 Object detection

In addition to the object detection, an estimation of the dimensions and orientation of the vehicles is required for many applications. Semantic segmentation networks are suitable for this purpose, since they detect objects in random shapes, based on a pixelwise prediction. From these shapes rotated rectangles can be derived, which serve as bounding boxes for the vehicles. In particular, networks of the subgroup of instance segmentation are advantageous, since these networks output directly object wise instances, instead of a pixelwise class prediction over the complete image. For



■ **Figure 3** Detection examples: The left column depicts examples from the experiments performed on the test track. Other images are taken on roads, partly from publicly available sources [59, 44]. The masks, depicted with random colors, are used to compute the location, size and orientation of the smallest rectangle containing all mask pixels of the detected object. Note, the non-rotated bounding boxes would be the final output of typical object detectors without the segmentations masks.

this work, the Mask R-CNN network [22] is chosen for its strong detection performance for traffic surveillance with drone images. Mask R-CNN extends Faster R-CNN [46] by adding a parallel, Fully Convolutional Network [39] branch for instance segmentation, next to the classification and bounding box regression from Faster R-CNN. The network is a so called two stage detector: In the first stage, feature maps generated by a backbone network are fed into a Region Proposal Network (RPN), which outputs Regions of Interest (RoI). In the second stage, the predictions are performed within the RoI Heads. One head predicts classification and regression, the second head provides segmentation masks. These masks are used to compute the location, size and orientation of the smallest rectangle containing all mask pixels of the detected object, in our case a vehicle. The combination of the RPN with a Feature Pyramid Network (FPN) [37], both part of Mask R-CNN, achieve strong detection results for rather small objects and additionally for objects of different scales, which result from varying flight altitudes, as well as the varying objects sizes from a small car up to a large truck, see Figure 3 for examples.

The network is pre-trained on the Common Objects in Context (COCO) data set [38]. To predict vehicles from the top view, transfer learning has been applied with an own, manually labeled data set. Further details, along to the extension with Kalman Proposals are presented in Section 3.9. Implementation details are provided in our repository. Figure 3 depicts some detection examples. The left column depicts examples from the experiments performed on the test track (Section 3.7). Other images are taken on roads, partly from publicly available sources [59, 44]. Note, the non-rotated bounding boxes in Figure 3 would be the final output of typical object detectors without the segmentations masks.

3.5 Post-processing steps

To complete the process for a single image, two more steps are performed. First, the output from the neural network, given in PCF, has to be mapped to the LTP. The mapping in this work is applied in order to perform the experiments with the reference sensors. Another use-case for the mapping is the translation of the pixelized information onto a road map. In the second step, measures are taken to reduce errors induced by the relief displacement.

3.5.1 PCF Mapping

A comparison to the reference sensors requires the mapping of the PCF to the LTP. For this, GCPs are placed on the $x_L y_L$ -plane, in such a way that they are visible on the image. The i -th GCP is defined in LTP as $\mathbf{g}_{i,L} = [x_{i,L} \ y_{i,L}]^T$, and in PCF as $\mathbf{g}_{i,P} = [x_{i,P} \ y_{i,P}]^T$. The GSD S is calculated from two GCPs by

$$S = \frac{|\mathbf{g}_{i+1,L} - \mathbf{g}_{i,L}|}{|\mathbf{g}_{i+1,P} - \mathbf{g}_{i,P}|}. \quad (3)$$

The i -th GCP can then be expressed in meters by

$$\tilde{\mathbf{g}}_i = \mathbf{g}_{i,P} \cdot S = [\tilde{x}_i \quad \tilde{y}_i]^T. \quad (4)$$

The orientation offset δ from the LTP to the PCF is calculated as

$$\delta = \theta_i - \theta_{i,L}, \text{ with} \quad (5)$$

$$\theta_i = \text{atan2}(\tilde{y}_{i+1} - \tilde{y}_i, \tilde{x}_{i+1} - \tilde{x}_i), \quad (6)$$

$\theta_{i,L}$ is calculated by analogy. The GCP $\tilde{\mathbf{g}}_i$ is then rotated as follows

$$\hat{\mathbf{g}}_i = \mathbf{R}_r(\delta)^T \tilde{\mathbf{g}}_i, \quad (7)$$

where $\mathbf{R}_r(\cdot)$ is a 2D rotation matrix. The linear offsets from the LTP to the PCF are calculated by $\Delta = \hat{\mathbf{g}}_i - \mathbf{g}_{i,L}$. Finally, a pixel $\mathbf{p}_P = [x_P \quad y_P]^T$ on the PCF can be mapped to the LTP by

$$\mathbf{p}_P^L = \left(\mathbf{R}_r(\delta)^T (\mathbf{p}_P \cdot S) \right) - \Delta. \quad (8)$$

The next stage is to semantically define the four bounding box corners. It is assumed that the box covers the complete shape of the vehicle. The i -th corner of the bounding box is defined in PCF as $\mathbf{b}_i = [x_{b,i,P} \quad y_{b,i,P}]^T$, and the bounding box is defined in PCF as

$$\mathbf{B}_P = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \mathbf{b}_3 \quad \mathbf{b}_4]. \quad (9)$$

The corners of the bounding box are mapped to the LTP as shown in Eq. (8) to obtain \mathbf{B}_P^L . The geometric centre of the vehicle \mathbf{o}_{veh} is calculated by

$$\mathbf{o}_{\text{veh}} = \begin{bmatrix} \frac{\max(\mathbf{B}_P^L_{1,i}) + \min(\mathbf{B}_P^L_{1,i})}{2} \\ \frac{\max(\mathbf{B}_P^L_{2,i}) + \min(\mathbf{B}_P^L_{2,i})}{2} \end{bmatrix}, \quad (10)$$

for $i = 1, \dots, 4$. The dimensions of the detected vehicle are calculated next. Let

$$\|\mathbf{b}_2 - \mathbf{b}_1\| < \|\mathbf{b}_3 - \mathbf{b}_1\| < \|\mathbf{b}_4 - \mathbf{b}_1\|, \quad (11)$$

then $\hat{w} = S \cdot \|\mathbf{b}_2 - \mathbf{b}_1\|$ and $\hat{l} = S \cdot \|\mathbf{b}_3 - \mathbf{b}_1\|$ are the estimated width \hat{w} and length \hat{l} of the vehicle in meters.

Knowing this, the orientation ψ_{veh}^L of the vehicle is given by

$$\psi_{\text{veh}}^L = \text{atan2}(y_{j,P}^L - y_{1,P}^L, x_{j,P}^L - x_{1,P}^L), \quad (12)$$

where j is the element of \mathbf{B}_P associated with the length \hat{l} of the vehicle. The measurement vector for the Kalman Filter is then defined as

$$\mathbf{z}_{\text{in}} = [\mathbf{o}_{\text{veh}}^T \quad \psi_{\text{veh}}^L]^T. \quad (13)$$

02:10 Drone Image Data For Joint Micro- and Macroscopic Road Traffic Analysis

The estimation of the measurement noise is detailed in what follows. As described above, the GSD and the orientation offset can be estimated from one pair of GCPs as long as they are visible on the PCF. By using the same method, if $n \geq 2$ GCPs are visible on the PCF, then the number of values c that can be calculated for the GSD and for the orientation offset can be calculated as

$$c = \frac{n!}{2!(n-2)!}, \quad (14)$$

where n is the number of GCPs. An arithmetic mean \bar{S} for the GSD can then be calculated as

$$\bar{S} = \frac{\sum_{i=1}^c S_i}{c}, \quad (15)$$

where S_i is the GSD estimated with the i -th pair of GCPs. Likewise, an arithmetic mean $\bar{\delta}$ for the orientation offset can be calculated as

$$\bar{\delta} = \frac{\sum_{i=1}^c \delta_i}{c}, \quad (16)$$

where δ_i is the orientation offset estimated with the i -th pair of GCPs. The measurement error $\zeta_{o,veh}$ for the position and the measurement error $\zeta_{\psi,veh}$ for the orientation can then be calculated by

$$\zeta_{o,veh} = \max(S_i - \bar{S}), \text{ with } i \in \{1, \dots, c\}, \quad (17)$$

$$\zeta_{\psi,veh} = \max(\delta_i - \bar{\delta}), \text{ with } i \in \{1, \dots, c\}. \quad (18)$$

The measurement noise for the Kalman Filter is then defined as the diagonal matrix ζ_z

$$\zeta_z = \text{diag}(\zeta_{o,veh}, \zeta_{o,veh}, \zeta_{\psi,veh}). \quad (19)$$

3.5.2 Relief displacement

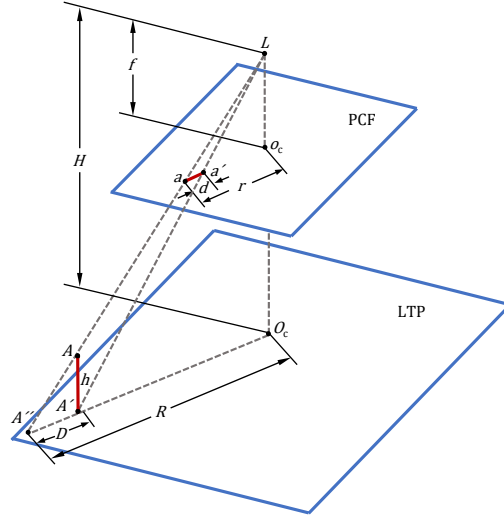
Photographs yield a perspective projection. A variation in the elevation of an object results in a different scale and a displacement of the object. An increase in the elevation of an object causes the position of the object's feature to be displaced radially outwards from the principal point O_c [36].

Assuming a vertical camera angle, the displacement can be computed from the similar triangles LO_cA'' and $AA'A''$, according to Figure 4:

$$\frac{D}{h} = \frac{R}{H}, \quad \frac{d}{h} = \frac{r}{H}, \quad (20)$$

where the second equation is expressed in GSD, with d defining the relief displacement and r the radial distance between o_c and the displaced point a in PCF. D defines the equivalent distance of d , projected on the ground, R the radial distance from O_c , H defining the flight altitude and h being the object height in LTP. L is the camera lens exposure station, where light rays from the object intersect before being imaged at the cameras' sensor. The relief displacement decreases with an increasing hovering altitude and is zero at O_c .

According to Eq. 20, the bounding box has to be shifted radially. Two approaches are considered: The first one requires knowledge of the vehicle sizes, and the second one is an approximation for unknown vehicle dimensions. Since the training is performed to detect the complete vehicle body, the corner closest to O_c can be identified as the bottom of the vehicle body. So the height of this point is equal to the ground clearance. Knowing the height of this corner, its displacement is corrected as described in what follows.



■ **Figure 4** Geometry of the relief displacement, adapted from [36]. The red bar depicts an object of height h . Due to the perspective projection and $R > 0$, the top of the bar A is displaced on the photo compared to the bottom A' . The relief displacement d is the distance between the corresponding points a and a' in the PCF.

Defining the horizontal and vertical resolution of the image as r_x and r_y , the coordinates in PCF of \mathbf{b}_i w.r.t. the image center are given by

$$\begin{bmatrix} x_{b,i,\text{img}} \\ y_{b,i,\text{img}} \end{bmatrix} = \begin{bmatrix} x_{b,i,P} - \frac{r_x}{2} \\ y_{b,i,P} - \frac{r_y}{2} \end{bmatrix}. \quad (21)$$

The shift $\Delta_{x,P}$ along the x_P axis is calculated on the PCF by

$$\Delta_{x,P} = \frac{x_{b,i,\text{img}} \cdot h_{b,i,L}}{H}, \quad (22)$$

where $h_{b,i,L}$ is the height of the i -th corner on the LTP. The shift for $\Delta_{y,P}$ is computed by analogy along the y_P axis. The shifted coordinates $\mathbf{b}_{i,\text{shift}}$ of \mathbf{b}_i are then given by

$$\mathbf{b}_{i,\text{shift}} = \mathbf{b}_i - [\Delta_{x,P} \quad \Delta_{y,P}]^T. \quad (23)$$

Let w be the width and l the known length of the vehicle and \mathbf{b}_1 be the closest corner to the image centre. Then, \mathbf{b}_1 is used for scaling \mathbf{b}_2 and \mathbf{b}_3 as follows

$$\mathbf{b}_{w,\text{scaled}} = \left(\frac{w}{\hat{w}} \cdot (\mathbf{b}_2 - \mathbf{b}_1) \right) + \mathbf{b}_1, \text{ and} \quad (24)$$

$$\mathbf{b}_{l,\text{scaled}} = \left(\frac{l}{\hat{l}} \cdot (\mathbf{b}_3 - \mathbf{b}_1) \right) + \mathbf{b}_1, \quad (25)$$

where w is the element of \mathbf{B}_P associated with $\|\mathbf{b}_2 - \mathbf{b}_1\|$ and l associated with $\|\mathbf{b}_3 - \mathbf{b}_1\|$, respectively. The shifted centre of the vehicle can then be calculated by

$$\mathbf{o}_{\text{veh,shift}} = \frac{\mathbf{b}_{w,\text{scaled}} + \mathbf{b}_{l,\text{scaled}}}{2}. \quad (26)$$

When gathering data on public roads, the vehicle dimensions are unknown and cannot be estimated with a mono camera. An approximation for the displacement can be performed by assuming that two sides of the bounding box closest to o_c , are collinear to the lowest part of the vehicle chassis. The ground clearance can be approximated as 15 cm for passenger cars [55]. The remaining two sides can usually be referred to as the vehicle body shoulders, which usually protrude further than the roof of the vehicle. The shoulders height is roughly half of the vehicle height and can be approximated with 75 cm for passenger cars. Then all four corners can be shifted following Eq. (22). Although this is only a coarse approximation, the overall error is reduced when compared to the initial situation of neglecting the displacement.

3.6 Tracking and state estimation

To enable the object tracking, the detections must be associated across frames in a sequence of images. The association procedure in this paper follows the computationally efficient SORT algorithm [8]. In contrast to batch tracking methods, this algorithm solely requires information from the previous and current frame. It is therefore suitable for real-time applications and endless video recording or streaming. In principle, an Intersection-over-Union (IoU) distance is computed for all detected vehicle shapes from two consecutive frames and stored in a cost matrix. The assignment is computed optimally using the Hungarian algorithm [31]. If no detection is associated to a vehicle no corresponding measurement vector can be generated. However, its state is continuously predicted without measurement variables using the Kalman Filter. After a defined number of frames without association, the vehicle track is withdrawn. The IoU distance allows implicitly short-term occlusions.

Having the measurement vector from Equation (13), the state vector from Equation (27) and measurement noise from Equation (19) assigned to an object, the next step is to estimate the vehicle state using the Kalman Filter as described in [24]. The Kalman Filter also allows to estimate state variables that are not part of the measurement vector by allowing the system noise to propagate. The specifics applicable to this work are described in the following.

The used state vector is defined as

$$\mathbf{x} = [x_{\text{car}}, y_{\text{car}}, v_{x,\text{car}}, v_{y,\text{car}}, a_{x,\text{car}}, a_{y,\text{car}}, \psi_{\text{car}}, \dot{\psi}_{\text{car}}]^T, \quad (27)$$

where x_{car} and y_{car} are the (x,y) coordinates of \mathbf{o}_{car} in LTP, $v_{x,\text{car}}$ and $v_{y,\text{car}}$ are the velocities of \mathbf{o}_{car} along the x_L and y_L axes, $a_{x,\text{car}}$ and $a_{y,\text{car}}$ are the accelerations of \mathbf{o}_{car} along the x_L and y_L axes, ψ_{car} is the angle from x_L to x_C in LTP, and $\dot{\psi}_{\text{car}}$ is the yaw rate around z_C . In this work a linear motion model is used.

Since the course over ground θ_{cog} in LTP of the vehicle is defined as

$$\theta_{\text{cog}} = \text{atan2}(v_{y,\text{car}}, v_{x,\text{car}}), \quad (28)$$

the sideslip β of the car can then be calculated by

$$\beta = \theta_{\text{cog}} - \psi_{\text{car}}. \quad (29)$$

Detailed information about vehicle dynamics and non-tractive driving can be found in [3] and [52].

In this work, the sideslip angle is estimated by means of a Linear Kalman Filter and Equation (29). This produces better results than using an Extended Kalman Filter. This is explained by the fact the sideslip angle is not part of the measurement vector. Using an Extended Kalman Filter would imply the estimation of the sideslip angle by noise propagation, whereas Equation (29) allows a direct calculation.

3.7 Experiments

This section details the experiments carried out on the test track for the validation of the proposed methods. The errors are defined as the deviation from the reference sensors to the estimated states. Figure 5 a) depicts the cumulative error curves for the position estimation before applying the Kalman Filter, so that the results represent unfiltered detections. A spiral template trajectory is driven to obtain different vehicle poses and to cover a large are of the image. The test vehicle is then equipped with a driving robot and a Satellite Navigation (SatNav) system that receives RTK correction data. This ensures an identical reproduction of the trajectory for all experiments, and a centimeter-accurate vehicle localization. No markers are placed on the vehicle to approach real-testing conditions on public roads.

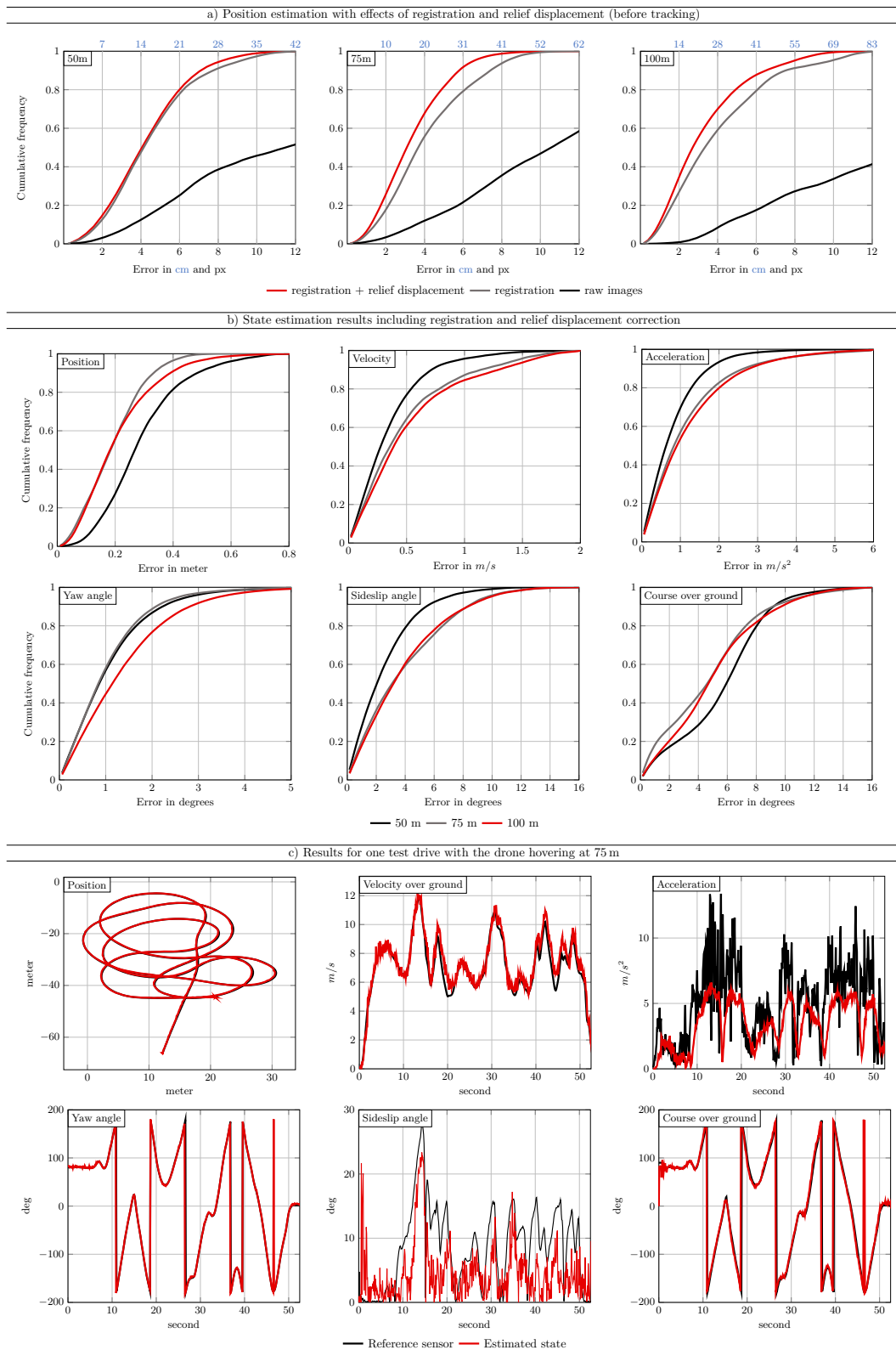
Depicted are curves for each flight altitude and the three main processing steps, where **—** depicts results for non-registered images, **—** for registered images, and **—** for registered images with corrected relief displacement. Image registration is the key to obtain reasonable results. The correction of the relief displacement improves the results by 0.8 px on a weighted average⁴. Note, that the impact of the relief displacement is dependent on the distance R of the vehicle to the image center O_c . Hence, data sets recording vehicles at the image border benefit more. The mean position error is 0.2 m and 0.14 m for a flight altitude of 100 m and 50 m, respectively. In terms of pixels, the errors are comparable for all flight heights. Around 90 % of all frames have an error of 7 px or less.

Figure 5 b) depicts the cumulative error curves for all estimated state variables, where **—** depicts cumulative error plots for 50 m, **—** for 75 m and **—** for 100 m hovering altitude, respectively. To generate the data from Figure 5 b) and c), which includes the tracking and Kalman Filter, the vehicle is equipped with the reference sensors and is driven in a random manner on a test track. No specific maneuver is driven in order to avoid tuning Kalman Filter parameters for a specific trajectory or specific driving style. The test drives include standstill, walking velocity, high acceleration, hard braking, tractive and non-tractive driving (drifting). The results show that, once steps are taken to minimize errors, the precision of the estimated vehicle state is comparable to the precision of consumer-grade sensors, such as silicon-based INSS or SatNav receivers with no correction data. This with the advantage of being able to record information for various traffic participants with a single drone. Also, the precision of the estimated sideslip angle allows to differentiate between tractive and non-tractive driving [3]. As part of the experiment strategy, the vehicle was forced to perform drift maneuvers, so that the side slip angle reached high values of up to 28°. The sideslip angle is a relevant state variable to determine the vehicle stability.

Figure 5 c) depicts the estimated state variables (**—**) against the reference sensors (**—**) for one test drive, which includes full-throttle acceleration, hard braking and sudden steering. The estimated position, course over ground and yaw angle for this trial have a mean error of 0.19 m, 4.7° and 1.0° respectively. This precision is equivalent to that of consumer-grade INSS. Subfigure c) also shows that the estimated velocity is affected by a dampening effect and by a time delay. Both are caused by the Kalman gains. A test-specific tuning of the gains could help to reduce the velocity error for this trial, but would increase the error for other tests with lower vehicle dynamics. The deviation of the estimated sideslip is caused mainly by the velocity error. This is because the sideslip angle is estimated using the course over ground, which is calculated from the velocity. During this test, a sideslip angle of 28° is reached, which clearly indicates that the vehicle is drifting. The error in the acceleration is explained by two facts: First, the estimated acceleration is calculated by system noise propagation, so it is low-pass filtered. Second, the reference acceleration, that is measured by the INS, includes vibrations from the drivetrain, the tires and the suspension, as well as from the pitch and roll of the vehicle.

⁴ weighted by the number of frames per height

02:14 Drone Image Data For Joint Micro- and Macroscopic Road Traffic Analysis



■ **Figure 5** Depicted are: a) Cumulative error curves for the position estimation with the effect of registration and relief displacement before applying the KF, b) Cumulative error curves for all estimated state variables after the KF, c) Estimated states variables against the reference sensor for one test drive with high longitudinal and lateral dynamics.

A summary of the experimental results leads to the following conclusions:

- 1) A robust image registration is crucial for a good performance. If the effects of the relief displacement are neglected, larger errors are present in the estimated vehicle state. This error increases as the hovering altitude decreases and as the objects are farther away from the nadir point.
- 2) Considering the pixelwise results, similar performance is observed for all three altitudes, which proves that the traffic data can be obtained at different flight heights by a single Mask R-CNN network. This advantage can also be helpful for detecting other object classes.
- 3) As a consequence of the GSD, the best results in metric units are achieved at lower altitudes. Alternatively, in order to capture a larger surface area, one can raise the drone to higher altitudes, increase the resolution and crop the image if necessary.
- 4) Regarding real-world applications, the vehicle can be associated to a lane with a precise velocity and orientation estimation, and it can be identified whether the vehicle is drifting or not.

To identify the root causes of the deviations from ground truth, the next section analyzes important sources of errors.

3.8 Limitations and sources of error

The methodology involves several processing steps with associated error sources. The main sources of errors are listed in what follows:

- pixel ambiguity and blurring effect,
- relief displacement,
- sensor synchronization.

The blurring effect that can be appreciated on the images can be caused by image compression, camera optics or light propagation. It is precisely this blurring effect that prevents to unambiguously associate a point to a GCP, or a pixel to the vehicle during the labeling of training data or object detection. A pixel ambiguity $\zeta \pm 1$ px in both, the x_P and y_P axes is not uncommon. Since the detection error of ± 1 px per axis is transferred to the bounding box, then \mathbf{b}_i could have an error of $\sqrt{2}$ px.

The effect on the PCF to LTP mapping is shown next. Similarly, the effect of pixel ambiguity can be applied to the image registration process, when pixel-pairs of two images do not match exactly. For a GCP, the $\mathbf{g}_{i,P}$ is rewritten as

$$\mathbf{g}_{i,P} = [x_{i,P} \quad y_{i,P}]^T \pm \zeta. \quad (30)$$

This association ambiguity has an effect on all three parameters that map the PCF to the LTP. The case of the spatial resolution is analyzed first.

Considering a pixel ambiguity of $\zeta \pm 1$ px per axis and squared pixels, the distance between the true and associated positions of a GCP on the PCF can be of $\sqrt{2}$ px. This mis-association causes an error on the spatial resolution. The similarity in percentage η_S between the seen and the true values of the spatial resolution is calculated by

$$\eta_S = \frac{|\mathbf{g}_{i+1,P} - \mathbf{g}_{i,P}|}{|\mathbf{g}_{i+1,P} - \mathbf{g}_{i,P}| + 2 \cdot \sqrt{2}\zeta^2}. \quad (31)$$

The Equation (4) is then rewritten as

$$\mathbf{g}'_i = \mathbf{g}_{i,P} \cdot S \cdot \eta_S = [x'_i \quad y'_i]^T. \quad (32)$$

02:16 Drone Image Data For Joint Micro- and Macroscopic Road Traffic Analysis

From the Equations (31) and (32), it can be deduced that the effect of η_f increases as $\mathbf{g}_{i,P} \rightarrow \mathbf{g}_{i+1,P}$. In a scenario for a Full-HD image, where both GCPs are placed on opposite diagonal corners of the picture, the similarity can drop to 99.87%. For example, if the true value of $|\mathbf{g}_{i+1,L} - \mathbf{g}_{i,L}|$ is 100 m, a similarity of 99.87% will rescale it as 99.87 m, meaning a 0.13 m difference.

Next, the effect on the orientation offset is analyzed. This is done in pixels to decouple errors caused by η_s . To consider the pixel ambiguity, the Equation (6) is rewritten as

$$\xi_{\theta_{\text{image}}} = \text{atan2}((\tilde{y}_{i+1} - \tilde{y}_i) \pm 2\zeta, (\tilde{x}_{i+1} - \tilde{x}_i) \pm 2\zeta), \quad (33)$$

where ζ is multiplied by two because $\xi_{\theta_{\text{image}}}$ is calculated using two GCPs. Similar as with η_s , the effect of the pixel ambiguity increases as $\mathbf{g}_{i,P} \rightarrow \mathbf{g}_{i+1,P}$. In a scenario for a Full-HD image, where the GCPs are in opposing diagonal corners of the image, then $\eta_{\xi_{\theta_{\text{image}}}}$ could reach 0.07° .

The orientation error affects the rotation step of the PCF to LTP mapping. So, the effect of orientation error increases as the points to map are farther from the rotation axis. For the i -th corner of the bounding box, the mapping error η_{b_i} due to the orientation error is given by

$$\eta_{b_i} = \left| \mathbf{R} \left(\eta_{\xi_{\theta_{\text{image}}}} \right) \mathbf{b}_i - \mathbf{b}_i \right| \cdot S. \quad (34)$$

For example, if the drone records a Full-HD video while hovering at 50 m, $\mathbf{b}_i = [1920 \quad 1080]^\text{T}$ and $\eta_{\xi_{\theta_{\text{image}}}} = 0.07^\circ$, then $\eta_{b_i} \approx 0.08$ m.

The error propagation causes a deviation on the linear offsets as well. The linear offset error η_{ξ_d} due to the orientation and scaling errors is expressed by

$$\eta_{\xi_d} = \left(\left(\mathbf{R} \left(\eta_{\xi_{\theta_{\text{image}}}} \right)^\text{T} (\mathbf{g}_{i,P} \cdot \eta_s) \right) - \mathbf{g}_{i,P} \right) \cdot S. \quad (35)$$

In a scenario where the drone records a Full-HD video while hovering at 50 m, $\mathbf{g}_{i,P} = [1920 \quad 1080]^\text{T}$, a similarity of 99.87% and orientation error of 0.07° , then $\eta_{\xi_d} \approx [-0.13 \text{ m} \quad -0.03 \text{ m}]^\text{T}$.

From the previous, it is deduced that the best way to minimize errors caused by the PCF to LTP mapping, is to locate the GCPs as far from each other as possible. Also, since the direction of the pixel ambiguity is not deterministic, the errors can be compensated by using different combinations of various GCPs.

Next, the effect of the relief displacement is analyzed. The maximum positioning error η_r , when assuming that the true centroid of the vehicle corresponds to the seen centroid yields:

$$\eta_r = \frac{\sqrt{x_{b,i,\text{img}}^2 + y_{b,i,\text{img}}^2} \cdot S \cdot (h - h_c)}{2 \cdot H}. \quad (36)$$

For example, if the drone hovers at 50 m, the vehicle height is $h = 1.4$ m, the ground clearance is $h_c = 0.15$ m and \mathbf{b}_i is on one corner of the image, then $\eta_r \approx 0.48$ m. From the discussion above it follows that the best way to minimize positioning errors due to relief displacement, is to correct it only for the corner nearest to the centre of the image, and to re-scale the bounding box.

Another relevant source of error is the synchronization of the sensors. This error is only relevant in a multi-sensor setup as used in this work for the experiments, which is usually not required for real world applications. In this work, the synchronization is performed by using the in-built PPS-LED light of a SatNav receiver as reference. The rising edge of the PPS pulse indicates the start of every second. This rising edge is used as a trigger for lighting up a LED that stays on for a determined period of time so that the light can be seen on the image to be processed. Since this LED is part of the SatNav module, the latency between the PPS reception and the LED lighting up is negligible. The start of each second can be known with frame-accuracy by recording this

LED. The limitation of this technique is that the videos are a series of static pictures. Hence, it is not known if the LED lights up when the shutter is closed, creating a synchronization error. The maximum synchronization error η_τ is given by

$$\eta_\tau = \frac{1}{f_i}. \quad (37)$$

In this work, 50 fps are used, so that $\eta_\tau \leq 0.02$ s. The positioning error η_{pos} caused by η_τ is given by

$$\eta_{\text{pos}} = \sqrt{v_{x,\text{car}}^2 + v_{y,\text{car}}^2} \cdot \eta_\tau. \quad (38)$$

As an example with a vehicle moving with 50 km/h and a drone hovering at 100 m recording a video with 50 fps, then $\eta_{\text{pos}} \leq 0.25$ m. It can be deduced from what is discussed above that synchronization errors, even in the millisecond order, have a significant influence.

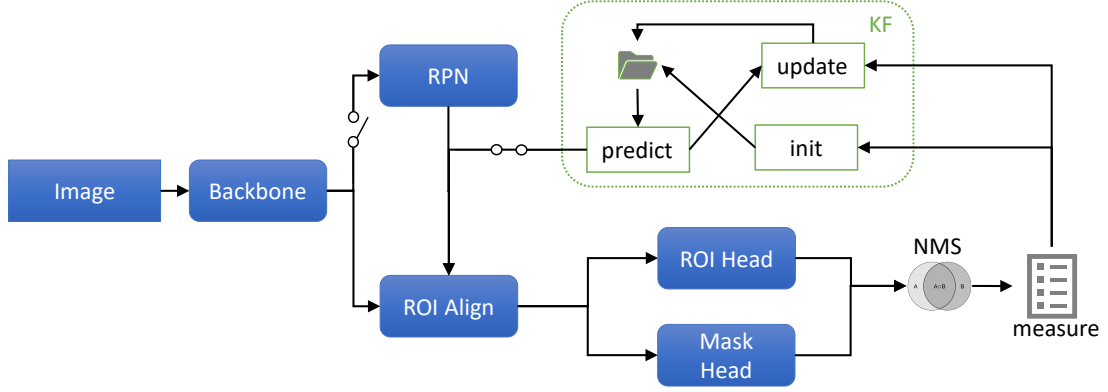
3.9 Extension with Kalman Proposals

Common object detection methods are designed for standstill images without spatiotemporal correlation. As a result, a massive amount of anchors are generated throughout the complete image. In the case of Mask R-CNN, these proposals are generated in the Region Proposal Network (RPN). The proposals are ranked by confidence and a predefined number of top ranked proposals is fed into the second stage for the computation of the classification, regression and mask output.

Image sequences are highly correlated, especially when traffic is recorded from a bird's-eye view. Several works propose architectures, which integrate the tracking task into a neural network. Usually, these papers deal with the challenge of the visual appearances changes in natural captured images, i. e. images from a human perspective [57, 26, 25, 5, 17]. To combine detection and tracking in a neural network is appealing. However, the approaches have some of the following drawbacks: 1) The methods require a batch-wise computation, which means they process a complete video sequence at once. The batch-wise computation makes them unsuitable for online tracking, which is required in real time applications such as surveillance video streams. 2) Visual tracking requires storing the information over consecutive frames, which limits the sequences length input.

Changing appearances are not a particular problem for detecting vehicles from bird's-eye view images. Therefore, in this paper an alternative approach is introduced, which efficiently utilizes the existing information from previous frames to accelerate the detection by coupling the Kalman Filter predictions into the neural network. Compared to the related works above, this approach is forward capable, i. e. one is not restricted to run through a complete video sequence at once. The Kalman proposal method can be added to any detector which predicts the output based on region proposals. The Kalman predicted proposals are guided into the second stage of the detector and replace the RPN network for a defined time window. Since only few Kalman Proposals are sufficient for accomplishing the task, less computational resources are required when compared to the typical brute-force proposal generation. To match the expected input for the ROI heads, the Kalman Proposals are fed in as non-rotated bounding boxes.

For initializing the tracks, the RPN is still required and switched on for a certain amount of frames. Then it is turned off and only Kalman Proposals are fed into the second stage of Mask R-CNN. The region proposals for the next video frames are estimated based on the Kalman Filter predicted position of the vehicle and its estimated size. In order to feed these regions to the Mask R-CNN network, they are transformed from the LTP back to the PCF. Afterwards, the same cycle is repeated for detecting new objects. Figure 6 illustrates the changes applied to a Mask R-CNN network, with Kalman Predictions active and the RPN deactivated.



■ **Figure 6** Kalman-Proposals, as part of the assignment and tracking task, are fed into a Mask R-CNN network. The RPN can be switched on and off depending on a time-based event. When the RPN is switched off, only Kalman Predictions are fed into the second stage of Mask R-CNN. The output of the Mask Head functions as measurements induced to the tracking task. The final output is acquired from the KF objects, depicted with a folder symbol.

■ **Table 3** Performance results for Kalman Proposals: On the left side the RPN is turned on for a varying number of consecutive frames and turned off for $\gamma = 23$ frames. On the right side the RPN is turned on for 10 frames and turned off for a varying number of frames, where Kalman Proposals are fed in instead. True Positives are depicted below (\checkmark), False Negatives below (\times) and hit rates below $\frac{\checkmark}{\checkmark + \times}$.

RPN on	RPN off	\checkmark	\times	$\frac{\checkmark}{\checkmark + \times}$	RPN on	RPN off	\checkmark	\times	$\frac{\checkmark}{\checkmark + \times}$
3	23	52544	396	99.1%	10	10	52808	132	99.7%
5	23	52574	366	99.2%	10	23	52616	324	99.3%
10	23	52616	324	99.3%	10	50	51973	967	98.1%
20	23	52657	283	99.4%	10	75	51143	1797	96.5%

In order to evaluate the Kalman Proposals, five video sequences with totally 10 000 frames are analyzed. The videos were recorded at a public roundabout location. When the vehicles enter or leave the image, they are not fully visible, which results in a shifted centroid \mathbf{o}_{veh} . The centroid is used for tracking, as described in Section 3.6. In other words, during entering or exiting the image frame, the vehicles' state estimations are not valid. Therefore, the tracking generally only starts at a minimum vehicle length ratio $\tau = \frac{\hat{l}_{q,t}}{\hat{l}_q}$, where $\hat{l}_{q,t}$ is the estimated vehicle length at frame t and \hat{l}_q the arithmetic mean value of all measurements for that specific q -th vehicle. In these experiments, the threshold is set to $\tau = 80\%$.





Next, a suitable time window, where only Kalman Proposals are fed into the network, has to be defined. The suggested approach is to set the trigger according to the maximum velocity observed for all vehicles in relation to the average vehicle length \hat{l} and frame rate f_f :

$$\gamma = \frac{\max(v)}{\hat{l}} f_f. \quad (39)$$

Each video is initialized with 50 frames to compute a first valid γ . Then, the RPN is turned off according to γ and turned on for between 3 and 20 frames, as depicted in Table 3 on the left side. Next, the RPN is turned on for 10 frames and its off cycle is varied up to a 75 frames, as depicted in Table 3 on the right side. The results show for experiments with γ a hit rate of over 99%. Furthermore, shorter switching cycles show better performance. Expanding the Kalman

■ **Table 4** Execution time in milliseconds. With Kalman Proposals, the speed gain is achieved at the RPN and ROI stage. Lowering the resolution by half reduces the execution by approximately two thirds.

Resolution [px]	Registration		Detection with RPN					Tracking	Total Runtime
	SURF	ORB	Backbone	RPN/KP	ROI Heads	Pre+ Post	CUDA Total		
1920x1080	90	38	57	17	8	12	94	4	136
960x540	33	18	16	7	7	4	34	4	56
			Detection with Kalman Proposals						
1920x1080	90	38	57	2	4	11	74	4	116
960x540	33	18	16	2	4	4	26	4	48

* A100 [2020]  2.9
 RTX 3090 [2020]  1.75
 * V100 [2017]  1.25
 2080 Ti [2018]  1

■ **Figure 7** Comparison between two generations of consumer and server grade (*) Nvidia GPUs with their release date in squared brackets. The numbers indicate the relative training throughput for a Mask R-CNN network according to [33].

Proposal bounding boxes to compensate prediction uncertainty yields lower performance. For example, increasing the boxes in both axis by 5% or 10% reduces the hit rate by approximately 0.4% and 0.7%. Finally, it should be noted, that the runtime and hit rate with Kalman proposals is tested with a Mask R-CNN network. Nevertheless, these proposals are applicable for replacing the common brutal force proposal generation as part of other neural network architectures as well. The execution time for the detection is not only reduced in the region proposal part, but also in the final detection and segmentation head, since only a few proposals are evaluated by the network.

3.9.1 Runtime and hardware requirements

This section takes a in-depth look at the runtimes. The measurements are depicted in Table 4. The code was processed on a workstation⁵. Image registration was performed with the ORB and SURF implementation from the OpenCV library. The measurement includes the writing of registered images on the hard drive. ORB outperforms SURF in terms of execution time. Nevertheless, the registration acquires 38 ms for a FHD resolution. The other major bottlenecks are the backbone (ResNet50-FPN) and the RPN. The average RPN execution time can be dramatically reduced by using the Kalman Proposals. The pre- and post-processing part within the detection performs image normalization, copies the image into the GPU RAM and expands the predictions to the input image size after the network again.

Regarding the object detection, the Kalman Proposals reduce the time for the proposal generation, as well as for the ROI heads. The proposal generation takes only 2 ms instead of 17 ms, and the ROI heads runtime is reduced by half. The runtime is reduced to about one third by decreasing the resolution by half. The tracking part includes copying the results to the hard disk,

⁵ Intel Xeon E-2176G, Nvidia Geforce 2080 Ti, M.2 SSD

the code is written in Python. Overall, the complete method achieves approximately 7 fps at FHD and 18 fps at half FHD resolution, considering the workstation in use. A benchmark from [33] indicates, that the throughput for a Mask R-CNN network increases by about 75% with the next generation GPU, compared to the GPU used in this work, see Figure 7. With new generation hardware, typical camera frame rates of 25 or 30 fps are within reach for half FHD resolution.

In order to achieve a higher throughput, besides faster hardware, several aspects can be considered for future work: 1) perform the registration on the GPU directly and pass the image tensor on to the detection network, 2) avoid image registration by detecting Ground Control Points automatically in order to apply the image transformation, 3) replace the backbone by a light-weight network.

On-board GPU accelerated hardware, mounted on a drone, can only process a fraction of the information compared to a full-size GPU. Therefore, running high resolution image registration and segmentation on such platforms remains a challenging task currently.

4 Macroscopic statistics

In the previous section the focus was to detail how to generate new traffic data sets by using drones. The focus of the following is to show how such data sets can be used to perform a macroscopic traffic analysis. The pre-processed image data from the publicly available highD dataset [28] is used because such type of analysis benefit from bigger data sets. The results are compared with literature data to examine the validity of the approach.

4.1 Data set description

The highD data set was published alongside with the publication [28]. Videos of German motorways were recorded from a drone perspective. The following list, adopted from [28], depicts some key facts about the data set:

- around 110 500 vehicles recorded,
- road length of about 420 m with two or three lanes,
- the total driven distance is 45 000 km,
- the GSD is 0.1 m/px.

The semantic segmentation was performed using a modified U-Net neural network [48]. Transfer learning was applied with around 3000 manually labeled image patches. The trajectories were smoothed and validated in a post-processing step according to the authors. Already preprocessed variables are for example the velocities, accelerations and time headway for each recorded frame. The vehicles are divided into two classes: cars and trucks.

The present work focuses on the aggregated, macroscopic view on the data. For an example for utilizing the data for a vehicle based view, interested readers are referred to [51]. In that work, cut-ins and hard braking maneuvers are analyzed and related to automatic emergency braking system alerts.

4.2 Variables of interest

The three macroscopic variables considered in this work are the traffic flow rate, the traffic density and the average velocity of the traffic stream. With these variables the fundamental diagrams are plotted. The flow rate q is defined as

$$q = \text{vehicles}/\text{time} \tag{40}$$

and is measured at a certain point over time. The traffic density ρ is defined as

$$\rho = \text{vehicles}/\text{distance}. \quad (41)$$

In order to measure the traffic density, a road segment of at least several hundred meters should be observed [20]. The third and last variable, the average velocity, can be estimated as “time mean velocity” or “space mean velocity” [20]. The difference of both is minor for stable traffic flow and therefore neglected in the following.

Two more relevant variables for traffic modeling are the distance to the front vehicle, the distance headway (DHW), and the time headway (THW). The THW is defined as

$$THW = \frac{x_l - x_f}{v_f}, \quad (42)$$

where x denotes the position and v the velocity. The subscript l denotes the leader vehicle, the subscript f the follower vehicle.

4.3 Fundamental diagrams

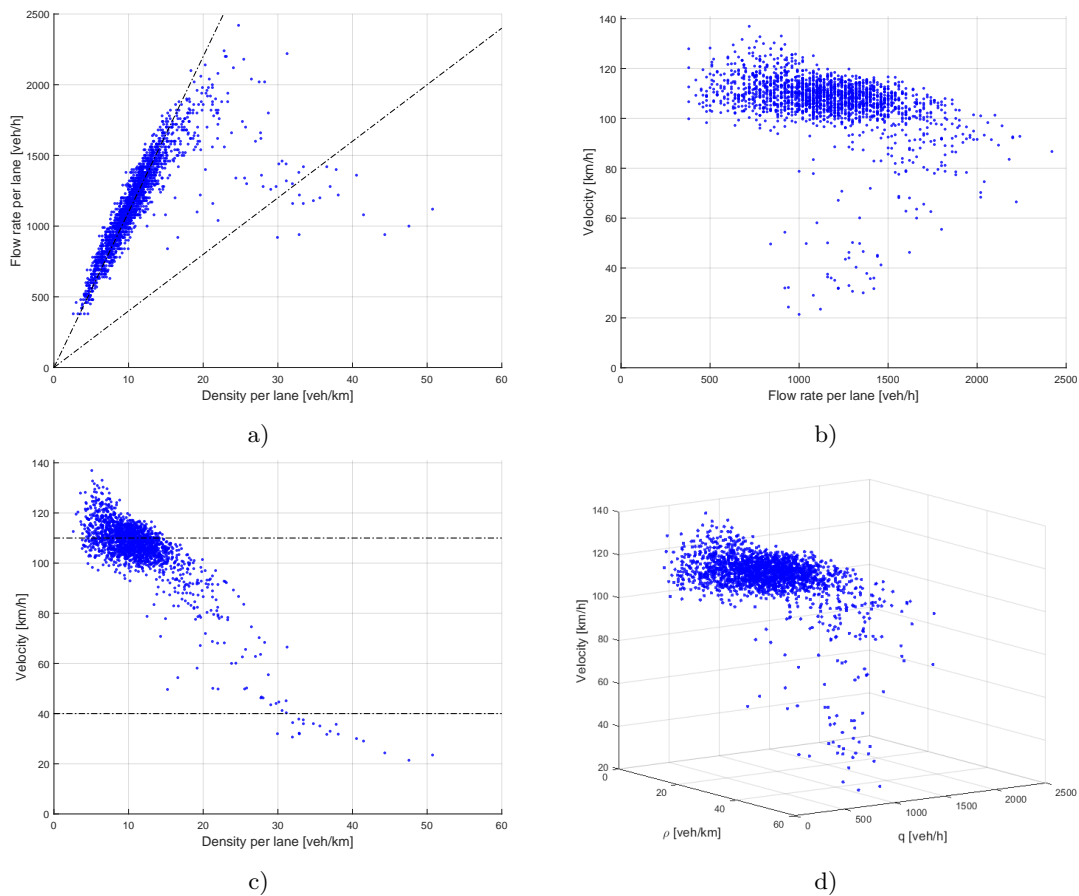
This subsection starts with a brief description of the terms free flow, bounded flow and congested traffic. Graphically, these terms are depicted in the fundamental traffic diagrams. The fundamental diagrams visualize the relationship of traffic flow, density and speed. In this work, these macroscopic variables are also linked to the microscopic based THW measure in order to demonstrate the interconnection between both domains. A synchronous linking of both domains is difficult with traditional methods, but can be easily achieved with drone based image data analysis.

The terms free flow, bounded flow and congested/jammed traffic are briefly discussed based on [43] and referenced literature. Graphically, they are depicted by the dashed lines in the $\rho - q$ diagram in Figure 8 a).

During free flow traffic, the dependency of the flow rate to the density can be fitted linearly, as the fluctuations are minor. The positive signed slope of the curve represents the average velocity for that traffic type. Exceeding a certain density threshold, the average velocity drops significantly. The critical density threshold depends on the drivers and environmental parameters, including the road infrastructure. Beyond the critical density threshold, the traffic characteristics change and reach the state of bounded traffic. Bounded traffic can be divided into three classes. First, the homogeneous flow, where density and velocities stay rather constant. This state can be depicted as a point in the fundamental diagram. In the second case, the velocity is homogeneous, but the flow rate and density vary over time and space. This state can be depicted as a line with negative gradient in the fundamental diagram. In the third and most frequently appearing class for bounded traffic, all three parameters vary, while vehicles still keep a certain velocity. Considering time series data, the consecutive datapoints of this inhomogeneous traffic stream class jump stochastically in the fundamental diagram. A further increase of the traffic density yields a stop-and-go situation followed by the jammed traffic with zero velocity.

The following part of this subsection depicts the classical fundamental diagrams derived from the data set. The relationship between the density and flow rate is depicted in Figure 8 a). For free flow traffic up to around $\rho = 10 - 15 \text{ veh/km}$, the curve can be fitted linearly. The correlation coefficient R_c for $\rho < 15 \text{ veh/km}$ is $R_c = 0.94$. The zone for bounded traffic up to stop-and-go traffic is depicted within the two dashed lines. The transition from stop-and-go to jammed traffic arises with a density of approximately $\rho = 30 - 40 \text{ veh/km}$, while saturating at around $\rho = 45 - 50 \text{ veh/km}$ per lane. The plot coincides with the results of the empirical studies in [14, 43], which were obtained with another data source.

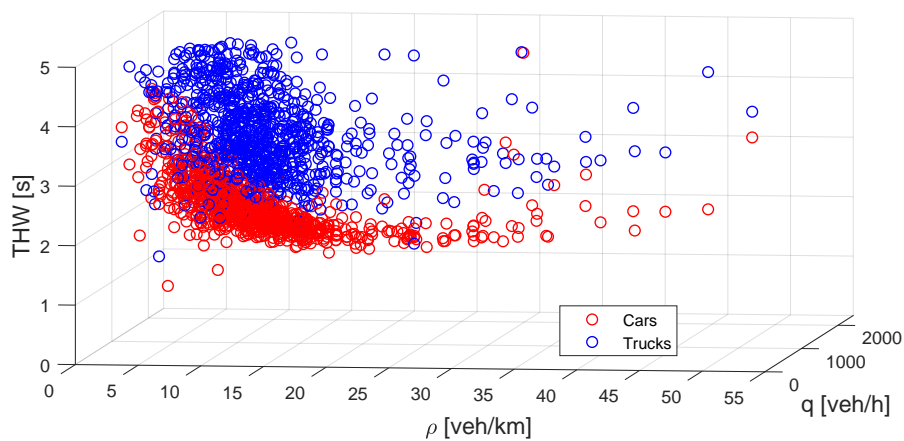
02:22 Drone Image Data For Joint Micro- and Macroscopic Road Traffic Analysis



■ **Figure 8** Fundamental diagrams: The dashed lines in a) and c) represent the transition from free flow to bounded flow up to stop-and-go traffic. Jammed traffic was not recorded within one-minute periods.

Figure 8 b) depicts the average flow rate over the velocity. Two different velocity ranges can be found at the same flow rate, which yields to the classification of stable and unstable traffic flow. Figure 8 c) depicts the velocity over the traffic density. The graph is approximately divided into the three traffic types (dashed lines). The average velocity decreases slightly up to a certain density threshold. Above the threshold the velocity drops significantly. Most of the recorded data is located in the transition zone of free flow to bounded traffic, while still retaining a velocity of $v \geq 100 \text{ km/h}$. Figure 8 d) depicts a three-dimensional plot, combining all three variables.

Finally, Figure 9 depicts the relationship between the flow rate, density and THW from all available time frames, aggregated to one minute slices. Overall, the THW is considerably lower for cars than for trucks. The datapoints for cars show a clear tendency towards smaller THW with an increasing density and flow rate. Trucks have a larger spread and are less correlated to the density and flow. The THW values reach a bottom of 1.2s around the transition zone between bounded traffic and stop-and-go traffic ($\rho \approx 30 \text{ veh/km}$). From there on a tendency towards larger THW values can be recognized. Drivers keep in average a certain minimum distance of some meters to their leader vehicle during stop-and-go traffic. This yields larger averaged THW values due to the low velocity.



■ **Figure 9** The traffic density, flow rate and the average THW over all recorded time frames, aggregated to one minute slices.

4.4 Lane changes and load per lane

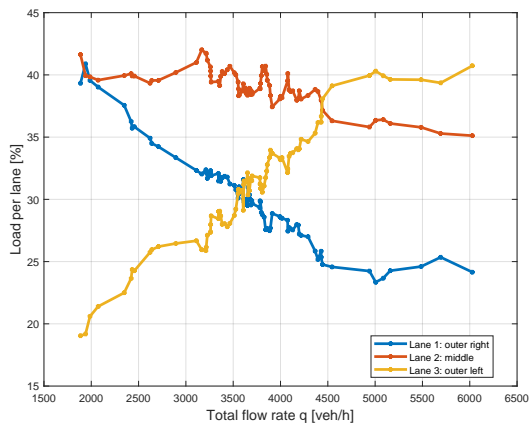
For the design of highways, it is important to know to which proportions a lane is used and where the saturation point is reached. From the perspective of traffic safety, accidents caused by lane changes are ranked third, behind speed violations and short distances [1, 2]. Since these two indicators are important for traffic analysis, the load per lane and lane changes in dependence to the traffic flow are examined and compared to literature.

Figure 10 depicts the relative lane load depending on the traffic flow rate of the road. Here, the load per lane is defined as the fraction of the overall number of vehicles on the road segment. Higher traffic flow rates increase the proportion of vehicles on the outer left lane. Smaller flow rates increase the proportion to the outer right lane, since drivers are legally obliged to drive on the right lane, if not driving at a higher velocity than the leader vehicle. In order to reach velocities above the truck velocity limits at higher flow rates, drivers switch to the middle and with further increasing traffic flow switch more often to the left lane. This observation holds as long as the traffic does not reach a stop-and-go state. The conclusions from [14] coincide by means of the tendency, that higher flow rates increase the proportion of vehicles on the left lane, decrease it on the middle lane slightly and decreases it heavily on the right lane. The publication [14] proposes flow rates per lane as depicted in Figure 11. The right lane saturates at around $q = 800 \text{ veh/h}$ and the left lane at around $q = 2600 \text{ veh/h}$ [14, 29].

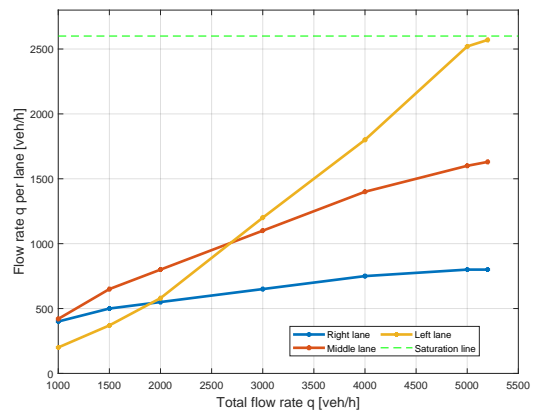
Figure 12 and Figure 13 depict the normalized number of lane changes over the flow rate and traffic density. The data points depicted in the figures are aggregated values of one-minute slices.

Publication [14] proposes a maximum lane change rate at around 3500 vehicles per hour for a 3 lane motorway. In [14] the lane change rate is described by step functions which form a triangular shape. Similar to [14], a triangular fit for the highD data set is depicted in Figure 12. It encloses at least 97% of all depicted one-minute slices for the upper and lower road. The results on the lane change rate coincide with the observation in [14]. Regarding the dependency on the traffic density, the lane change rate increases up to around 10-12 vehicles per kilometer and lane [14], according to the findings depicted in Figure 13.

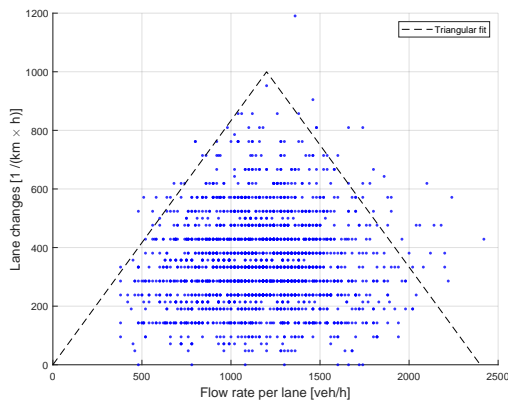
The results from this chapter confirm that the drone-based approach can provide valid estimates on macroscopic traffic data. Furthermore, the results can be interpreted in the context of microscopic variables as well. This is demonstrated with the relationship of the THW to the traffic density and flow.



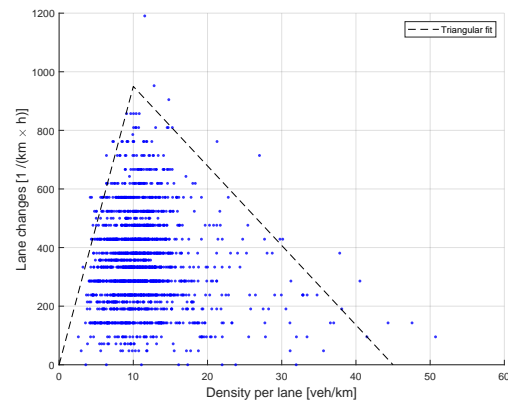
■ **Figure 10** The relative lane load depending on the averaged traffic flow rate. The curves are processed with a smoothing filter. For this plot only roads with 3 lanes are considered.



■ **Figure 11** A schematic comparison of the average load for a 3 lane motorway from [14]. The right lane saturates around $q = 800 \text{ veh/h}$, the left lane around $q = 2600 \text{ veh/h}$.



■ **Figure 12** The number of lane changes per lane, hour and kilometer depending on the averaged traffic flow rate per lane.



■ **Figure 13** The number of lane changes per lane, hour and kilometer depending on the traffic density.

5 Conclusions

The technological improvement of drones combined with the capabilities of computer vision methods, namely deep neural networks, open a new chapter for applications in road traffic surveillance and analysis. Observation campaigns can be carried out at every place, without the need of installing and maintaining roadside units. First projects attempt to expand the field of view by operating with swarms of drones and thereby capture whole city districts. The accuracy of the state estimation by means of aerial imagery is suitable for many trajectory based applications as well. This enables the individual analysis of objects, as well as observing the interaction between traffic participants.

In this work, a methodology for generating microscopic traffic data is proposed. More precisely, the state of vehicles and the resulting trajectories are estimated. The method is validated with experiments and reaches consumer-grade INS precision for the vehicle state estimation. A detailed look into the error sources, limitations and runtimes complements the proposed method.

Additionally, a new approach for reducing the average computational runtime, named Kalman Proposals, are presented. With Kalman Proposals, the runtime for object detection can be decreased by up to 20% with minor losses on the detection performance.

Furthermore, a publicly available data set for highway traffic is analyzed in order to validate the drone based approach for macroscopic traffic analysis. The statistics derived in this work are compared to related publications, where data was collected and processed with alternative approaches. The analysis confirms the applicability for capturing macroscopic traffic data as well.

In conclusion, the results of this paper underline the versatility of drone based image processing for synchronous macro- and microscopic road traffic analysis.

References

- 1 Daten und Fakten: Autobahn-Unfälle, 2010.
- 2 Runter vom Gas, Unfallursachen, 2018.
- 3 Mujahid Abdulrahim. On the dynamics of automobile drifting. *SAE Mobilus*, April 2006. doi:10.4271/2006-01-1019.
- 4 Seyed Majid Azimi, Eleonora Vig, Reza Bahmanyar, Marco Körner, and Peter Reinartz. Towards multi-class object detection in unconstrained remote sensing imagery, 2018. arXiv:1807.02700.
- 5 S. Bae and K. Yoon. Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1218–1225, 2014. doi:10.1109/CVPR.2014.159.
- 6 Emmanouil Barmponakis and Nikolas Geroliminis. On the new era of urban traffic monitoring with massive drone data: The pneuma large-scale field experiment. *Transportation Research Part C: Emerging Technologies*, 111:50–71, 2020. doi:10.1016/j.trc.2019.11.023.
- 7 Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded Up Robust Features. In *Computer Vision – ECCV*, 2019.
- 8 A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Uppcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, 2016.
- 9 Peter J. Bickel, Chao Chen, Jaimyoung Kwon, John Rice, Erik van Zwet, and Pravin Varaiya. Measuring Traffic. *Statistical Science*, 22(4), 2007.
- 10 Laura Bieker-Walz. Wie kann eine Verkehrssimulation den Rettungsdienst unterstützen? In *WAW DLR.Open II*, oktober 2017. URL: <https://elib.dlr.de/114841/>.
- 11 Ilker Bozcan and Erdal Kaya. Au-air: A multimodal unmanned aerial vehicle dataset for low altitude traffic surveillance. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.
- 12 Antonia Breuer, Jan-Aike Termöhlen, Silviu Homoceanu, and Tim Fingscheidt. Opended: A large-scale roundabout drone dataset. In *Proceedings of International Conference on Intelligent Transportation Systems*, September 2020.
- 13 Bundesverband der Deutschen Luftverkehrswirtschaft. Analyse des deutschen Drohnenmarktes. URL: <https://www.bdl.aero/de/publikation/analyse-des-deutschen-drohnenmarktes/>.
- 14 Fritz Busch. Spurbelastungen und Häufigkeit von Spurwechseln auf einer dreispurigen BAB-Richtungsfahrbahn. *ATZ - Automobiltechnische Zeitschrift*, June 1984. doi:10.1007/s35148-012-0485-x.
- 15 CATT (University of Maryland). Traffic Flow Measures Implementation Guide, 2008. URL: http://www.catt.umd.edu/sites/default/files/documents/traffic_flow_measure_guidelines_v8.pdf.
- 16 Bo-Chiuan Chen and Feng-Chi Hsieh. Sideslip angle estimation using extended kalman filter. *Vehicle System Dynamics - VEH SYST DYN*, 46, September 2008. doi:10.1080/00423110801958550.
- 17 C. Feichtenhofer, A. Pinz, and A. Zisserman. Detect to track and track to detect. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3057–3065, 2017. doi:10.1109/ICCV.2017.330.
- 18 B Grienshields. The photographic method of studying traffic behavior. In *Proceedings of the Thirteenth Annual Meeting of the Highway Research Board*, 1933.
- 19 Giuseppe Guido, Vincenzo Gallelli, Daniele Rogano, and Alessandro Vitale. Evaluating the accuracy of vehicle tracking data obtained from Unmanned Aerial Vehicles. *International Journal of Transportation Science and Technology*, 2016.
- 20 Hall, Fred. TRAFFIC STREAM CHARACTERISTICS, 1996.
- 21 Jörg Haus and Norbert Lauinger. Optische gitter: Die abbildung der realität – 75 jahre berührungslose dynamische meßtechnik auf der basis optischer gitter. *Laser Technik Journal*, 4:43–47, April 2007. doi:10.1002/latj.200790155.
- 22 Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.
- 23 Dirk Helbing. Traffic and related self-driven many-particle systems. *Rev. Mod. Phys.*, 73:1067–1141, December 2001. doi:10.1103/RevModPhys.73.1067.
- 24 R.E. Kalman. A new approach to linear filtering and prediction problems, 1960.
- 25 Kai Kang, Hongsheng Li, Tong Xiao, Wanli Ouyang, Junjie Yan, Xihui Liu, and Xiaogang Wang. Object detection in videos with tubelet proposal networks. *2017 IEEE Conference on Com-*

- puter Vision and Pattern Recognition (CVPR), July 2017. doi:10.1109/cvpr.2017.101.
- 26 Kai Kang, Wanli Ouyang, Hongsheng Li, and Xiaogang Wang. Object detection from video tubellets with convolutional neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. doi:10.1109/cvpr.2016.95.
 - 27 Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images, 2017.
 - 28 Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems. In *IEEE 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.
 - 29 Stefan Krauß. Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics.
 - 30 F. Kruber, E. Sánchez Morales, S. Chakraborty, and M. Botsch. Vehicle Position Estimation with Aerial Imagery from Unmanned Aerial Vehicles. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.
 - 31 H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart.*, pages 83–97, 1955.
 - 32 C. Kyrkou, G. Plastiras, T. Theocharides, S. I. Venieris, and C. Bouganis. DroNet: Efficient convolutional neural network detector for real-time UAV applications. In *Design, Automation Test in Europe Conference Exhibition (DATE)*, 2018.
 - 33 Lambda Labs. Deep Learning GPU Benchmarks, 2021. URL: <https://lambdalabs.com/gpu-benchmarks>.
 - 34 Qingpeng Li, Lichao Mou, Qizhi Xu, Yun Zhang, and Xiao Xiang Zhu. R³-Net: A Deep Network for Multi-oriented Vehicle Detection in Aerial Images and Videos. *CoRR*, 2018. arXiv:1808.05560.
 - 35 H. Lietz, Petzoldt T., M. Henning, J. Haupt, G. Wanielik, J. Krems, H. Mosebach, J. Schomerus, M. Baumann, and U. Noyer. Methodische und technische Aspekte einer Naturalistic Driving Study. *FAT Schriftenreihe*, 2011.
 - 36 Thomas Lillesand, Ralph W. Kiefer, and Jonathan Chipman. *Remote Sensing and Image Interpretation*, volume 5. Wiley, 2003.
 - 37 T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature Pyramid Networks for Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.
 - 38 Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV*, 2014.
 - 39 J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.
 - 40 A.L. Majdik, C. Till, and D. Scaramuzza. The zurich urban micro aerial vehicle dataset. *International Journal of Robotics Research*, 2017.
 - 41 L. Mou and X. X. Zhu. Vehicle Instance Segmentation From Aerial Image and Video Using a Multitask Learning Residual Fully Convolutional Network. *IEEE Transactions on Geoscience and Remote Sensing*, 2018.
 - 42 Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for uav tracking. In *Computer Vision – ECCV 2016*. Springer International Publishing, 2016.
 - 43 Lutz Neubert. *Statistische Analyse von Verkehrsdaten und die Modellierung von Verkehrsfluss mittels zellulärer Automaten*. PhD thesis, Universität Duisburg, 2000.
 - 44 Pix4D SA. Example projects - real photogrammetry data (<https://www.pix4d.com>). URL: <https://www.pix4d.com>.
 - 45 J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.
 - 46 Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems 28*. Curran Associates, Inc., 2015.
 - 47 A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese. Human Trajectory Prediction In Crowded Scenes. In *European Conference on Computer Vision (ECCV)*, 2016.
 - 48 Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, 2015.
 - 49 E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, 2011.
 - 50 E. Sánchez Morales, F. Kruber, M. Botsch, B. Huber, and A. García Higuera. Accuracy Characterization of the Vehicle State Estimation from Aerial Imagery. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.
 - 51 Patrick Schneider, Martin Butz, Christian Heinzemann, Jens Oehlerking, and Matthias Woehrle. Scenario-based threat metric evaluation based on the highd dataset. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.
 - 52 Dieter Schramm, Manfred Hiller, Roberto Bardini, et al. *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*, volume 124. Springer, 2010.
 - 53 Marc René Zofka Tobias Fleck, Sven Ochs and J. Marius Zöllner. (accepted) robust tracking of reference trajectories for autonomous driving in intelligent roadside infrastructure. In *Intelligent Vehicles Symposium (IV) 2020*, oktober 2020.
 - 54 Philip Torr and A. Zisserman. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, June 2000.
 - 55 Verband der TÜV e.V. Merkblatt 751, 2008.
 - 56 Bas Vergouw, Huub Nagel, Geert Bondt, and Bart Custers. *Drone Technology: Types, Payloads, Applications, Frequency Spectrum Issues and Future Developments*, pages 21–45. T.M.C. Asser Press, 2016. doi:10.1007/978-94-6265-132-6_2.

- 57 L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3119–3127, 2015. doi: 10.1109/ICCV.2015.357.
- 58 Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clause, Maximilian Naumann, Julius Kümmerle, Hendrik Königshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv*, 2019. arXiv:1910.03088.
- 59 Pengfei Zhu, Longyin Wen, Xiao Bian, Ling Haibin, and Qinghua Hu. Vision Meets Drones: A Challenge. *arXiv preprint*, 2018. arXiv:1804.07437.