

Michał FERENC, Marek SZCZEPAŃSKI
Politechnika Śląska

AUTOMATIC CAMERA CALIBRATION FOR UAV-BASED VEHICLE SPEED ESTIMATION

Summary. Vehicle speed estimation from UAV video requires accurate camera calibration and scale determination. This paper presents a near real-time vision-based pipeline that estimates vehicle speed using automatic calibration derived from checkerboards, crosswalk markings, or vehicle dimensions. Experiments on real UAV recordings show that checkerboard- and crosswalk-based calibration achieve speed estimation accuracy within approximately ± 2 km/h.

AUTOMATYCZNA KALIBRACJA KAMERY DO ESTYMACJI PRĘDKOŚCI POJAZDÓW NA PODSTAWIE NAGRAŃ UAV

Streszczenie. Estymacja prędkości pojazdów na podstawie nagrań UAV wymaga dokładnej kalibracji kamery oraz wyznaczenia skali obrazu. W pracy przedstawiono działający w czasie zbliżonym do rzeczywistego system wizyjny wykorzystujący automatyczną kalibrację opartą na szachownicy, oznakowaniu przejść dla pieszych lub wymiarach pojazdów. Badania przeprowadzone na rzeczywistych nagraniach UAV wykazały, że metody wykorzystujące szachownicę i przejścia dla pieszych pozwalają osiągnąć dokładność estymacji prędkości na poziomie około ± 2 km/h.

1. Introduction

Accurate vehicle speed estimation plays an important role in road safety analysis, traffic management, and transportation planning. Conventional speed measurement systems, including radar devices and fixed speed cameras, provide reliable measurements but require dedicated infrastructure and are typically limited to predefined locations [9]. Unmanned aerial vehicles (UAVs) equipped with onboard cameras provide a flexible alternative, enabling rapid deployment, easy relocation of measurement sites, and observation of larger road segments [4, 6].

A typical UAV-based speed estimation pipeline consists of image stabilization, vehicle detection, object tracking, camera calibration, and speed estimation. Image stabilization is required to compensate for residual platform motion caused by wind and flight control corrections. Existing approaches include optical-flow-based motion compensation [6], feature-based registration methods such as SIFT [12], and global alignment techniques based on homography estimation [5]. More recently, intensity-based registration methods, including Enhanced Correlation Coefficient (ECC) maximization, have demonstrated high robustness in UAV image processing applications [3, 10].

Vehicle detection in aerial imagery is currently dominated by convolutional neural networks, particularly models from the YOLO family, due to their favorable balance between accuracy and computational efficiency [7]. Object tracking is typically performed using tracking-by-detection frameworks combined with Kalman filtering or more advanced multi-object tracking methods [6, 11, 2]. Speed estimation itself has been addressed using both photogrammetric reconstruction approaches [8] and simpler pixel-to-meter scaling techniques [2].

Despite significant progress, camera calibration and scale estimation remain major sources of uncertainty in UAV-based speed measurement systems. Manufacturer-provided camera parameters are often incomplete or dependent on the selected video mode, while manual calibration procedures reduce the flexibility of field deployments [4]. Therefore, this work focuses on automatic camera calibration using visual references available directly in recorded UAV video. The proposed method combines YOLO-based vehicle detection, Kalman-filter-based trajectory smoothing, and three alternative calibration strategies based on checkerboard patterns, pedestrian crossings, and vehicle dimensions. The objective is to provide a practical and easy-to-deploy solution for vehicle speed estimation from UAV imagery.

2. Hardware

The proposed method was designed for near real-time operation on consumer-grade hardware. The dataset was acquired using DJI Air 2S and DJI Mini 4 Pro UAVs, while the evaluation videos were recorded using a DJI Mini 2 SE. The DJI Air 2S and DJI Mini 4 Pro support video recording at up to 4K resolution and 60 FPS, whereas the DJI Mini 2 SE supports recording at up to 2.7K resolution and 30 FPS. Consequently, the evaluation videos were acquired at a lower spatial resolution than the training dataset, although their resolution remained higher than the input size of the evaluated YOLO models. Video recordings from the onboard RGB cameras constituted the primary input to the processing pipeline. All model training, evaluation, and application testing were performed on a Windows 10 workstation equipped with a 12th generation Intel Core i5 processor, 32 GB DDR5 RAM, and an NVIDIA RTX 3060 GPU with 12 GB VRAM.

3. Methodology

This section presents the proposed vision-based pipeline for vehicle speed estimation from UAV video. The method consists of video stabilization, vehicle detection and tracking, camera calibration, and speed estimation. It operates on standard RGB video acquired during typical UAV flights without requiring dedicated calibration targets or additional onboard sensors.

3.1. Overview of the Processing Pipeline

The proposed processing pipeline is illustrated in Figure 1. It consists of video preprocessing, vehicle detection and tracking, automatic camera calibration, and vehicle speed estimation.

Video frames are first decoded and preprocessed to ensure geometric consistency. Vehicles are subsequently detected using a YOLO-based model and tracked across consecutive frames using Kalman filtering. The core of the proposed method is the automatic

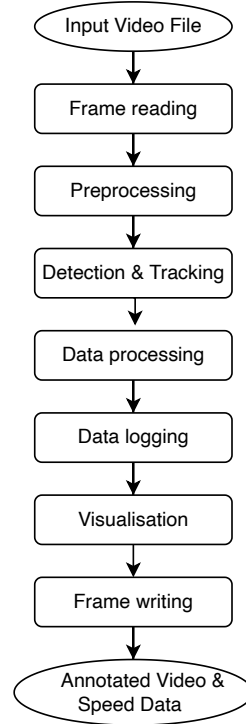


Fig. 1. Overview of the proposed vehicle speed estimation pipeline.

camera calibration module, which estimates the image scale from visual references and enables three-dimensional position reconstruction under planar ground assumptions. Finally, vehicle speeds are estimated, stored in CSV format, and visualized in the output video.

3.2. Vehicle Detection and Tracking

Vehicle detection and tracking are key components of the proposed pipeline, as both camera calibration and vehicle speed estimation rely on accurate and temporally stable object localization. This section presents the selected detection model, the training dataset, and the evaluation procedure used to identify the final configuration.

YOLOv11 was selected because it provides pre-trained weights for oriented bounding box (OBB) detection required by the proposed calibration procedure.

Five model variants (Nano, Small, Medium, Large, and Extra Large) were trained using input resolutions of 720×720 , 1000×1000 , 1280×1280 , and 1920×1920 pixels. Training was performed for up to 300 epochs using the standard Ultralytics `train` routine with early stopping.

The trained models were evaluated on the test set in terms of localization accuracy and computational performance. Since a one-pixel localization error may correspond to an apparent speed error of approximately 5 km/h, precise estimation of bounding box position, dimensions, and orientation is essential for reliable speed estimation and automatic camera calibration.

The dataset was created from photographs captured using consumer-grade UAVs and manually annotated for OBB-based training. It consists of 642 training images, 124 validation images, and 83 test images acquired across different seasons, times of day, and flight altitudes. The annotations cover eight object classes: rooftops, cars, people, trucks,

photovoltaic panels, chimneys, tractors, and crosswalks. Due to the characteristics of the surveyed areas, chimneys and cars are the most frequent classes, while people, trucks, and tractors are underrepresented.

The evaluated YOLO configurations were compared in terms of OBB localization accuracy and average processing time to identify the most suitable detector for vehicle speed estimation and automatic camera calibration. The quantitative results are summarized in Tables 1 and 2. Table 1 reports mean squared errors (MSE) of the OBB center coordinates, dimensions, and rotation angle together with the average processing time, while Table 2 presents the corresponding normalized scores used for comparative evaluation.

Table 1

Mean squared errors (MSE) of bounding box parameters for models trained using different input image resolutions.

Model size	Training resolution [px]	MSE					Avg. frame processing time [ms]*
		Δx_c [px ²]	Δy_c [px ²]	Width [px ²]	Length [px ²]	Rotation [deg ²]	
Nano	720	28.18	28.46	104.45	1177.42	981.17	20.96
	1000	13.04	7.51	49.17	579.66	958.86	12.96
	1280	12.91	8.67	54.02	707.72	1077.85	16.43
	1920	12.04	71.41	112.32	1314.31	982.95	28.22
Small	720	14.71	21.45	68.93	1031.37	985.24	12.85
	1000	12.51	14.51	160.02	1086.94	1056.12	16.93
	1280	24.82	8.27	234.32	584.12	915.03	21.30
	1920	-	-	-	-	-	-
Medium	720	14.01	50.72	365.67	2220.83	945.34	17.99
	1000	36.10	72.85	660.37	464.89	1050.51	28.75
	1280	21.59	11.43	122.57	1093.87	871.99	41.87
	1920	-	-	-	-	-	-
Large	720	48.15	16.02	111.13	613.12	950.94	21.73
	1000	21.94	11.17	372.75	604.95	993.60	35.40
	1280	-	-	-	-	-	-
	1920	-	-	-	-	-	-

* Benchmark environment: Windows 10, Intel i5 (12th gen), 32 GB DDR5, NVIDIA RTX 3060 (12 GB VRAM).

Table 1 summarizes the localization accuracy and processing performance of the evaluated models. Increasing the training image resolution did not consistently improve localization accuracy, indicating that higher input resolution alone does not guarantee better detector performance. Missing entries in Tables 1 and 2 correspond to configurations that exceeded the available GPU memory of the test platform during training.

Since even a one-pixel localization error may translate into an apparent speed error of several kilometers per hour, accurate estimation of object position was considered the primary criterion for model selection.

Table 2 presents normalized localization errors, normalized average frame processing times, and the overall score used for model ranking. Although intended only as a relative ranking criterion, the overall score indicates that the Nano models trained with 1000×1000 and 1280×1280 px images provide the best trade-off between localization accuracy and computational efficiency. In contrast, the Small model trained at 720×720 px achieved the shortest processing time but exhibited noticeably larger

Table 2

Normalized localization errors, normalized frame processing-time values, and overall scores of the evaluated YOLO models.

Model size	Training resolution [px]	Normalized errors					Normalized avg. frame processing time	Overall score
		Δx_c	Δy_c	Width	Length	Rotation		
Nano	720	0.59	0.39	0.16	0.53	0.91	0.50	3.07
	1000	0.27	0.10	0.07	0.26	0.89	0.31	1.91
	1280	0.27	0.12	0.08	0.32	1.00	0.39	2.18
	1920	0.25	0.98	0.17	0.59	0.91	0.67	3.58
Small	720	0.31	0.29	0.10	0.46	0.91	0.31	2.39
	1000	0.26	0.20	0.24	0.49	0.98	0.40	2.57
	1280	0.52	0.11	0.35	0.26	0.85	0.51	2.60
	1920	-	-	-	-	-	-	-
Medium	720	0.29	0.70	0.55	1.00	0.88	0.43	3.85
	1000	0.75	1.00	1.00	0.21	0.97	0.69	4.62
	1280	0.45	0.16	0.19	0.49	0.81	1.00	3.09
	1920	-	-	-	-	-	-	-
Large	720	1.00	0.22	0.17	0.28	0.88	0.52	3.07
	1000	0.46	0.15	0.56	0.27	0.92	0.85	3.21
	1280	-	-	-	-	-	-	-
	1920	-	-	-	-	-	-	-

localization errors.

Consequently, the two best-performing Nano models were selected for subsequent evaluation on real traffic video sequences. Under ideal conditions, both configurations are capable of processing video streams at frame rates exceeding 60 FPS.

3.3. Camera Calibration

Camera calibration establishes the relationship between image coordinates and real-world distances and is therefore essential for accurate vehicle speed estimation. Three calibration strategies were investigated: a conventional checkerboard-based approach, calibration using the average width of passenger cars, and a scene-based method exploiting standardized pedestrian crosswalk markings. The latter two estimate camera parameters directly from information available in the recorded video, eliminating the need for dedicated calibration targets during data acquisition.

3.3.1. Checkerboard-Based Calibration

Camera calibration commonly relies on planar calibration patterns, such as checkerboards or fiducial markers, with precisely known geometry [13, 1]. This approach is widely adopted due to its simplicity, robustness, and implementation in standard computer vision libraries, including OpenCV.

Unlike professional cameras, consumer-grade UAV cameras often differ substantially between still-image and video modes because of sensor cropping, digital stabilization, resolution changes, and aspect-ratio modifications. Consequently, calibration parameters obtained from still images are generally not applicable to video data.

To address this issue, a dedicated calibration script extracts representative frames directly from the recorded video. Frames captured during takeoff, landing, or abrupt

camera motion are discarded, and the remaining frames are uniformly downsampled to a maximum of 200.

Checkerboard corners are detected using `findChessboardCorners`, refined with `cornerSubPix`, and passed to the `calibrateCamera` routine to estimate the intrinsic camera parameters and lens distortion coefficients.

3.3.2. Vehicle Width-Based Calibration

Passenger vehicles exhibit relatively limited variability in width due to manufacturing standards and infrastructure constraints. According to a large-scale survey published in 2020, the average passenger-car width is approximately 1.813 m. With a sufficiently large number of observations, the mean estimated width is expected to converge toward this value.

Given the average vehicle width measured in image space, its nominal real-world value, and the UAV flight altitude, the effective focal length and the camera intrinsic matrix can be estimated. However, the bounding box dimensions reported by the YOLO detector vary with object position because of perspective effects and vehicle orientation.

To compensate for these effects, the detected vehicle width is corrected using the geometric relation:

$$w \simeq \frac{w_{\text{detected}} \cos(\alpha)}{\cos(\alpha) + \frac{H}{W} \sin(\alpha) \sin(\theta)} \quad (1)$$

Here, w denotes the corrected vehicle width in pixels, w_{detected} is the detected width, H and W are the nominal vehicle height and width, α is the angle between the optical axis and the line connecting the camera center to the vehicle, and θ is the acute angle between the bounding box major axis and the image centerline. The angle α is estimated from the assumed camera field of view (FOV):

$$\begin{cases} \tan\left(\frac{FOV}{2}\right) = \frac{w_{\text{img}}}{Z} \\ \tan(\alpha) = \frac{r}{Z} \end{cases} \quad (2)$$

$$\alpha = \tan^{-1} \left(\tan\left(\frac{FOV}{2}\right) \frac{r}{w_{\text{img}}} \right) \quad (3)$$

where w_{img} is the image width, Z is the distance to the image plane, and r is the object's horizontal distance from the centre of the image. Under correct assumptions regarding the camera field of view and vehicle dimensions, the width correction introduces an error of approximately 0.8%. Corrected widths are averaged over consecutive frames and subsequently across all detected vehicles to obtain a robust scale estimate.

3.3.3. Crosswalk-Based Calibration

Road markings provide a reliable geometric reference because of their legislative standardization and widespread presence. Pedestrian crosswalks are particularly suitable, as they consist of alternating 0.5 m white stripes separated by 0.5 m gaps.

Crosswalks are automatically detected using the trained object detection model. Samples are discarded if nearby objects overlap the enlarged crosswalk axis-aligned bounding box (AABB) by more than 10%. The selected region is cropped, converted to grayscale, histogram-equalized, rotated to align the stripe pattern vertically, and median-filtered.

Stripe segmentation is performed using adaptive thresholding followed by vertical opening and horizontal closing to isolate stripe regions and suppress small artifacts.

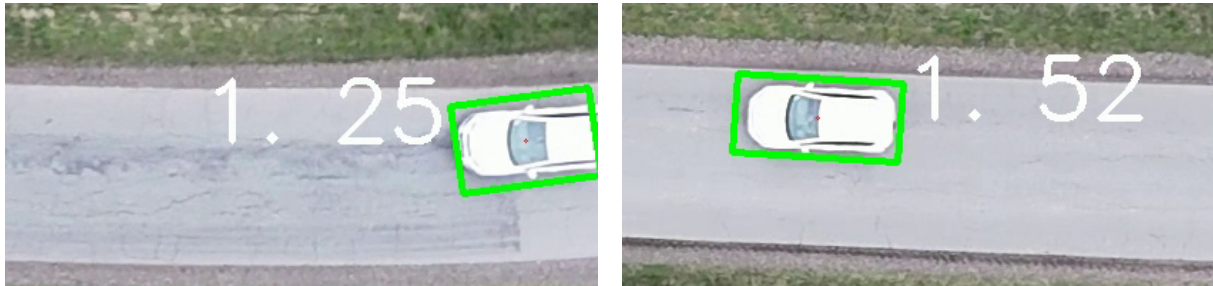
Stripe spacing is estimated from three randomly sampled image rows. After rejecting edge artifacts, the measured distances are averaged and compared with the nominal stripe width of 0.5 m to estimate the effective focal length and reconstruct the camera intrinsic matrix.

3.4. Speed Estimation

The UAV is assumed to hover with its camera in nadir orientation. The camera coordinate system is defined with the optical axis aligned with the Z -axis, the X -axis pointing horizontally, and the Y -axis vertically. Assuming a planar ground, 3D vehicle positions are reconstructed by back-projecting image coordinates using the inverse intrinsic matrix and the known UAV altitude. The resulting trajectories are subsequently smoothed using a Kalman filter.

As vehicles enter or leave the image frame, only part of the object is visible. Consequently, one side of the bounding box follows the true vehicle motion, whereas the opposite side remains constrained by the image boundary, resulting in underestimated object displacement and speed. This effect, illustrated in Figures 2 and 3, is further amplified during the initial frames of a new track by the Kalman filter convergence. Partial visibility may also lead to incorrect estimation of the vehicle width, adversely affecting camera calibration.

To mitigate these effects, object measurements are discarded whenever the axis-aligned bounding box (AABB) lies closer than half of its current width or height to the corresponding image boundary. Consequently, only fully visible vehicles contribute to camera calibration and speed estimation.



(a) Partially visible vehicle

(b) Fully visible vehicle

Fig. 2. Effect of partial vehicle visibility near the image boundary.

4. Evaluation

4.1. Reference Measurements

Vehicle speed estimates were initially verified against a vehicle-mounted GPS speedometer. The reference vehicle was driven at approximately 50 km/h whenever traffic conditions allowed. During intervals of steady motion maintained by cruise control, the manually calibrated speed estimates agreed with the GPS measurements within the 1 km/h resolution of the speedometer for both evaluated models (figures 4 and 5).

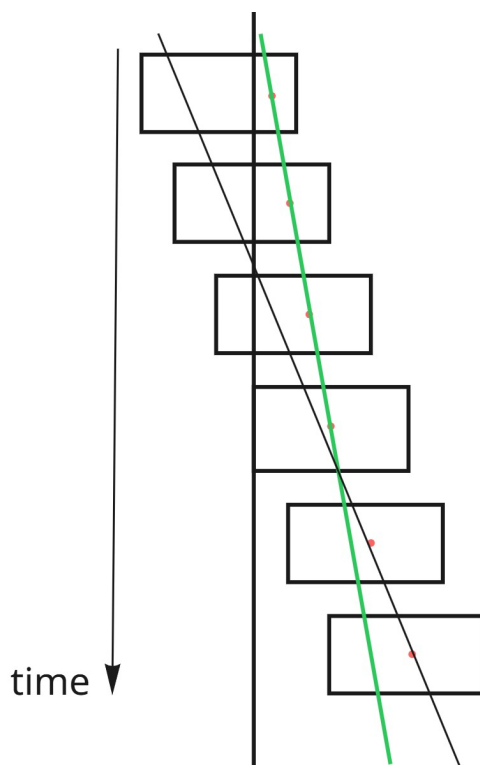
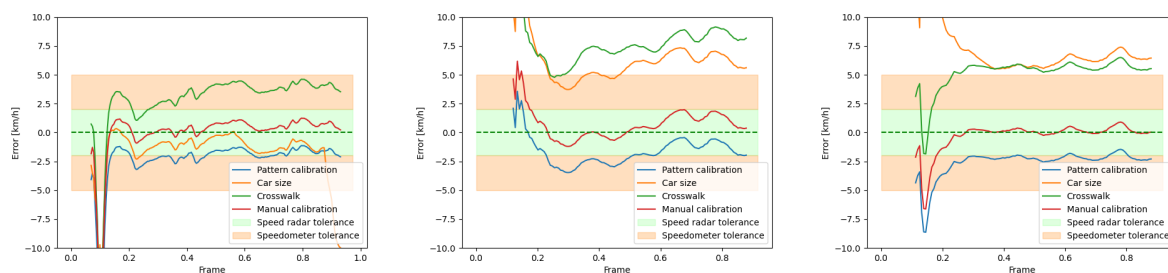


Fig. 3. Geometric explanation of the frame boundary effect.

Because of the limited resolution and short-term fluctuations of consumer-grade GPS measurements, the manually calibrated results were adopted as the reference base-line for the quantitative evaluation of the proposed calibration methods.

Figures 4 and 5 compare the evaluated calibration methods with the manually calibrated reference. For both detector configurations, the checkerboard-based calibration provides the closest agreement with the reference, followed by the crosswalk-based method, while the vehicle width-based calibration exhibits the largest deviations.



(a) Video 440, altitude 80 m

(b) Video 445, altitude 50 m

(c) Video 446, altitude 50 m

Fig. 4. Target vehicle speed estimation error using model trained on 1000x1000 px images.

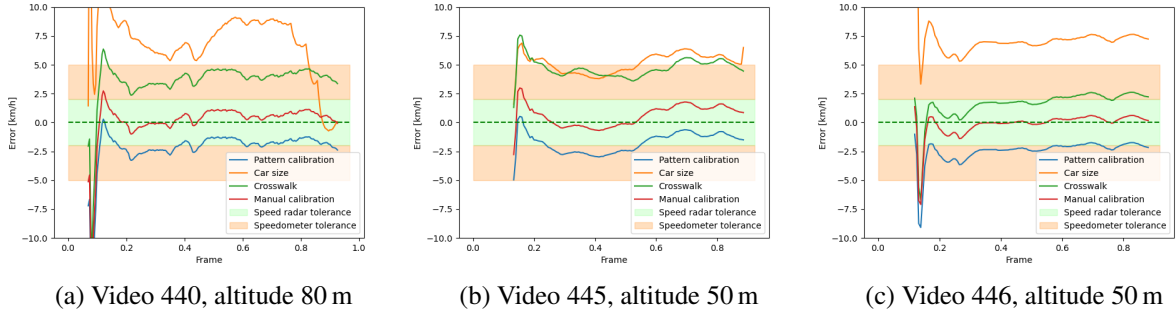


Fig. 5. Target vehicle speed estimation error using model trained on 1280x1280 px images.

4.2. Results

Figures 6 and 7 show the temporal evolution of vehicle speed estimation errors for the two selected YOLO Nano configurations. Both checkerboard-based and crosswalk-based calibration exhibit stable convergence, with errors remaining predominantly within ± 2 km/h after the initial transient period. In contrast, the vehicle width-based calibration shows higher variability and occasional error spikes, reflecting its greater sensitivity to detection noise, object orientation, and simplifying geometric assumptions.

For all evaluated calibration strategies, convergence is typically achieved after approximately 100 frames. Subsequently, the estimated speeds remain stable, indicating that the proposed approach is suitable for near real-time vehicle speed estimation from hovering UAV platforms.

Tables 4 and 3 summarize the error statistics computed after the convergence period and confirm the trends observed in the temporal analysis. Checkerboard-based calibration achieves the lowest errors, closely followed by the crosswalk-based approach, with mean absolute errors typically around 1 km/h.

Vehicle width-based calibration performs less consistently, reaching a maximum MAE of 17.73 km/h. Nevertheless, for several recordings, particularly those acquired from higher flight altitudes, the mean error remains below approximately 5 km/h. These results indicate that vehicle width-based calibration can serve as a fallback solution when explicit calibration references are unavailable, although its accuracy is lower than that of the remaining methods.

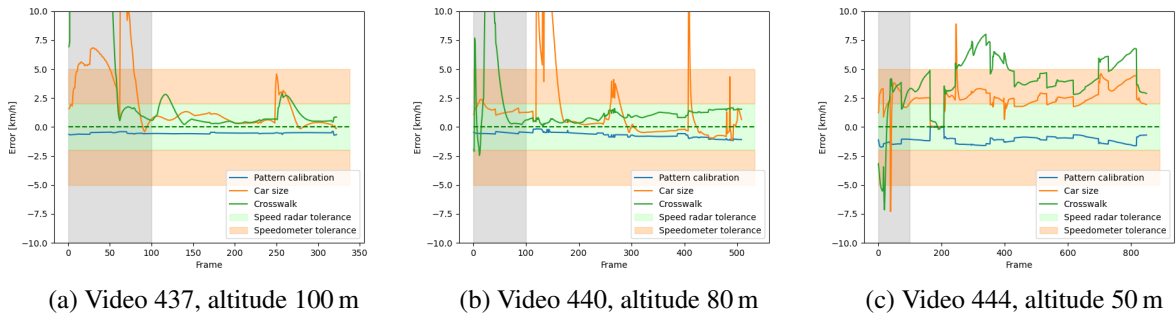


Fig. 6. Speed estimation errors for the YOLO Nano model trained on 1000x1000 px images.

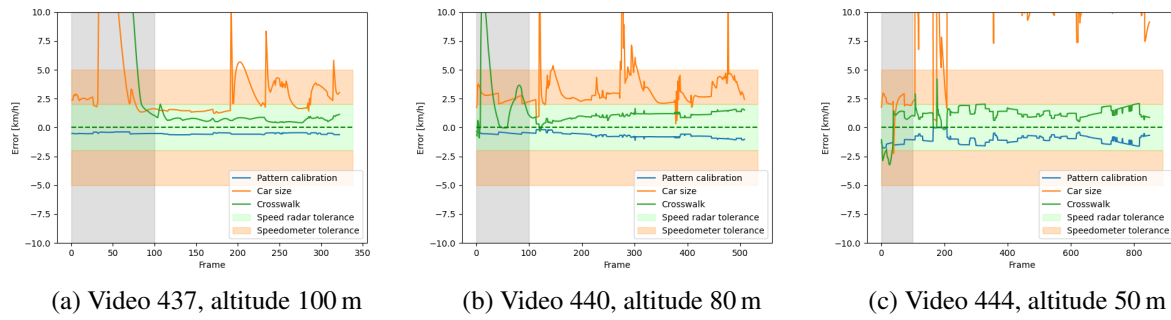


Fig. 7. Speed estimation errors for the YOLO Nano model trained on 1280×1280 px images.

Table 3

Speed estimation errors relative to the manually calibrated reference after the 100-frame convergence period for the YOLO Nano model trained on 1000×1000 px images.

Altitude [m]	Video	Checkerboard		Vehicle width		Crosswalk	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
100	0436	0.73	0.74	0.97	1.56	0.93	1.01
	0437	0.54	0.54	2.57	2.95	0.72	0.76
	0438	0.79	0.79	2.97	3.13	1.19	1.26
	0439	0.97	0.98	3.59	3.78	1.07	1.09
80	0440	0.70	0.73	3.29	3.68	1.03	1.08
	0442	1.55	1.56	2.58	2.97	2.07	2.09
	0443	1.36	1.37	4.10	4.52	2.25	2.27
50	0444	0.98	1.05	17.73	23.55	1.28	1.36
	0445	1.43	1.45	3.62	3.71	2.32	2.38
	0446	1.08	1.13	3.76	4.00	0.81	0.86
	0447	1.60	1.60	4.78	4.81	1.46	1.52
	0449	1.66	1.67	9.14	12.52	1.06	1.07
	0450	2.08	2.09	4.46	5.01	1.50	1.55

5. Conclusions

This article presented a vision-based pipeline for vehicle speed estimation from UAV video, with particular emphasis on automatic camera calibration using scene-derived references. The proposed approach operates on standard RGB recordings without requiring dedicated calibration targets or auxiliary onboard measurements.

Experimental results show that the checkerboard-based and crosswalk-based calibration methods achieve stable speed estimation accuracy within approximately ± 2 km/h after a short convergence period. In contrast, vehicle width-based calibration is more sensitive to detection noise and geometric assumptions, although its performance improves with increasing flight altitude.

The proposed pipeline operates close to real time on consumer-grade hardware and is suitable for practical traffic monitoring applications. The presented methodology

Table 4

Speed estimation errors relative to the manually calibrated reference after the 100-frame convergence period for the YOLO Nano model trained on 1280×1280 px images.

Altitude [m]	Video	Checkerboard		Vehicle width		Crosswalk	
		MAE	RMSE	MAE	RMSE	MAE	RMSE
100	0436	0.72	0.72	1.22	1.39	0.86	0.88
	0437	0.53	0.54	0.84	1.14	0.89	1.16
	0438	0.84	0.85	1.3	1.43	1.07	1.13
	0439	0.94	0.94	2.0	2.18	0.67	0.67
80	0440	0.71	0.75	1.84	4.14	0.96	1.04
	0442	1.75	1.77	3.28	3.77	3.25	3.28
	0443	1.37	1.38	2.05	2.31	3.08	3.35
50	0444	1.06	1.12	2.53	2.72	4.44	4.74
	0445	1.42	1.45	3.32	3.4	4.24	4.31
	0446	1.08	1.14	3.8	4.18	2.39	2.51
	0447	1.6	1.6	3.83	3.83	1.68	1.8
	0448	2.13	2.15	7.13	7.96	-	-
	0449	1.76	1.76	6.89	6.91	-	-
	0450	2.08	2.09	6.45	6.56	2.39	2.4

provides a compromise between computationally demanding photogrammetric approaches and simple pixel-based methods, while maintaining sufficient accuracy for traffic analysis.

Future work will focus on extending the proposed calibration methods to non-nadir camera views and more complex road geometries.

REFERENCES

1. Bradski G., Kaehler A.: Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly Media, Inc. Sebastopol, CA 2008.
2. Chen Y., Zhao D., Er M. J., Zhuang Y., Hu H.: A novel vehicle tracking and speed estimation with varying UAV altitude and video resolution. *International Journal of Remote Sensing*, 2021, Vol.42(12), p. 4441–4466.
3. Evangelidis G. D., Psarakis E. Z.: Parametric image alignment using enhanced correlation coefficient maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, Vol.30(10), p. 1858–1865.
4. Guido G., Gallelli V., Rogano D., Vitale A.: Evaluating the accuracy of vehicle tracking data obtained from unmanned aerial vehicles. *International Journal of Transportation Science and Technology*, 2016, Vol.5(3), p. 136–151. *Unmanned Aerial Vehicles and Remote Sensing*.
5. Hartley R. I., Zisserman A.: Multiple View Geometry in Computer Vision. Cambridge University Press Cambridge, UK second edition 2004.

6. Li J., Chen S., Zhang F., Li E., Yang T., Lu Z.: An adaptive framework for multi-vehicle ground speed estimation in airborne videos. *Remote Sensing*, 2019, Vol.11(10).
7. Redmon J., Divvala S., Girshick R., Farhadi A.: You only look once: Unified, real-time object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, p. 779–788.
8. Shahbazi M., Simeonova S., Lichti D., Wang J.: Vehicle tracking and speed estimation from unmanned aerial videos. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2020, Vol.XLIII-B2-2020, p. 623–630.
9. Soole D. W., Watson B. C., Fleiter J. J.: Effects of average speed enforcement on speed compliance and crashes: A review of the literature. *Accident Analysis Prevention*, 2013, Vol.54, p. 46–56.
10. Szczepański M.: Improving UAV-Based Detection of Low-Emission Smoke with an Advanced Dataset Generation Pipeline. *Remote Sensing*, 2025, Vol.17(6).
11. Wang L., Chen F., Yin H.: Detecting and tracking vehicles in traffic by unmanned aerial vehicles. *Automation in Construction*, 2016, Vol.72, p. 294–308.
12. Zhang C., Tang Z., Zhang M., Wang B., Hou L.: Developing a more reliable aerial photography-based method for acquiring freeway traffic data. *Remote Sensing*, 2022, Vol.14(9).
13. Zhang Z.: A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000, Vol.22(11), p. 1330–1334.