YOLOgraphy: Image Processing Based Vehicle Position Recognition

Ákos T. Köpeczi-Bócz[1(✉)], Tian Mi[2], Gábor Orosz[2,3], and Dénes Takács[1,4]

[1] Department of Applied Mechanics, Budapest University of Technology and Economics, Budapest 1111, Hungary
{kopeczi,takacs}@mm.bme.hu
[2] Department of Mechanical Engineering, University of Michigan, Ann Arbor, MI 48109, USA
{tianm,orosz}@umich.edu
[3] Department of Civil and Environmental Engineering, University of Michigan, Ann Arbor, MI 48109, USA
[4] HUN-REN-BME Dynamics of Machines Research Group, Budapest University of Technology and Economics, Budapest 1111, Hungary

Abstract. A methodology is developed to extract vehicle kinematic information from roadside cameras at an intersection using deep learning. The ground truth data of top view bounding boxes are collected with the help of unmanned aerial vehicles (UAVs). These top view bounding boxes containing vehicle position, size, and orientation information, are converted to the roadside view bounding boxes using homography transformation. The ground truth data and the roadside view images are used to train a modified YOLOv5 neural network, and thus, to learn the homography transformation matrix. The output of the neural network is the vehicle kinematic information, and it can be visualized in both the top view and the roadside view. In our algorithm, the top view images are only used in training, and once the neural network is trained, only the roadside cameras are needed to extract the kinematic information.

Keywords: Image processing · Vehicle dynamics · Machine learning

1 Introduction

The detection of vehicles via real-time image processing is a crucial task not just for autonomous vehicles but also for intersection management systems. However, identifying bounding boxes and extracting vehicle kinematic data (like position, yaw angle, velocity and yaw rate) with satisfying accuracy are challenging problems. In [2], the problem is approached through object detection and post-processing with a trained network. The training data is collected through GPS and LIDAR sensors. As presented in [4], it is also possible to estimate the distance of an object based on the size of the bounding box. Instead of focusing

G. Mastinu et al. (Eds.): AVEC 2024, LNME, pp. 392–398, 2024.
https://doi.org/10.1007/978-3-031-70392-8_56

on object classification and tracking (position and speed), we introduce a novel methodology to extract vehicle kinematic data (position, velocity, orientation and yaw rate) with the help of a neural network trained on high-precision data.

In our experiments, the vehicle kinematic information is collected at an intersection of the Mcity Test Facility at the University of Michigan, Ann Arbor. A DJI Phantom 4 Pro drone is sent above the intersection, and a standing vehicle at the intersection equipped with a camera facing forward serves as the roadside camera. The movement of a truck at the intersection is captured by both the drone camera (top view) and the roadside camera (roadside view). The experimental setup is detailed in [3]. The ground truth data, i.e., the high-precision top view bounding boxes, are obtained by classic image processing algorithms of the drone view recordings, as shown in the top row of Fig. 1.

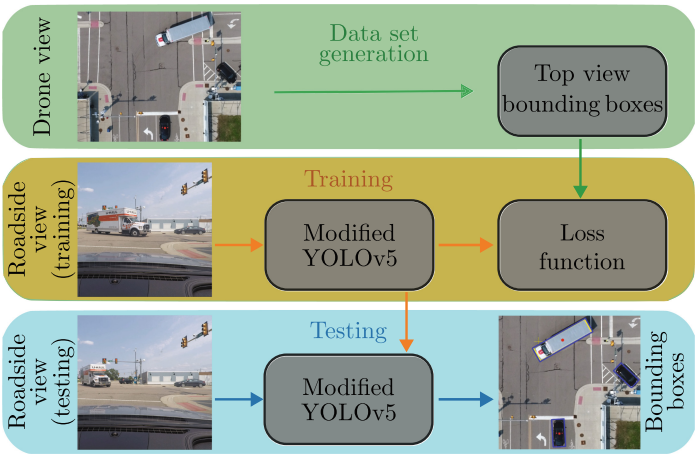

Fig. 1. Main steps of the kinematic data extraction method and the neural network training/testing.

2 Neural Network-Based Kinematic Data Extraction

The YOLOv5 [1] convolutional neural network serves as the basis of the proposed algorithm shown if Fig. 1. The structure of the original YOLOv5 network is modified to incorporate the discrepancy between the input (roadside view images) and the output (top view data), and the optimization method is modified to include kinematic information in the algorithm.

Originally, the YOLO network maps the bounding boxes onto the input image. Our goal, however, is not to obtain the bounding boxes on the roadside image but to reconstruct the top view bounding boxes of the vehicles. The top view perspective can be converted to the roadside view perspective with a homography transformation matrix, which can be obtained by selecting reference points from both perspectives. By decoupling the output space of the YOLOv5

Fig. 2. Sample output grid of the modified YOLOv5 network mapped on the (a) roadside and the (b) top view. The (a) roadside view is the input of the network.

from the input image and mapping the detection results on the top view, the network is trained to learn the homography transform connecting the top view images and roadside view images. We call this modified algorithm *YOLOgraphy*.

The original YOLOv5 output contains the center point, width, and height of the bounding boxes, while the orientation of the detected object is missing. To incorporate this, an additional parameter (representing the yaw angle) is added to the output of YOLOv5, and the loss function is extended with this parameter. Similar to the the original YOLOv5 algorithm, we detect the objects on three different grids (20×20, 40×40 and 80×80). Depending on the size of the object, the network detects them on different grid layers, larger-sized objects on the coarser grids and smaller objects on the finer grids. A sample grid (6×6) is shown in Fig. 2. For each grid-cell we have the output

$$\mathbf{p} = \left[p_1 \; b_x \; b_y \; w_x \; w_y \; \varphi\right]^\top , \tag{1}$$

where $p_1 \in [0, 1]$ is the confidence of an object being present in the given grid-cell, b_x and b_y denote the bounding box center point positions within the cell relative to the top left corner of the grid-cell. For example, $b_x = b_y = 0.5$ represents the centerpoint, while $b_x = b_y = 1$ corresponds to the bottom right corner of the grid-cell. Outputs $w_x \geq 0$ and $w_y \geq 0$ are the width and height of the bounding box as the scaling factors of the anchor box, and $\varphi = \psi/2\pi \in [0, 1]$ is the newly introduced output, the normalized yaw angle of the bounding box (vehicle).

Originally, YOLOv5 used different anchor boxes. In many cases, it is optimal to have horizontal/vertical rectangles and a square as three anchor boxes, for example, vertical for a pedestrian, horizontal for a vehicle, and square for a cyclist in side view. In our solution, the introduction of the yaw angle makes such differentiation of the anchor boxes redundant, namely, horizontal and vertical rectangles can be transformed into each other by a 90-degree rotation. Hence, our algorithm is based on a single anchor box.

The loss function in the YOLOv5 training consists of three main parts: the classification loss (*cls_loss*), the objectness loss (*obj_loss*), and the bounding

box regression loss (*box_loss*). The classification loss corresponds to the classification of the detected objects and is excluded from the study at this stage, although it could be considered in the future. The objectness loss shows the confidence of an object being present in a grid cell and is kept as it is. Lastly, the bounding box regression is modified to include the yaw angle. Originally, the box loss was calculated based on the *Intersection over Union* (IoU) algorithm, which divided the area of the intersection of the predicted and ground truth bounding boxes with the area of the union of the two (IoU is 1 if they overlap perfectly). When the bounding boxes are not aligned horizontally/vertically due to their non-zero yaw angles, the calculation of the intersection of the boxes is a more complex geometric problem. Thus, it may be computationally more efficient to use a simple mean-squared-error-based loss for the regression instead of the IoU. We introduce the weighted sum of *position loss*, *size loss* and the *yaw loss* as

$$loss = obj\_loss + \alpha \cdot pos\_loss + \beta \cdot size\_loss + \gamma \cdot yaw\_loss\,, \tag{2}$$

where α, β and γ are tuneable dimensionless hyperparameters, and are chosen to be 5, 1 and 10, respectively. These hand-tuned parameters and the mean-squared-error-based loss function perform well for the current experiments (see Sect. 3), but may be learned and modified. These results provide a proof of concept that we will extend with additional measurements in the future.

Two recordings (with the corresponding datasets) are used to train the neural networks separately, as the roadside camera has slightly different perspectives in the two cases, yielding different homography transformation matrices. For each dataset, the frames are mixed randomly, with 75% for training, 15% for validation, and 10% for testing. The neck and heads of the upper layer YOLOv5 network are trained, while the main convolutional layers are frozen during the training. This way, the network does not need to learn what a vehicle looks like but only learns how to place it on the top view plane. Overall, the networks perform well even for the test and validation sets, which were not used during training. In Fig. 3, the output of one experiment is visualized both in the roadside view panel (a) and the top view panel (b). The yellow bounding boxes are the ground truth obtained from drone measurements, and the blue bounding boxes are the YOLOgraphy output. The trajectory of the center point of the bounding box is shown in panel (c). The blue curve (network prediction) and the yellow curve (ground truth) have good agreement, which validates our approach.

3 Data Analysis

We compare the results of the trained YOLOgraphy output with the drone measurements (ground truth). The positions of one experiment are shown in Fig. 4(a), where the blue dashed line is the YOLOgraphy output, and the orange solid line is the ground truth. The two curves overlap with minimal difference throughout the whole measurement. Note that the visualization includes all the training, validation, and test frames. The yaw angles are compared in Fig. 4(b). While the two curves have good agreement, the YOLOgraphy output looks more

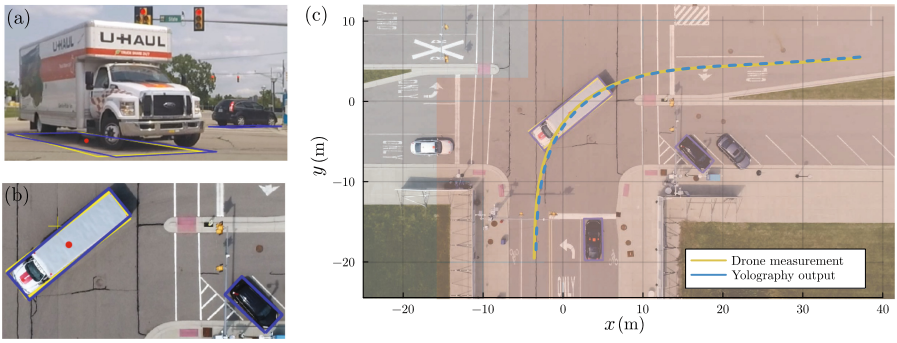

Fig. 3. YOLOgraphy output and its comparison with ground truth data: (a) roadside view, (b) top view, (c) trajectories.

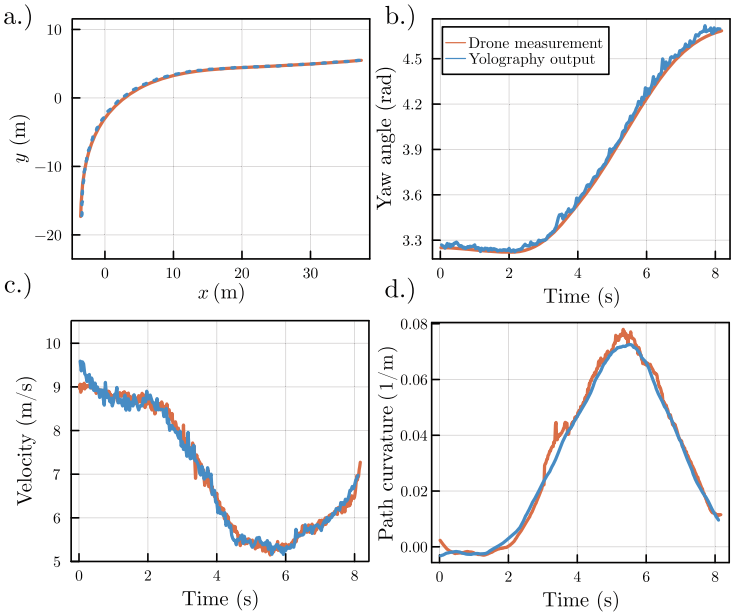

Fig. 4. Comparison of the drone measurements (ground truth) and the output of the trained Yolo network. (a) Trajectories, (b) yaw angles, (c) longitudinal velocities and (d) path curvatures for the rear axle center point (RAC).

noisy. This suggests that the YOLOgraphy struggles more with the prediction of the yaw angle, which is expected since it is a challenging task to predict the yaw angle based on the roadside view (cf. Fig. 2 and Fig. 3(a)).

In Fig. 4(c), the speed of the rear axle center (RAC) point is plotted, and since the RAC's velocity aligns with the yaw angle, this is referred to as longitudinal velocity. Between 5 and 6 s, the velocity hits the minimum, which is at the apex

of the turning. The velocity of the drone measurement and the YOLOgraphy output show a good agreement. Since we calculate these values with the method of finite differences, it is expected to amplify the noise.

In Fig. 4(d), the curvature of the rear axle center (RAC) is shown. Assuming that the RAC's heading angle is close to the yaw angle, the curvature is calculated from the yaw angle as $\kappa = \frac{\Delta\psi}{\Delta s}$ where $\Delta\psi$ is the change in the yaw angle between two adjacent frames, and Δs is the distance between two positions. To smooth the data, a Savitzky-Golay filter is applied. The curvature from YOLOgraphy is (somewhat surprisingly) smoother compared to the drone measurement.

4 Conclusion and Discussion

This work provides a proof of concept of YOLOgraphy, based on a modified YOLOv5 neural network. The roadside view images are mapped to the top view, and the neural network essentially learns the transformation during training. After training, YOLOgraphy can take the images from a roadside camera as input and output the kinematic data of vehicles on the top view plane. The validation results demonstrate the feasibility of the proposed method.

As future work, we plan to extend the dataset by additional measurements using a fixed-location roadside cameras. With the roadside view angle being more steep, the robustness of the detection can be potentially increased. Generally, the higher the camera is positioned, the easier it is to detect the vehicle. We may face a potential challenge that a large vehicle close to the roadside camera may obstruct its view. To overcome this issue, we plan to include input images from multiple roadside cameras from different angles. We also plan to introduce kinematic vehicle models to filter the results and predict vehicle trajectories.

Acknowledgments. The research reported in this paper was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences, the National Research, Development and Innovation Office under grant no. NKFI-146201, and the University of Michigan's Center for Connected and Automated Transportation through the US DOT grant 69A3552348305. The project supported by the Doctoral Excellence Fellowship Programme (DCEP) is funded by the National Research Development and Innovation Fund of the Ministry of Culture and Innovation and the Budapest University of Technology and Economics, under a grant agreement with the National Research, Development and Innovation Office. The research is partly supported by the Foundation for Mechanical Engineering Education.

References

1. Jocher, G.: YOLOv5 repository. Software. https://github.com/ultralytics/yolov5
2. Fang, L., Malm, O., Wu, Y., Xiao, T., Zhao, M.: Using machine learning to estimate road-user kinematics from video data. Project report, Chalmers University of Technology (2024)
3. Mi, T., Takács, D., Liu, H., Orosz, G.: Capturing the true bounding boxes: vehicle kinematic data extraction using unmanned aerial vehicles. J. Intell. Transport. Syst. 1–13 (2024)

4. Xu, X., Chen, X., Wu, B., Wang, Z., Zhen, J.: Exploiting high-fidelity kinematic information from port surveillance videos via a YOLO-based framework. Ocean Coastal Manag. **222**, 106117 (2022)