ORIGINAL ARTICLE

TopView: vectorising road users in a bird's eye view from uncalibrated street-level imagery with deep learning

Mohamed R. Ibrahim[1]

Received: 29 June 2023 / Accepted: 26 February 2025 / Published online: 24 March 2025

Abstract
Generating a bird's eye view of road users is beneficial for a variety of applications, including navigation, detecting agent conflicts, and measuring space occupancy, as well as the ability to utilise the metric system to measure distances between different objects. In this research, we introduce a simple approach for estimating a bird's eye view from images without prior knowledge of a given camera's intrinsic and extrinsic parameters. The model is based on the orthogonal projection of objects from various fields of view to a bird's eye view by learning the vanishing point of a given scene. Additionally, we utilised the learned vanishing point alongside the trajectory line to transform the 2D bounding boxes of road users into 3D bounding information. The introduced framework has been applied to several applications to generate a live Map from camera feeds and to analyse social distancing violations at the city scale. The introduced framework shows a high validation in geolocating road users in various uncalibrated cameras. It also paves the way for new adaptations in urban modelling techniques and simulating the built environment accurately, which could benefit agent-based modelling by relying on deep learning and computer vision.

Keywords Bird's eye view · Homography · Deep learning · Urban scenes

1 Introduction

Scene awareness across different views of a given scene represents an important subject not only in machine learning but also in studies related to understanding flows in cities and transports. Estimating a vectorised bird-eye view (BEV) representation of a given visual scene is useful for many real-world applications [1, 2], including navigation, motion prediction [3], robotics, or simply measuring distances and evaluating conflicts among road users whether it is a for understanding occupancy rates, social distancing, accidents, or near misses.

Understanding cities using computer vision, or more generally through machine learning, has gained the interest of planners and urbanists in the last few years [4]. However, the obstacles to combining both machine-based approaches and their outputs with the most frequent planning tool (maps) persist. If only the content of street-level imagery could automatically blend and localise to maps, this might benefit the utility of machine learning in cities at scale. Most recently, substantial progress has been made to create methods that can learn to generate a BEV representation for autonomous driving [1, 2]. However, most of these methods either tend to rely on a given camera's parameters (intrinsic and extrinsic) for calibration making it limited for generalisation when cameras' parameters are unknown [5–11], or generate an image-based representation (i.e. image-to-image transition) [1, 12, 13] makes the yielding outcome useful only for few applications, excluding the ability to generate

trajectories, or measuring distances without the need for an extra step of vectorising the raster output. Here we introduce a simple approach but a powerful one for generating a vector BEV representation from uncalibrated images which makes it applicable for both known and unknown cameras' parameters, including internet data, and CCTV feeds whereas other current methods face shortcomings. The introduced approach, known as TopView, relies on learning the vanishing point of a given scene, while geometrically estimating the BEV vector space of a given space and the 3D representation of a given object from its 2D boxes estimated from the backbone of the model.

This study significantly extends the current methodologies used in computer vision for urban analytics by introducing a novel framework that supports robust and scalable analysis without the need for camera calibration. Our major contributions are detailed as follows: We propose a novel method for estimating bird's eye view (BEV) that operates independently of the camera's intrinsic and extrinsic parameters. This facilitates the application of BEV generation techniques to a broader range of uncalibrated images sourced from varied devices and viewpoints, thus making advanced visual analytics accessible in environments where camera calibration is impractical or unknown. Our approach simplifies the process of mapping and geolocating road users' trajectories from uncalibrated 2D images to the geographic coordinate system through the innovative application of vanishing points to infer depth and scale. This significantly enhances the accuracy of object placement in virtual space, providing crucial data for traffic management and urban planning. Extending our method to video streams, we introduce a spatiotemporal representation of moving objects, encapsulating them as streams of tokens that capture dynamic changes over time. This provides a detailed and continuous narrative of object movements, which is invaluable for traffic flow analysis and surveillance. Additionally, we ensure the privacy of individuals by anonymising the representation of road users in both still images and video streams. This adherence to privacy laws and ethical standards makes our method suitable for sensitive environments where user consent may be unattainable. Finally, the practicality of our method is demonstrated through applications on diverse datasets, including CCTV footage from urban traffic systems (see Fig. 1). These applications showcase the method's robustness across different settings and its capability to provide actionable insights for real-world challenges.

2 Background

We are not aware of any method for estimating BEV based solely on learning a vanishing point without knowing the cameras' matrix or providing key points for a given perspective. However, our introduced method links with several knowledge domains:

2.1 Object detection

Object detection is a cornerstone of computer vision with applications ranging from autonomous driving to security surveillance [14–16]. Traditionally, object detection relied on manual feature extraction combined with machine learning algorithms, using techniques like histogram of oriented gradients (HOG) [17] and scale-invariant feature transform (SIFT) [18] alongside classifiers such as support vector machines (SVM) [19]. However, the advent of deep learning has revolutionised the field, introducing more sophisticated and effective methods [14–16]. Convolutional neural networks (CNNs) now dominate object detection, facilitating powerful feature extraction and recognition capabilities. Significant milestones include the development of Region-based CNNs (R-CNN) and its iterations [20, 21], which efficiently localise and classify objects using region proposals. The you only look once (YOLO) framework and its successors [22–24]) simplify detection into a single regression problem, enhancing the speed and feasibility of real-time applications. Similarly, the single shot multibox detector (SSD) eliminates the need for proposal generation [25], directly predicting multiple bounding boxes and class probabilities, thus balancing speed with accuracy. These advancements in object detection pave the way for robust object localisation in bird's-eye view applications.

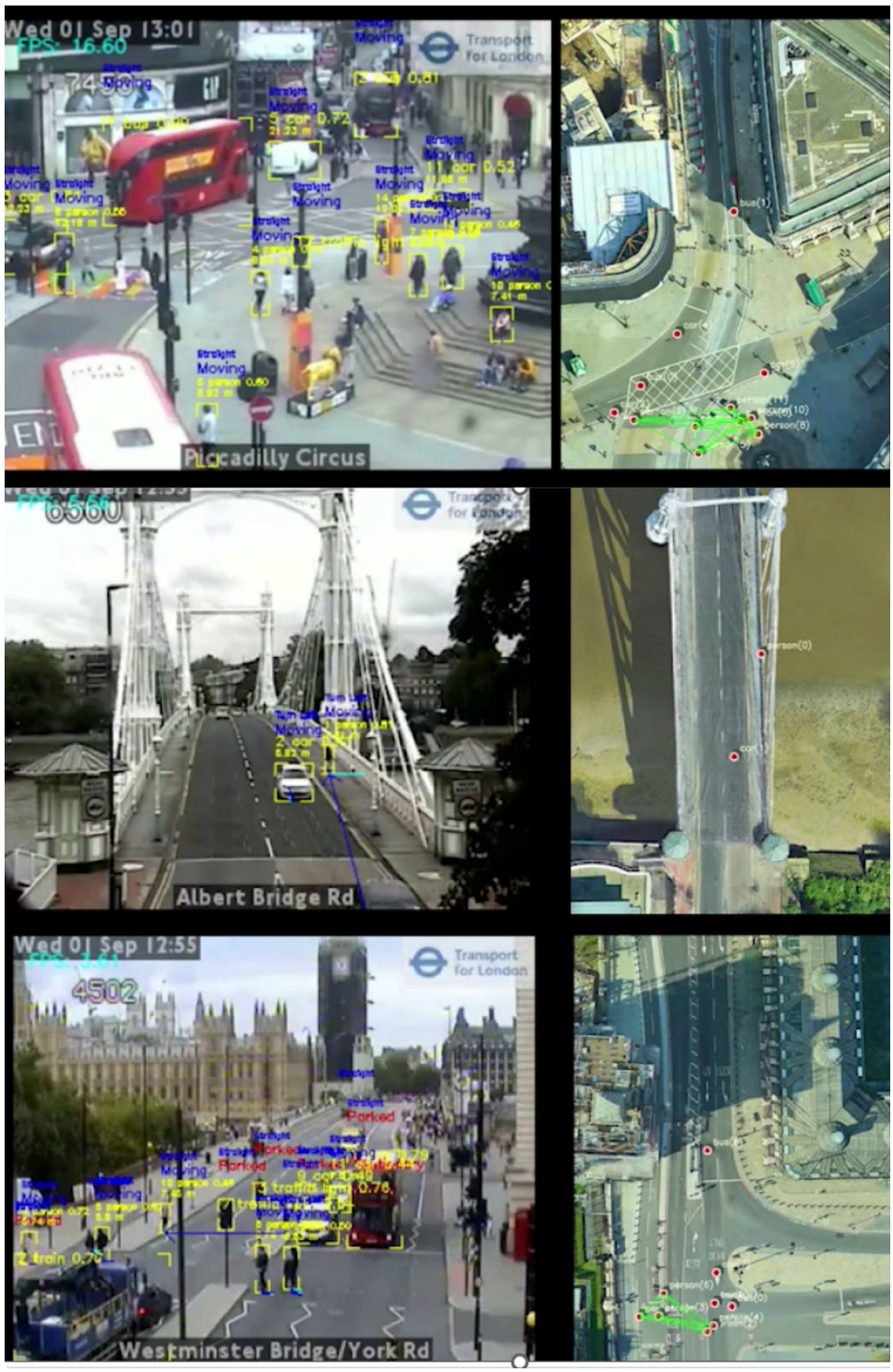

Fig. 1 Examples of different sites of various street layouts and their estimated BEV map on a Google map

2.2 Multiview awareness based on homography

In photogrammetry, moving from a given camera's coordinates system to the world coordinate system is achieved by knowing the camera matrix including both intrinsic and extrinsic parameters [6, 9–11, 26–28], as follows:

$$z_c \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = M \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix}, \text{ where } M = K[R \quad T] \tag{1}$$

given that M represents the camera matrix, K and $R\,T$ denote the intrinsic and extrinsic parameters of the camera, respectively. They are defined as:

$$K = \begin{bmatrix} \alpha_x & \gamma & c_x & 0 \\ 0 & \alpha_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \text{ where } \alpha_x = f \cdot m_x \text{ and } \alpha_y = f \cdot m_y \tag{2}$$

given that f represents the focal length of the camera in pixels, m_x and m_y represent the scale factors of relating pixels to distance, γ is the skew coefficient between the two axes of x and y, which is often equal to zero. Lastly, c_x and c_y denote the principle point.

$$[R \quad T] = \begin{bmatrix} R_{3\times3} & T_{3\times1} \\ 0_{1\times3} & 1 \end{bmatrix}_{4\times4} \tag{3}$$

Generally, extrinsic parameters represent the camera's position and heading in the world coordinates, where T represents the origin position of the world coordinate system defined in the camera coordinate system, and R represents the rotation matrix of the camera. After calibrating cameras to world coordinates and by relying on homography, we can move from one camera pose to another as follows:

$$\begin{pmatrix} z_i x_i' \\ z_i y_i' \\ z_i \end{pmatrix} = H \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}, \tag{4}$$

where $\text{dst}(i) = (x_i', y_i')$, $\text{src}(i) = (x_i, y_i)$, $i = 0, 1, 2, 3$

given that src and dst represent the coordinates of the quadrangle vertices in the camera view and world coordinates, respectively, (x_i, y_i) and (x_i', y_i') represent paired coordinate points in the camera and top-view planes, respectively. Lastly, H represents the homography or the transformation matrix that is defined as:

$$H = \begin{bmatrix} h_{00} & h_{01} & h_{02} \\ h_{10} & h_{11} & h_{12} \\ h_{20} & h_{21} & h_{22} \end{bmatrix} \tag{5}$$

where H is solved and calibrated by inputting the four paired points in the camera and top-view planes. Accordingly, by solving H, the detected object in the camera plane can be transformed into the top-view plane.

Several studies have provided approaches with slight changes for image calibration based on this method [6–9, 26, 28–30]. However, this method faces several shortcomings for automation and scalability such as (1) its requirement for calibrating cameras, limiting its usability to internet data, (2) the requirement for at least inputs of 4 points to represent the perspective to unwrap them in BEV image or corresponding points to estimate the transformation matrix, (3) Even when providing these points, without knowing the vanishing point, the BEV faces a high level of distortion for objects outside the bounds of the provided points.

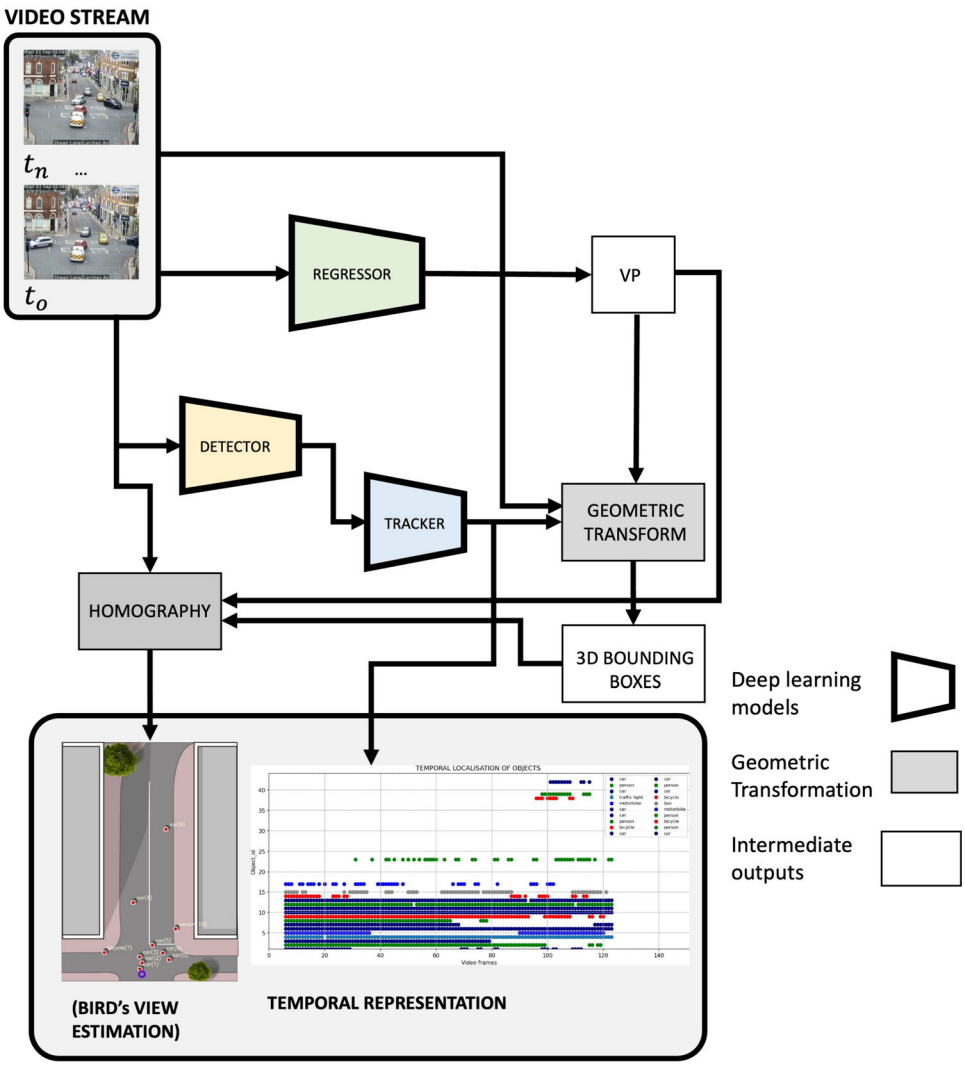

Fig. 2 The overall architecture of temporal localisation and spatial representation of scene objects from video streams

2.3 Geometric-based models for scene awareness

Combining both features of Geometric constraints and machine learning, several methods have been achieved to generate a BEV map from a camera view [1, 2, 5, 31, 32] relied on a CNN model to obtain a homography matrix by transforming a monocular camera input to a BVP map. However, this approach lacks vectorising road users. However, these methods still require the camera's model or several camera inputs lacking the ability of these models to apply directly to the ubiquitous uncalibrated images.

2.4 Multi-sensors fusion-based models

Several methods have focused on estimating a BEV map based on fusing both RGB images and actual LiDAR data [1, 33–35] or pseudo-LiDAR generated from depth estimation [36]. This approach relies on generating a BEV map by encoding both data sources with early fusion or post-feature extraction to guide the model to learning orthogonal features. For instance, [37] introduced a method for producing spatiotemporal birds-eye-view (BEV) representations from multi-camera footage and reasoning about multiple tasks collaboratively for vision-centric autonomous driving. [38] developed a model that learns the 3D representation of road users by fusing multiple camera inputs and extracting 2D and 3D feature streams based on the underlying geometric constraints

in the BEV. While this approach has shown strength in localising objects, it is limited to when LiDAR data is available, limiting its utility to certain applications primarily autonomous driving. Furthermore, it requires multiple calibration processes, as it requires not only calibrating the RGB images but also integrating the LiDAR data.

2.5 Image-to-image translation

While the objective of this research is to provide a vector representation rather than a raster one, it is worth mentioning that there have been several approaches that utilised image-to-image translation whether through adversarial learning or other approaches to generate an estimate of BEV map [1, 12, 13, 39]. These approaches included generating not only road users but also a semantic representation of the street layout [13, 40, 41] used encoder-decoder architecture to generate semantic segmentation for vehicle layouts from multiple camera sources. [42] developed a transformer-based model to extract the local road network layout in a BEV map based on a directed graph representation. [43] introduce a framework that includes a Hybrid Feature Transformation module that decouples learning and camera-based model approaches to output a semantic BEV map. [44] introduced a two-stage geometry-guided framework to generate a semantic BEV map from a monocular camera input. However, in practice, this approach, without geometric constraints, tends to provide noisy and unreliable outcomes when experimented with in unseen scenes, which we will report when comparing our results to existing methods.

3 Methodology

3.1 TopView framework

Humans tend to navigate by knowing the relationship between objects and avoiding obstacles instead of knowing the exact depth of each point in a given scene. Here we present a framework, called TopView, to generate temporal and BEV representations of road users when feed with sequential images or BEV representation alone when feed with single images. The topview framework only requires an image input without the need camera's model which makes it scalable to different data sources when camera parameters are alone. Figure 2 shows the architecture of the overall framework. After a given input, the framework comprises five sub-models that output a vector BEV map of road users only in case of a given image, or a vector BEV map and temporal localisation of the tracked road users. First, the framework takes a given input to pass it through a deep model to regress the vanishing point (VP) and the horizon of a given scene. Afterwards, an object detector with a tracker system is utilised to localise road users. The tracked road users alongside the VP and the horizon line are passed through a Geometric transformation module that aims to transform the 2D bounding representation of road users into 3D bounding boxes. Last, all outcomes are based on a Homographic module to transform the data into the spatial representation of the BEV map alongside temporarily localising road users and finetuning the generated temporal paths to account for spatial occlusion and objects re-identification related issues such as miss-matched objects' ids filtered based on the spatiotemporal patterns over multiple frame sequence. Besides the BEV map, the final output is a stream of paths representing road users' trajectories of multi-dimensional information such as object id, type, 3D bounding boxes, and stationary status.

VP model: By relying on geometric principles, vanishing points are a well-known concept in 3D vision research for their ability to estimate 3D structures from 2D images [45, 46]. Accordingly, we developed a model to estimate vanishing points from natural uncalibrated scenes. To learn the vanishing point of a given scene, we trained a deep learning model that takes a given image to output the X and Y coordinates of its vanishing point.

The model is built on the backbone of a truncated pre-trained model, and two additional branches of two Fully-Connected layers output each value of the coordinates of the vanishing point. The main reason for training our model is to ensure its accuracy and performance when used on the overall framework.

Object detection and tracking system: We relied on YOLO architecture [22, 24], in particular, we used a YoloV5m [23] as a backbone for TopView to detect road users including persons, cars, buses, trucks, bicycles, and motorbikes pre-trained on COCO dataset [47], in which we find the results are optimal in terms of accuracy and speed in deployment. In the case of sequential frames are given, we used DeepSort architecture [48] to track objects, which based on Sort algorithms [49] coupled with a deep learning model to handle object occlusion.

Geometric transformation: To define the confined 3D bounding box within a 2D box, we used a simple geometric transformation that utilised both trajectory lines and the vanishing point to estimate the 3D bounding box for a given road user within a scene. Figure 3 shows a few examples of different poses of a given object and the potential representation of the 3D bounding box that belongs to a given motion pose. We are aware of the efforts in the literature that aimed towards learning to estimate the 3D bounding box from the 2D bounding box of a given object [22, 36, 50–54]. However, despite learning the 3D representation from 2D representation, this method is still a given camera's model limiting its utility to other data sources. Here we show that we can achieve the same output with a simple Geometric transformation between both scenes, utilising scene information such as VP that we already automated and trajectory lines and therefore we will not need a given camera's model as presented by the learned method.

Algorithm 1 shows the algorithm for transforming 2D bounding boxes to 3D bounding boxes by heuristically estimating and bounding the 6DOF of a 3D bounding box in the given 2D bounding box. The algorithm estimates the geometry of a given 3D box by understanding the orientation of a given road user. This can be estimated by understanding the relationship between a given object's trajectory line, horizon, and the reference line derived from the location of the vanishing point in a given scene. The variables used in the algorithm are as follows: $Q_{\text{trajectory\_line}}$ is the set of points that form the trajectory line of the moving object. Each point in the set is represented as q. The vanishing point in the image, represented by its coordinates (vp_x, vp_y), is denoted as vp. The width of the image is represented by w. The coordinates that define the corners of the 2D bounding box around

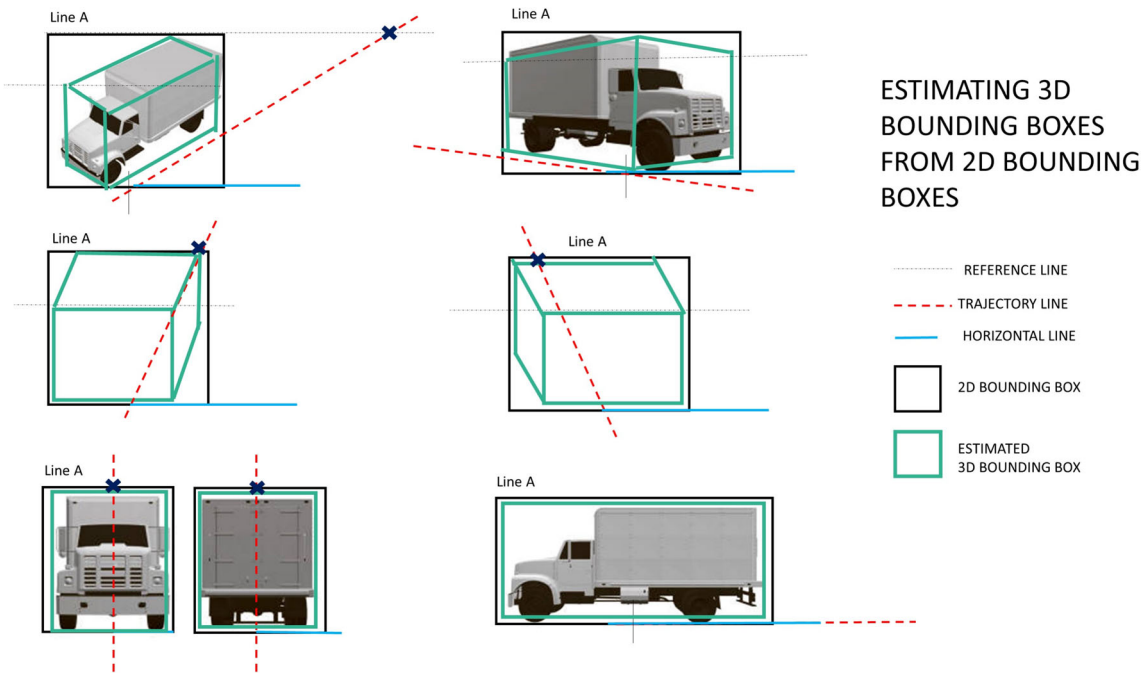

Fig. 3 Estimating 3D bounding boxes from 2D bounding boxes from different poses of a given road user

the detected object are x_1, x_2, y_1, y_2, where x_1 and x_2 are the x-coordinates of the left and right edges of the 2D bounding box, respectively, and y_1 and y_2 are the y-coordinates of the top and bottom edges, respectively. The variable F represents the estimated orientation of the car (moving object) and is initially set to "Undefined." The top edge of the 2D bounding box, L_A, is represented by the two points: (x_1, y_1) and (x_2, y_1). The midpoint of the top edge, denoted as M, is calculated as $M = \frac{(x_1+x_2)}{2}$. The algorithm checks for intersections between points on the trajectory line and the top edge L_A to determine the object's orientation relative to the midpoint M and the vanishing point vp. If no intersection is found, the orientation F is set to "side view." Based on these calculations and conditions, the estimated car orientation F and the 3D bounding box are the outputs. Our approach is effective in creating reliable and explainable bounding boxes inside the scene for non-stationary objects even without learning. This approach can also be utilised for stationary objects by replacing the trajectory line with the edge line of a given object; however, we leave this for future investigation.

Algorithm 1 Estimation of car orientation and 3D bounding box from trajectory lines

Input: $Q_{\text{trajectory\_line}}$ set of trajectory line points q
Input: vp vanishing point in image coordinates (vp_x, vp_y)
Input: w width of the image
Input: x_1, x_2, y_1, y_2 coordinates of the 2D bounding box
$F \leftarrow$ Undefined ▷ Initialize car orientation
$L_A \leftarrow [(x_1, y_1), (x_2, y_1)]$ ▷ Top edge of the bounding box
for all $q \in Q_{\text{trajectory\_line}}$ such that $q \cap L_A \neq \emptyset$ **do**
    $(q_x, q_y) \leftarrow q \cap L_A$ ▷ Intersection point of trajectory line with L_A
    $M \leftarrow \frac{x_1+x_2}{2}$ ▷ Midpoint of L_A
    if $\left|vp_x - \frac{w}{2}\right| \approx 0$ **then** ▷ Check if vanishing point is at image center
        if $q_x < M$ **then**
            $F \leftarrow$ "turning left"
        else if $q_x > M$ **then**
            $F \leftarrow$ "turning right"
        else
            $F \leftarrow$ "moving straight"
        end if
    else
        if $q_x < M - \left|vp_x - \frac{w}{2}\right|$ **then**
            $F \leftarrow$ "turning left"
        else if $q_x > M - \left|vp_x - \frac{w}{2}\right|$ **then**
            $F \leftarrow$ "turning right"
        else
            $F \leftarrow$ "moving straight"
        end if
    end if
end for
if $F =$ Undefined **then**
    $F \leftarrow$ "side view" ▷ Case when no intersection is found
end if
Output: F ▷ Estimated car orientation
Output: 3D bounding box ▷ Based on F and 2D box dimensions

Homography: To estimate a perspective plane grid, we create a horizontal line at the bottom of a given scene that is evenly subdivided by several points, and we utilise the detected VP to draw several lines from this VP to the bottom of each point at the above-mentioned horizontal line. We employed the four intersection points generated

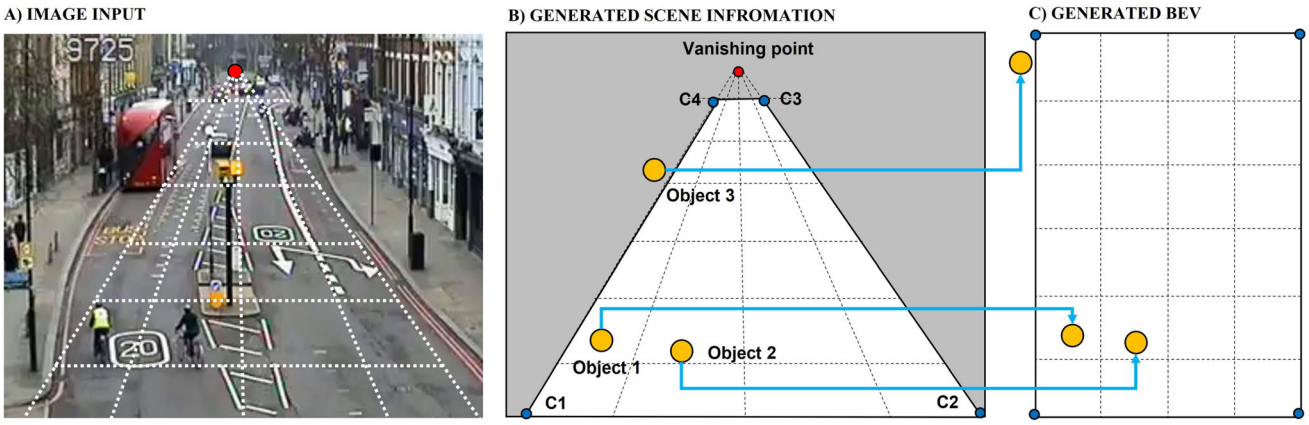

Fig. 4 Transforming a given image input to a Bird's eye view

by these radial lines originating by the VP point and the upper and lower horizontal lines inside the confines of a specific image's VP and lower half. As a result, we developed an automated four-point representation of a particular scene in which they match a corresponding four-point rectangle representation of the BEV map's vector space. We applied homography to connect road users' point coordinates in a particular image plane to the newly estimated vector space of the BEV map (see Fig. 4).

3.2 Objective loss and evaluations

For the VP model, we trained the model based on the logcosh loss function for each coordinate of the point. For small values of x and the big one, respectively, $\log(\cosh(x))$ is roughly equivalent to $(x^2)/2$ and $|x| - \log(2)$. Consequently, the logcosh function is mostly like the mean squared error while being less sensitive to the rare extremely inaccurate prediction.

For Object localisations, the objective loss is defined based on the weighted sum of the localisation loss (L_{loc}) and confidence loss (L_{conf}) for the introduced backbone of object detection to detect and localise humans as follows:

$$L(x, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \tag{6}$$

By cross-validating the model, the loss is set to 0 if $N = 0$ and α is set to 1 given that N is the default bounding box. Based on a Softmax loss for each class, the confidence loss is a cross-entropy loss (c). The default bounding box's centre (cx, xy), as well as its width (w) and height (h), establish the parameters of the predicted box (l), and the localisation loss is defined as a smooth loss between those parameters and the ground truth bounding box (g). It is defined as follows:

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^{N} \left[\sum_{m \in \{cx, xy, w, h\}} x_{ij}^{k} \text{smooth} L_1(l_i^m - \hat{g}_i^m) \right] \tag{7}$$

Evaluating a BEV map in a new dataset remains a challenge which poses an open question in the literature. Even though we have taken a heuristic approach to generate a BEV map, we evaluate the relationship between the different mapped objects after calibrating the image to its geolocation. We used Google Maps as a qualitative measure for verifying the localisations of the objects from the image plane to the real-world coordinate.

3.3 Implementations

Data processing: Each image used for training the VP model was normalised and downsized to a (300×300) grayscale image. To ensure the model focused on the geometric structures within the image, we applied the Canny edge detector filter to all images, preserving the edge details. The filtered images were then fed into the VP model for further processing.

VP model training: We employed a MobileNet architecture [55], pre-trained on ImageNet, as the backbone for our VP model. This involves removing the fully connected layers from the pre-trained MobileNet. We then applied a Global Average Pooling layer to determine the X and Y coordinates of the vanishing point. The network was extended with two separate branches, each designed to predict one of the coordinates (X or Y). Each branch consisted of two Fully-Connected layers with 100 neurons each, activated by a ReLU function to ensure nonlinearity. To prevent overfitting, a dropout layer with a dropout rate of 0.5 was added after each Fully-Connected layer. The final output layer of each branch consisted of a single neuron activated by a linear function. This architecture was trained for 100 epochs with a batch size of 256, using the Adam optimiser [56]. An early stopping callback was utilised to halt the training process if the model's loss did not improve for 5 consecutive epochs, thereby optimising training time and improving the model's generalisation capability.

Object detection and tracking system: For object detection, we used the YOLOv5m [23] architecture, pre-trained on the COCO dataset [47] for detecting several classes of road users, including pedestrians, cars, buses, trucks, bicycles, and motorbikes. Given sequential frames as input, object tracking was achieved using the DeepSORT algorithm [48], which combines the SORT algorithm for data association based on bounding box overlaps, and a deep appearance descriptor to maintain object identities during occlusion. This tracking mechanism is crucial for analysing the temporal localisation and movement patterns of objects.

Geometric transformation: The transformation from 2D to 3D bounding boxes leverages the geometric relationship between the object's position, its trajectory line, and the vanishing point. This transformation is automated using the algorithm described in Algorithm 1. In essence, the algorithm adjusts the 3D bounding box to fit within the 2D bounding box, derived from the predicted trajectory line and vanishing point, ensuring minimal error in the computed 3D orientation and dimensions.

Homography: We automated the computation of the homography matrix to map points from the image plane to the Bird's Eye View (BEV) vector space. This was done by delineating a horizontal line at the bottom of the image, subdividing it evenly, and drawing lines from the vanishing point to each of these subdivisions. We identified four intersection points between these lines and upper and lower horizontal lines, using them for the homography transformation. These points represent the bounding quadrilateral in the image plane, which was mapped to a corresponding rectangle representing the BEV space. This transformation facilitated the precise localisation of road users in the BEV map.

Table 1 Datasets for training VP model

Dataset	Number of images
London Streetview	46,281
Boston Streetview	38,215
Norway Streetview	84,447
Flickr	959
AVA	1315
TfL London CCTV	1359

3.4 Materials and experiments

We trained the model for inferring the vanishing point in a given scene by combining multiple open-access data sources to ensure the diversity of outdoor scenes. These datasets include six different types of datasets: (1) London Streetview [57], (2) Boston Streetview [57], (3) Norway Streetview [57], (4) Flickr [58], (5) AVA [58] and (6) TfL London CCTV [59] (See Table 1). The combined dataset comprises 172,576 images. The variety of these datasets ensures that the model sees data at different times of the day (for instance, Google Streetview data is only day-time whereas the rest contains nighttime dataset), different weather, and different fields of view. We randomly divided the dataset into training, validation, and testing groups in the following ratios: 80%, 10%, and 10%.

To further report on the robustness of our orthogonal-based approach to the overall framework, we implemented and tested our model in sequential datasets by utilising London CCTV video streams to verify geolocational references of objects in a given scene in a Google map. Nevertheless, we also apply our approach to a sample of internet data to scale its validity.

4 Results

Performance across different scenarios: We assessed the robustness of the proposed framework under various conditions, including different lighting environments and weather scenarios. The results of this evaluation are summarised in Table 2. We utilise mean average precision (mAP) to evaluate the accuracy of object detection and mean squared error (MSE) to measure the accuracy of vanishing point (VP) estimation. Table 2 illustrates that the framework performs exceptionally well under daylight conditions, achieving an object detection mAP of 90.1% and a VP estimation MSE of 0.038. These results demonstrate the system's high accuracy and low error in optimal lighting conditions. However, in nighttime scenarios, the performance slightly decreases, achieving an object detection mAP of 85.7% and a VP estimation MSE of 0.045. The decrease is expected due to the challenges presented by low-light environments. In adverse weather conditions, the framework faces the most significant challenges, with the object detection mAP dropping to 82.3% and the VP estimation MSE increasing to 0.060. These variations indicate the impact of environmental factors on detection and estimation accuracy. Overall, while the framework exhibits robustness across different scenarios, there is a noticeable performance decline in less favourable conditions, emphasising the need for algorithms that can adapt to such variability.

Estimating vanishing points: After training the VP model, Fig. 5 shows a sample of the predicted vanishing point (in red) and the ground truth one (in blue) in a variety of images with varying lighting conditions and fields of view obtained from various data sources. Despite the complexity of the presented scene layouts and their varying conditions, the trained model demonstrates good validation in grasping the orthogonal structure of a given scene and recognising its vanishing point. Based on these observed scenarios, there is still a small margin of error between the predicted and ground truth values of the vanishing points, particularly for the vanishing points' X coordinates.

Localising road users in a BEV map: Figure 6 shows an example of estimating a BEV map for a given video file from TfL London CCTV data. It shows first the transition of video frames to BEV maps with and without

Table 2 Robustness evaluation across different scenarios

Scenario	Object detection mAP (%)	VP estimation MSE
Daylight	90.1	0.038
Nighttime	85.7	0.045
Adverse weather	82.3	0.060

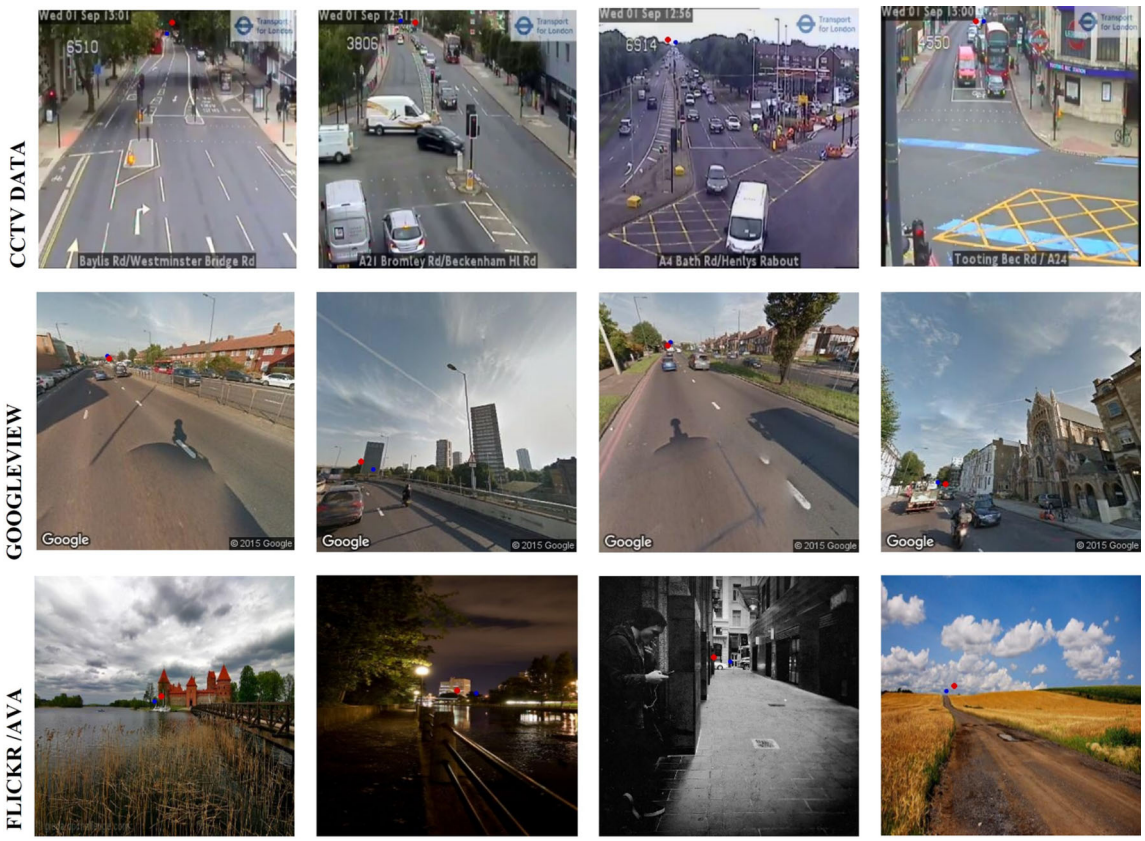

Fig. 5 Sample of estimated VP points in the various dataset types (Predicted point in red dot, ground truth in blue dot)

Google Maps. It shows a highly accurate localisation when qualifying the patterns of road users in Google Maps. It also shows the temporal localisations of road users as tokens, highlighting the appearance and disappearance through the time interval of the given video file. Furthermore, Fig. 7 shows another scene from the dataset, with a semantic representation of the street layout. This scene exemplifies the accuracy of localising a variety of road users, such as the bicycle on the pavement (left-hand side) and the pedestrian on the pavement (right-hand side), as well as the complexity of the many vehicles at the road junction, taking into account their stationary state. It is worth mentioning that we only highlight the semantic segmentation here without showing or assessing how to generate a given street layout, leaving it for future investigation on how to use this approach for the gamification of London CCTV video streams. In doing so, this gamified approach could be useful for several studies and modelling techniques, particularly agent-based modelling and data assimilation while protecting individual road users' privacy.

3D bounding box estimation: Figure 8 shows an example of how a 3D bounding box can be effectively inferred, deterministically without prior learning, based on geometry and the orientation of road users in a given scene, using only the introduced algorithm that constrains 3D bounding boxes based on a given input of a 2D bounding box, a trajectory line, and a vanishing point. The diagram also depicts road users' orientation and stationary state.

4.1 Comparing the results with state-of-the-art methods

The comparison evaluates various bird's eye view (BEV) generation methods across multiple dimensions, including their ability to localise road users, perform object tracking, and generate 3D bounding boxes. Tables 3 and 4 present a qualitative and quantitative comparison of various BEV generation methods. The qualitative

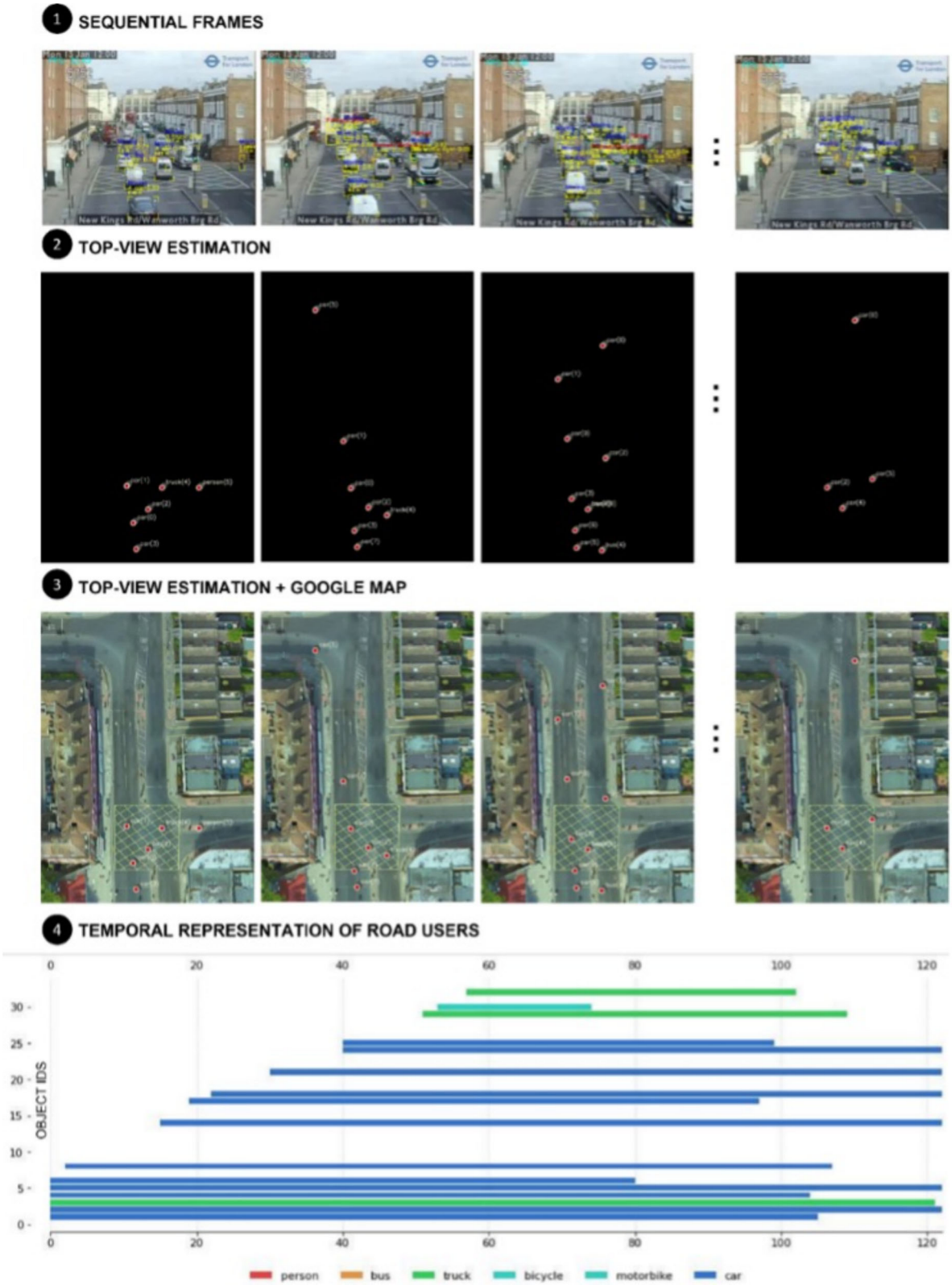

Fig. 6 Transforming road users in London CCTV camera to a Bird's eye estimation with Google maps to verify the geolocation of road users

https://doi.org/10.1007/s00521-025-11152-2

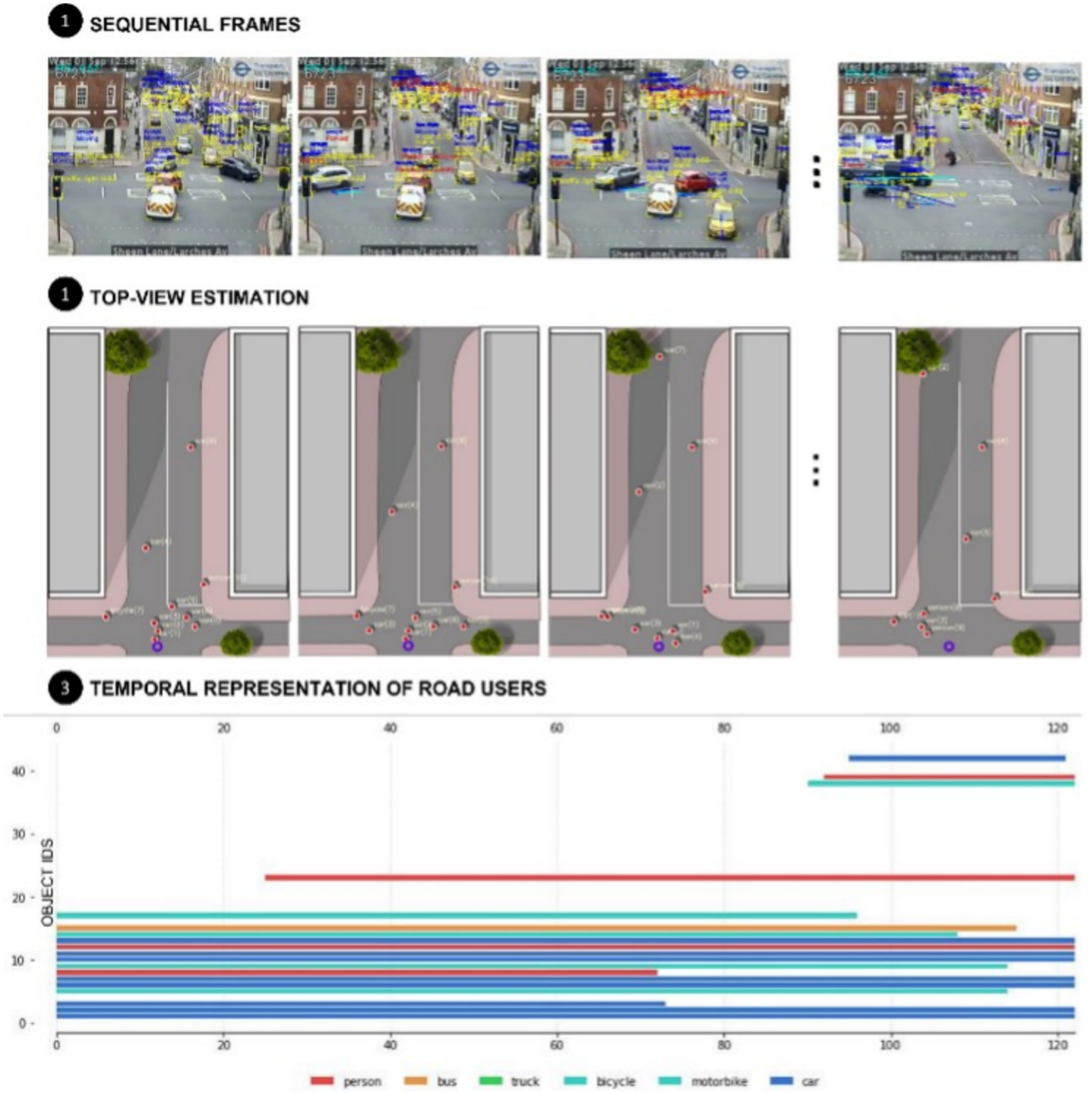

Fig. 7 Transforming road users in London CCTV camera to a Bird's eye estimation with a Semantic map to gamify a given video input

table (Table 3) highlights key features of each method, such as the need for camera calibration, object classes detectable, ability to generate 3D bounding boxes, tracking capabilities, and whether the method produces vector data. The "TopView" framework, proposed in this paper, stands out by not requiring camera calibration and supporting a wide range of object classes, including pedestrians, cars, and bicycles. It can generate 3D bounding boxes through geometric transformation and includes temporal tracking capabilities, producing vector data suitable for multiple applications. In contrast, methods like Geometry-based Homography and Learned Depth Estimation require camera calibration and are limited in their object class detection and tracking capabilities. The

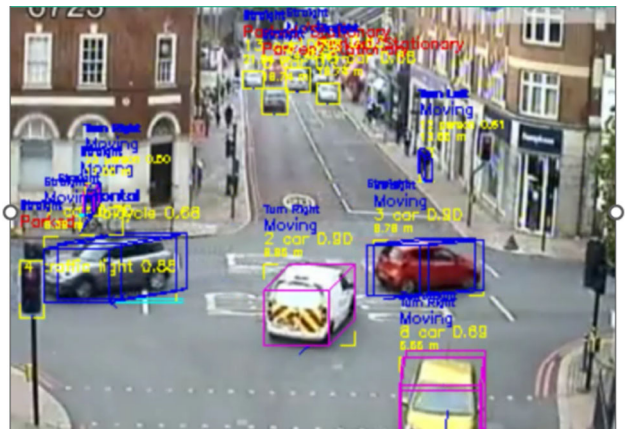

Fig. 8 An example of the estimated 3D bounding box based on the introduced Algorithm

Table 3 Qualitative comparison of BEV generation methods. Modality: 1 = Camera, 2 = LiDAR, 1 & 2 = Both. All objects includes Pedestrian, Cars, Bicycles

Method	Modality	Calib. Req.	Obj. classes	3D BBox	Tracking	Vector data
OFT-net (BEV) [5]	1	Yes	Vehicles only	No	No	No
Image-to-image [12]	1	No	None	No	No	No
PillarFlow [34]	2	Yes	Vehicles	Yes	Yes	No
BEVerse [37]	1	Yes	Vehicles	Yes	No	No
GitNet [44]	1	No	All	No	No	No
CenterFusion [60]	1 & 2	Yes	All	Yes	Yes	Yes
VoxelNet [61]	2	Yes	All	Yes	Yes	Yes
PointPillar [62]	2	Yes	All	Yes	Yes	Yes
CenterNet [63]	1	Yes	All	Yes	Yes	Yes
FCOS3D [64]	1	Yes	All	Yes	Yes	Yes
DETR3D [65]	1	Yes	All	Yes	Yes	Yes
PGD [66]	1	Yes	All	Yes	Yes	Yes
PETR-R50 [67]	1	Yes	All	Yes	Yes	Yes
PETR-R101 [67]	1	Yes	All	Yes	Yes	Yes
PETR-Tiny [67]	1	Yes	All	Yes	Yes	Yes
BEVDet-Tiny [68]	1	Yes	All	Yes	Yes	Yes
BEVDet-Base [68]	1	Yes	All	Yes	Yes	Yes
TopView (Proposed)	1	No	All	Yes	Yes	Yes

Multi-sensor fusion approach, although capable of producing 3D bounding boxes, relies on additional data sources such as LiDAR.The quantitative comparison table (Table 4) presents metrics such as average translation error (mATE), average scale error (mASE), average orientation error (mAOE), average velocity error (mAVE), and average attribute error (mAAE), as well as NuScenes detection score (NDS) and mean average precision (mAP) for the different methods. The "TopView" framework shows competitive performance across various metrics. Other methods, such as CenterFusion and VoxelNet, are also compared to highlight their effectiveness in 3D object detection on the nuScenes validation set. These tables collectively provide a comprehensive comparison of the methods, highlighting the strengths and limitations of each and positioning the "TopView" framework as a versatile and effective approach for BEV generation in uncalibrated street-level imagery.

4.2 Analysis, discussion and applications

In this research, we introduce a new framework to transform CCTV video feeds into a live representation of objects and events in Google Maps that can be used for multipurpose urban analytics. We provide two major contributions: (1) We provide extensive analysis of London traffic at scale, detailing the contributions of each

Table 4 Comparison with the state-of-the-art methods for 3D object detection on the nuScenes [69] validation set. Modality: 1 = Camera, 2 = LiDAR, 1 & 2 = Both

Method	Modality	mATE ↓	mASE↓	mAOE ↓	mAVE ↓	mAAE ↓	NDS ↑	mAP ↑
CenterFusion [60]	1 & 2	0.540	0.142	0.649	0.263	0.535	0.453	0.332
VoxelNet [61]	2	0.292	0.253	0.316	0.328	0.306	0.716	0.264
PointPillar [62]	2	0.295	0.803	0.268	0.511	0.374	0.303	0.860
CenterNet [63]	1	0.321	0.818	0.188	0.326	0.191	0.330	0.183
FCOS3D [64]	1	0.285	0.935	0.200	1.242	0.361	0.311	0.751
DETR3D [65]	1	0.303	0.794	0.216	1.152	0.356	0.343	0.710
PGD [66]	1	0.278	0.909	0.267	0.938	0.346	0.352	0.681
PETR-R50 [67]	1	0.225	0.859	0.314	0.862	0.271	0.376	0.605
PETR-R101 [67]	1	0.219	0.873	0.302	0.870	0.268	0.378	0.608
PETR-Tiny [67]	1	0.285	0.913	0.311	1.014	0.295	0.372	0.612
BEVDet-Tiny [68]	1	0.299	0.925	0.290	0.995	0.302	0.362	0.631
BEVDet-Base [68]	1	0.281	0.946	0.284	0.912	0.326	0.323	0.672
TopView (Proposed)	1	0.312	0.869	0.291	0.914	0.323	0.354	0.612

Fig. 9 A manual calibration tool for adjusting the estimated bird's eye view map from uncalibrated camera input such as internet images

traffic mode to congestion during their various actions (i.e. standing or moving). (2) We provide a new approach for visualising CCTV data spatially and temporally. In doing so, we transform video data into a summary of lower-dimensional anonymised data that can be stored and retrieved with minimal memory and computational requirements. Google Maps' realistic approach may enable further spatial analysis using standard spatial methods directly from scaled maps.

Generated trajectories and stream of paths:

The introduced approach is not only useful for BEV map representation but also for representing high-dimensional streams of events of multifaced features as token objects with unique IDs over a given time interval (i.e. length of a given video file). Accordingly, this provides a summary of highly dimensional data such as video streams into ordered, easy to retrieve and anonymous data representation that is suitable for multipurpose analysis. We showed multiple examples of how a video file can be transformed into a multi-dimensional vector

representation, with road users represented as tokens in the video file's time interval, where each point in time of a given token carries information such as 3D bounding box, stationary status, class name, and so on.

A simple manual calibration: Figure 9 shows a simple tool for calibrating a bird's eye map based on two values: (1) the z-value and (2) the x-value, without knowing the intrinsic and extrinsic parameters of a given camera. The z-value represents the spatial adjustment among the different objects in a given scene, and the x-value represents the shifting of road users in x-coordinates. While our framework generates an automatic bird's eye map, this tool provides additional control over the quality of the bird's eye map for manual calibration when necessary, particularly when linked to a Google Map.

Application 1: high precision geo-localisation of road users and objects in a given scene: A direct application for the TopView framework is to localise and generate GPS trajectories of road users through a given camera's location without knowing the GPS coordinates of individual road users. Here we show a few examples of localising road users through our framework in several CCTV cameras in London to show its versatility despite the complexity of road layouts. We showed a high precision in localising road users in the BEV map when compared to the road layouts of the camera feeds (see Fig. 1). Despite the complexity of the street layout, we showed some examples of the estimated BEV map of road users at a top of a Google Map. It shows how accurately the model localises objects within a given street layout, paving the way for many applications that rely on data related to GPS trajectories or understanding the interactions among road users in a given scene.

Application 2: analysis of spatial occupancy of road and open spaces layouts: Understanding who uses which space of a given street, sidewalk or open space could be useful for several studies related to urban analytics and street design. Through the introduced framework, several studies can translate video streams of a given space to an occupancy map, highlighting the busiest spaces used by several road users, and spaces that are more likely to be deserted by road users. By doing so, current road layouts and open spaces can be evaluated and re-designed to meet the needs of their users based on their occupancy. In the future, pedestrian activities and actions can be analysed alongside their spatial occupancy to give empirical evidence of how spaces are used post-occupation.

Application 3: exposure based on violating social distancing at the city scale: Another application of the introduced framework is to utilise the BEV map to analyse distances and safety-based thresholds among different

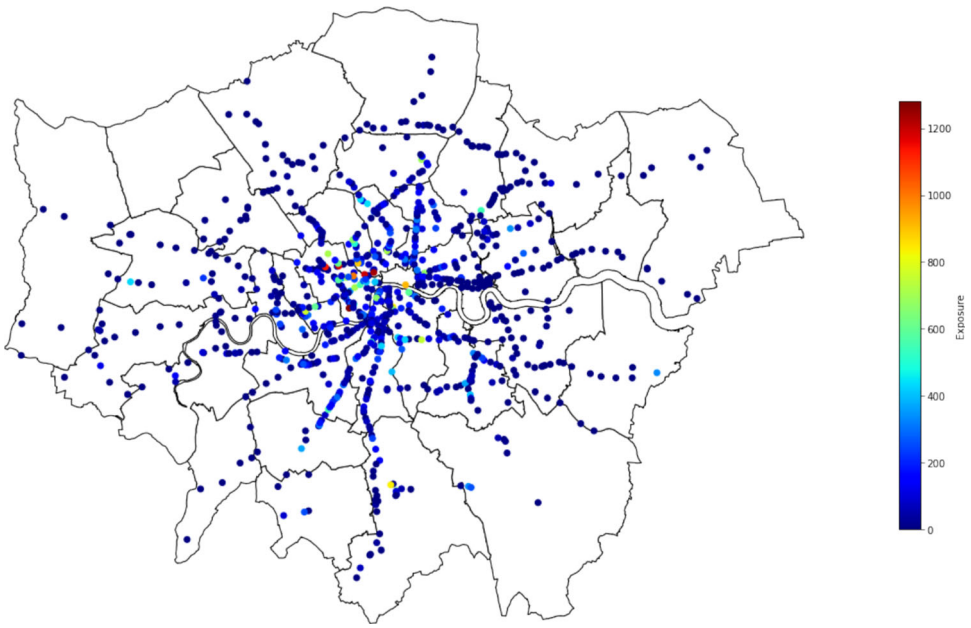

Fig. 10 A map showing the level of exposure of pedestrians based on violating social distancing for a given hour (20210813 – 0800_0900) as an application of the estimated BEV map

road users to understand for instance, exposure, collisions, or even near misses. Here we utilised the BEV map to analyse distances between pedestrians with a two-metre threshold to show the level of exposure across London from 857 cameras at a given hour of a given day. Figure 10 shows the count of human contacts that violate social distancing. Across all of London, the map shows that the majority of violations occurred in the city centre of London. This application shows the framework's versatility in shifting from observing several sites at a micro-level to a city scale.

Limitations and future work: Several future investigations can be conducted to advance the introduced framework. First, the estimated 3D bounding boxes are based on moving objects, whereas it is limited when objects are stationary in a given scene. Further study can be done to rely on the relationship between the vanishing point of a given scene and the pose of a given object for stationary objects, instead of its trajectory line. Second, developing an automated method to transform scene components such as road layout [38, 39] and lane lines [70, 71] into a semantic vector representation would appear to be the next logical step in improving the introduced framework.

5 Conclusion

Scene awareness for a multiview representation represents a crucial domain in vision and machine learning research. In this paper, we presented a hybrid method for estimating a vector representation of objects in a BEV without relying on the camera matrix information. This offers a similar approach to human navigation in spaces by understanding the relationship between objects rather than the exact depth of individual ones. Nevertheless, based on simple calibration, we also presented a geo-tagging of objects in a Google map with a very high spatial resolution which is useful for many applications related to urban analytics and autonomous navigation. Furthermore, this approach also provides a 3D bounding representation based solely on the geometric transformation of the 2D bounding box and trajectory lines, in the case of sequential frames. This paper presents two opportunities for future research such as (1) 3D mesh representations of objects in complex scenes by learning from their multi-dimensional vector representation of point data, and (2) gamification of urban scene data and anonymising video stream data.

Data availability All raw data used in this study can be accessed online. The sources and methods for obtaining this data have been explained in the methodology section of this article.

Declarations

Conflict of interest The author declares that there is no Conflict of interest regarding the publication of this article.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s00521-025-11152-2.

References

1. Ma Y, et al (2022) Vision-Centric BEV Perception: A Survey. Accessed: Dec. 10, 2022. arXiv:abs/2208.02797
2. Li H, et al (2022) Delving into the Devils of Bird–s-eye-view Perception: A Review, Evaluation and Recipe. Accessed: Dec. 10, 2022. arXiv:abs/2209.05324
3. Houston J, et al (2020) One Thousand and One Hours: Self-driving Motion Prediction Dataset. Accessed: Dec. 10, 2022. arXiv:abs/2006.14480
4. Ibrahim MR, Haworth J, Cheng T (2020) Understanding cities with machine eyes: a review of deep computer vision in urban analytics. Cities 96:102481. https://doi.org/10.1016/j.cities.2019.102481
5. Roddick T, Kendall A, Cipolla R (2018) Orthographic Feature Transform for Monocular 3D Object Detection. Accessed: Dec. 10, 2022. arXiv:abs/1811.08188
6. Xingfang Y, Yumei H, Feng G (2010) A simple camera calibration method based on sub-pixel corner extraction of the chessboard image. In: 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems, Xiamen, China, pp. 688–692. https://doi.org/10.1109/ICICISYS.2010.5658280
7. Schoepflin TN, Dailey DJ (2003) Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation. IEEE Trans Intell Transp Syst 4(2):90–98. https://doi.org/10.1109/TITS.2003.821213
8. Mardiati R, Mulyana E, Maryono I, Usman K, Priatna T (2019) The derivation of matrix transformation from pixel coordinates to real-world coordinates for vehicle trajectory tracking. In: 2019 IEEE 5th International Conference on Wireless and Telematics (ICWT), Yogyakarta, Indonesia, pp. 1–5. https://doi.org/10.1109/ICWT47785.2019.8978254
9. Venkatesh M, Vijayakumar P (2012) A simple bird's eye view transformation technique. Int J Sci Eng Res 3(5):4
10. Lin C-C, Wang M-S (2012) A vision based top-view transformation model for a vehicle parking assistant. Sensors 12(4):4431–4446. https://doi.org/10.3390/s120404431
11. Kholopov IS (2017) Birds eye view transformation technique in photogrammetric problem of object size measuring at low-altitude photography. In: Proceedings of the International Conference Actual Issues of Mechanical Engineering 2017 (AIME 2017), Tomsk, Russia. https://doi.org/10.2991/aime-17.2017.52
12. Regmi K, Borji A (2018) Cross-View Image Synthesis using Conditional GANs. Accessed: Dec. 10, 2022. arXiv:abs/1803.03396
13. Mani K, Daga S, Garg S, Shankar NS, Jatavallabhula KM, Krishna KM (2020) MonoLayout: Amodal scene layout from a single image. Accessed: Dec. 10, 2022. arXiv:abs/2002.08394
14. Zou Z, Chen K, Shi Z, Guo Y, Ye J (2023) Object detection in 20 years: a survey. Proc IEEE 111(3):257–276
15. Liu L, Ouyang W, Wang X, Fieguth P, Chen J, Liu X, Pietikäinen M (2020) Deep learning for generic object detection: a survey. Int J Comput Vision 128:261–318
16. Jiao L, Zhang F, Liu F, Yang S, Li L, Feng Z, Qu R (2019) A survey of deep learning-based object detection. IEEE Access 7:128837–128868
17. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 886–893. Ieee
18. Lowe G (2004) Sift-the scale invariant feature transform. Int J 2(91–110):2
19. Hearst MA, Dumais ST, Osuna E, Platt J, Scholkopf B (1998) Support vector machines. IEEE Intell Syst Appl 13(4):18–28
20. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448
21. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1440–1448
22. Redmon J, Farhadi A (2017) Yolo9000: Better, faster, stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, pp. 6517–6525. https://doi.org/10.1109/CVPR.2017.690
23. Ultralytics: YOLOv5. https://github.com/ultralytics/yolov5 (2021)
24. Redmon J, Farhadi A (2018) YOLOv3: An Incremental Improvement
25. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu C-Y, Berg AC (2016) Ssd: Single shot multibox detector. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, pp. 21–37. Springer
26. Zhang Z (1999) Flexible camera calibration by viewing a plane from unknown orientations. In: Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, pp. 666–6731. https://doi.org/10.1109/ICCV.1999.791289
27. Nassar A, Lefevre S, Wegner JD (2019) Simultaneous multi-view instance detection with learned geometric soft-constraints. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea (South), pp. 6558–6567. https://doi.org/10.1109/ICCV.2019.00666
28. Escalera AD, Armingol JM (2010) Automatic chessboard detection for intrinsic and extrinsic camera parameter calibration. Sensors 10(3):2027–2044. https://doi.org/10.3390/s100302027
29. Frana JA, Stemmer MR, M Frana MB, Piai JC (2012) A new robust algorithmic for multi-camera calibration with a 1d object under general motions without prior knowledge of any camera intrinsic parameter. Pattern Recognit. 45(10):3636–3647. https://doi.org/10.1016/j.patcog.2012.04.006

30. Ernst S, Stiller C, Goldbeck J, Roessig C (1999) Camera calibration for lane and obstacle detection. In: Proceedings 199 IEEE/IEEJ/JSAI International Conference on Intelligent Transportation Systems (Cat. No.99TH8383), Tokyo, Japan, pp. 356–361. https://doi.org/10.1109/ITSC.1999.821081
31. Kim Y, Kum D (2019) Deep learning based vehicle position and orientation estimation via inverse perspective mapping image. In: 2019 IEEE Intelligent Vehicles Symposium (IV), Paris, France, pp. 317–323. https://doi.org/10.1109/IVS.2019.8814050
32. Abbas A, Zisserman A (2020) A Geometric Approach to Obtain a Birds Eye View from an Image. Accessed: Jan. 16, 2023. arXiv:abs/1905.02231
33. Hendy N, et al (2020) FISHING Net: Future Inference of Semantic Heatmaps In Grids. Accessed: Dec. 10, 2022. arXiv:abs/2006.09917
34. Lee K-H, et al (2020) PillarFlow: End-to-end Birds-eye-view Flow Estimation for Autonomous Driving. Accessed: Dec. 10, 2022. arXiv:abs/2008.01179
35. Liu Y, et al (2022) PETRv2: A Unified Framework for 3D Perception from Multi-Camera Images. Accessed: Jan. 16, 2023. arXiv:abs/2206.01256
36. Li Y, et al (2022) BEVDepth: Acquisition of Reliable Depth for Multi-view 3D Object Detection. Accessed: Jan. 16, 2023. arXiv:abs/2206.10092
37. Zhang Y, et al (2022) BEVerse: Unified Perception and Prediction in Birds-Eye-View for Vision-Centric Autonomous Driving. Accessed: Jan. 16, 2023. arXiv:abs/2205.09743
38. Lu J, Zhou Z, Zhu X, Xu H, Zhang L (2022) Learning Ego 3D Representation as Ray Tracing. Accessed: Jan. 16, 2023. arXiv:abs/2206.04042
39. Pan B, Sun J, Leung HYT, Andonian A, Zhou B (2020) Cross-view semantic segmentation for sensing surroundings. IEEE Robot. Autom. Lett. 5(3):4867–4873. https://doi.org/10.1109/LRA.2020.3004325
40. Peng L, Chen Z, Fu Z, Liang P, Cheng E (2022) BEVSegFormer: Birds Eye View Semantic Segmentation From Arbitrary Camera Rigs. Accessed: Dec. 10, 2022. arXiv:abs/2203.04050
41. Bartoccioni F, Zablocki , Bursuc A, Prez P, Cord M, Alahari K (2022) LaRa: Latents and Rays for Multi-Camera Birds-Eye-View Semantic Segmentation. Accessed: Jan. 16, 2023. arXiv:abs/2206.13294
42. Can YB, Liniger A, Paudel DP, Gool LV (2021) Structured birds-eye-view traffic scene understanding from onboard images. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, pp. 15641–15650. https://doi.org/10.1109/ICCV48922.2021.01537
43. Zou J et al (2022) HFT: Lifting Perspective Representations via Hybrid Feature Transformation. Accessed: Jan. 16, 2023. arXiv:abs/2204.05068
44. Gong S et al (2022) GitNet: Geometric Prior-based Transformation for Birds-Eye-View Segmentation. Accessed: Jan. 16, 2023. arXiv:abs/2204.07733
45. Liu S, Zhou Y, Zhao Y (2021) Vapid: A rapid vanishing point detector via learned optimizers. In: ICCV
46. Zhou Z, Farhat F, Wang JZ (2017) Detecting Dominant Vanishing Points in Natural Scenes with Application to Composition-Sensitive Image Retrieval. Accessed: Jan. 16, 2023. arXiv:abs/1608.04267
47. Lin T-Y, et al (2014) Microsoft coco: Common objects in context. In: Computer Vision ECCV 2014. Lecture Notes in Computer Science, vol. 8693, pp. 740–755. https://doi.org/10.1007/978-3-319-10602-1_48
48. nwojke: DeepSort. https://github.com/nwojke/deepsort.git (2019)
49. Bewley A, Ge Z, Ott L, Ramos F, Upcroft B (2016) Simple online and realtime tracking. In: 2016 IEEE Int. Conf. Image Process. ICIP, pp. 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003
50. Brazil G, Liu X (2019) M3D-RPN: Monocular 3D Region Proposal Network for Object Detection. Accessed: Jan. 16, 2023. arXiv:abs/1907.06038
51. Hung W-C, Kretzschmar H, Casser V, Hwang J-J, Anguelov D (2022) LET-3D-AP: Longitudinal Error Tolerant 3D Average Precision for Camera-Only 3D Detection. Accessed: Jan. 16, 2023. arXiv:abs/2206.07705
52. Rukhovich D, Vorontsova A, Konushin A (2021) ImVoxelNet: Image to Voxels Projection for Monocular and Multi-View General-Purpose 3D Object Detection. Accessed: Jan. 16, 2023. arXiv:abs/2106.01178
53. Wang T, Zhu X, Pang J, Lin D (2021) Accessed: Jan. 16, 2023. arXiv:abs/2104.10956
54. Shi S, Wang X, Li H (2019) PointRCNN: 3D Object Proposal Generation and Detection from Point Cloud. Accessed: Jan. 16, 2023. arXiv:abs/1812.04244
55. Howard AG, et al (2017) MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. Accessed: Nov. 17, 2022. arXiv:abs/1704.04861
56. Kingma DP, Ba J (2015) Adam: A Method for Stochastic Optimization. arXiv:1412.6980 [cs]. Accessed: Apr. 23, 2019. https://hdl.handle.net/11245/1.505367
57. Chang C-K, Zhao J, Itti L (2018) Deepvp: Deep learning for vanishing point detection on 1 million street view images. In: 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, pp. 4496–4503. https://doi.org/10.1109/ICRA.2018.8460499
58. Zhou Y, Qi H, Huang J, Ma Y (2021) NeurVPS: Neural Vanishing Point Scanning via Conic Convolution. Accessed: Jan. 19, 2023. arXiv:abs/1910.06316
59. TfL: TfL London cameras. https://www.tfljamcams.net/ (2021)

60. Nabati R, Qi H (2021) Centerfusion: Center-based radar and camera fusion for 3d object detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)
61. Maturana D, Scherer S (2015) Voxelnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
62. Lang AH, Vora S, Caesar H, Zhou L, Yang J, Beijbom O (2019) Pointpillars: Fast encoders for object detection from point clouds. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
63. Zhou X, Wang D, Krahenbuhl P (2019) Objects as points. In: IEEE Int. Conf. Comput. Vis
64. Wang T, Zhu X, Pang J, Lin D (2021) Fcos3d: Fully convolutional one-stage monocular 3d object detection. In: Proceedings of IEEE International Conference on Computer Vision (ICCV)
65. Wang Y, Guizilini V, Zhang T, Zhao H, Solomon JM (2021) Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In: Proceedings of the Conference on Robot Learning (CoRL)
66. Wang T, Zhu X, Pang J, Lin D (2022) Pgd: Probabilistic and geometric depth: Detecting objects in perspective. In: Proceedings of the Conference on Robot Learning (CoRL)
67. Liu Y, Chen W, Li Q, Jia L, Liu Z, Li X, Guan T (2022) Petr: Position embedding transformation for multi-view 3d object detection. ArXiv preprint arXiv:2203.05625
68. Huang T, Guo J, Wang H, Liao R, Xie E, Han J, Qiao Y, Liu W (2022) Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. ArXiv preprint arXiv:2203.05650
69. Caesar H, Bankiti V, Lang AH, Vora S, Liong VE, Xu Q, Krishnan A, Pan Y, Baldan G, Beijbom O (2020) nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)
70. Garnett N, Cohen R, Pe'er T, Lahav R, Levi D (2019) 3D-LaneNet: End-to-End 3D Multiple Lane Detection. Accessed: Jan. 16, 2023. arXiv:abs/1811.10203
71. Guo Y, et al (2020) Gen-lanenet: A generalized and scalable approach for 3d lane detection. In: Lecture Notes in Computer Science. Computer Vision ECCV 2020, vol. 12366, pp. 666–681. https://doi.org/10.1007/978-3-030-58589-1_40

Authors and Affiliations

Mohamed R. Ibrahim[1]

✉ Mohamed R. Ibrahim
m.ibrahim1@leeds.ac.uk

[1] Institute for Spatial Data Science, Department of Environment, University of Leeds, Leeds, UK