# Traffic Signs Mapping in Cambodia with Deep Learning and GNSS

Sophal Ratitya[1*] and Mitsuharu Tokunaga[2]

[1]Graduate Student, Department of Civil and Environmental Engineering,

Kanazawa Institute of Technology, Japan

[2]Professor, Department of Civil and Environmental Engineering,

Kanazawa Institute of Technology, Japan

*titan.titya@gmail.com

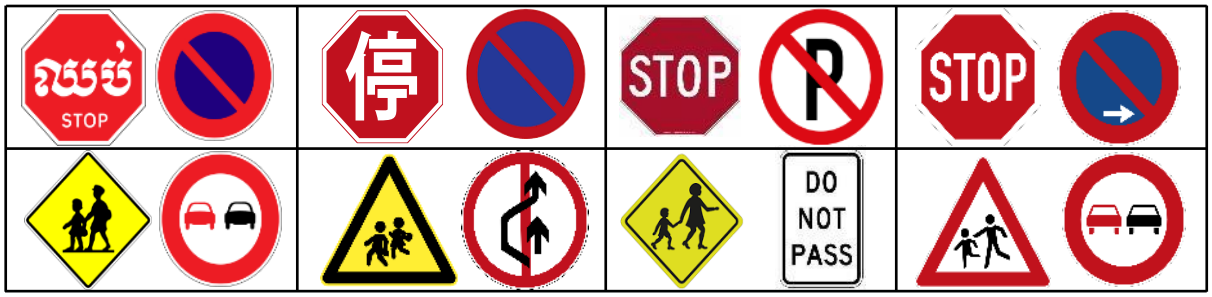
***Abstract*** *Traffic signs are necessary road features that help ensure safety. However, mapping these objects can be a time-consuming and tedious task. This study explores the use of deep learning and low-cost devices to improve the efficiency of traffic signs mapping in Cambodia. A traffic signs dataset was created from selected frames captured by a GoPro 12 action camera mounted on a car. These images were then labelled using image annotation software. Next, the traffic signs detection and classification models were trained using the pretrained YOLOv8 model in the Ultralytics framework. The Ultralytics tracking was used to maintain a unique ID of each sign in the video, allowing us to capture pair frames as the object crosses a designated line. To determine the frame coordinate, the video starting time and the Global Navigation Satellite System (GNSS) time are synchronised. The next step is to calculate the detected sign in real-world coordinates. This involves finding the camera to GNSS difference based on camera height and field of view. The Haversine equation is used to find the distance between the camera in the pair frame. The objects' bounding boxes in both frames are obtained from the detection model. Then, the camera-to-object distance is calculated using a pair of frames for triangulation, while the angle of the objects to the camera's line of sight is also measured. Finally, coordinates are calculated based on distance and angle. The experiments indicate that the approach accurately detects and identifies the coordinates of traffic signs with a mean error of 4.633 m. Consequently, the mapping process becomes economical, straightforward, and time-effective. Furthermore, an inventory can be easily built and updated frequently, facilitating efficient road asset management. While the system's accuracy is mainly on the GNSS signal, it can be further improved using a high-precision GNSS integrated with other sensors or a real-time kinematic (RTK) positioning application. This research contributes to remote sensing by reducing the time needed for object mapping and by effectively integrating deep learning, computer vision, and GNSS for enhanced data acquisition.*

*Keywords: Deep Learning, Object Geolocation, Object Mapping, Road Inventory, Traffic Signs Dataset*

## Introduction

Traffic signs are vital for all road users, as they provide information on the situation on the road ahead. Much research in recent years has focused on traffic signs detection and recognition; however, most models target autonomous driving. By integrating GNSS and other sensors, studies can locate the sign locations. These techniques have also attracted attention in the road inventory field. Traffic signs mapping is a process to as-built or identify every location of road sign in real-world coordinates using Differential Global Positioning Systems (DGPS),

hand-held Global Positioning System (GPS), or other GNSS device, then put those points in a geographic information system (GIS). Some research has relied on advanced hardware or street view imagery (SVI), such as stereo vision, light detection and ranging (LiDAR), and platforms like Google Street View for object mapping. These methods are expensive and often technically difficult to perform. This paper focuses on using a low-cost, consumer-grade portable device and an efficient method to improve the mapping process.




Cambodia (*MPWT*) China US Germany

Figure 1: Pattern differences of cross-regional samples. Four representative traffic signs (Stop, No Parking, Children Crossing, and No Overtaking).

**Literature Review**

There are several potential gaps in previous research, including limitations related to datasets, the use of expensive technologies, reliance on external platforms like Google or Mapillary, and the inaccurate and slow processes involved in using deep learning (DL) for estimating object location.

Datasets play a crucial role in training DL models. Several remarkable datasets of traffic signs have been established, such as the Germany Traffic Sign Detection Benchmark (GTSDB) by Houben et al. (2013), the Tsinghua-Tencent 100K for traffic-sign classes by Zhu et al. (2016), and the Automotive Repository of Traffic Signs (ARTS) for traffic signs recognition in the United States (US) by Almutairy et al. (2021). However, these datasets primarily focus on safe-driving cars, and there is a significant lack of a comprehensive traffic signs inventory. Moreover, the signs can vary from country to country; constructing a specific dataset is necessary (Figure 1). To address this, Zhao et al. (2025) proposed a method that can recognise worldwide traffic signs. The model can generalise new knowledge on traffic signs, but it worked only for 41 class signs, which is still not suitable for detecting signs for inventory purposes. Chen & Ren (2023) also considered sign statuses and built a model that can differentially key features for a damaged sign. Even so, it could not identify the type of damage.

Some studies relied on high-cost technologies. For instance, Balado et al. (2020) presented a mapping of traffic signs based on images and point clouds from MMS (Mobile Mapping System). The signs are detected by DL from images obtained by a 360° camera and positioned with a point cloud obtained from a LiDAR sensor. The model accuracy is 89.7% with a location error below 0.5 m. On the other hand, Wang et al. (2010) proposed an automated road sign inventory system based on stereo vision and tracking. This system worked in real-time, and all sensors synchronised with a high-resolution clock. His method is not scalable and compact because it is a heavy computation on an onboard computer and runs in the Digital Highway Data Vehicle (DHDV). The automated system provided a location error of 5 to 18 m for the stereo camera.

Other research is emerging on the SVI and DL for object localisation. Campbell et al. (2019) explored the DL for detecting traffic signs on Google SVI to assist in traffic assets monitoring and maintenance. However, this method only identified the STOP and Give Way signs at the intersection. Moreover, the photogrammetry approach was applied to calculate the object location, which required the use of a known-sized reference object. This method achieved an accuracy of 95.61% and a mean location error of 3.677 m. Furthermore, Krylov et al. (2018) proposed a monocular depth estimation and triangulation to map traffic lights and telegraph poles. The depth estimation consumes significant computational resources, and the experiments reported a 93.36% recall rate and around 2 m positioning. In addition, Ning et al. (2025) developed SVI to conduct a mapping framework for built environments, such as measuring the width of a road, 3D localisation of a STOP sign, and tree diameter measurement. In 3D localisation, the pipeline relied on the known target object size in a single image. Then the Tachometric survey principle is used to find the object-to-camera distance with the field of view (FOV) angle. These methods still rely on the Google platform, which incurs a cost when exceeding the free tier, and are not up-to-date SVI in developing countries. The result achieved a mean error of less than 2 m. On the other hand, Mapillary is an open-source mapping platform that incorporates global traffic signs recognition for over 1,500 categories. This allows for detailed asset inventories and navigation mapping. However, it has some limitations: it may detect duplicate signs, cannot recognise all traffic signs in Cambodia, takes a longer time to upload and localise data, and its positions can be far from the base point. Furthermore, Wilson et al. (2022) proposed a two-stage DL model to track and geo-localisation objects from street images. A GPS-RetinaNet model was designed to estimate the offset position of each sign to the camera. The ARTS2 dataset is a geo-dataset containing sign classes and geospatial

locations. Even though it is open source, it works only for American signs. Thus, building that geo dataset is costly and currently unfeasible in our context. Their tracker is designed to work only for a low frame rate (1 fps); thus, it does not work for a simple video recorder. The mean error of predicted distances is within 5 m. Moreover, Chaabane et al. (2021) also proposed end-to-end deep learning for traffic light geo-localisation in video. This method still required a geo dataset acquired from nuScene that used 6 cameras and LiDAR, and camera metadata for calibration. Their method showed the median depth error of about 1.5 m with tracking accuracy of 85.52%.

In summary, previous research used high-cost hardware, depended on other platforms, required high computation in the localisation DL process, and had inadequate traffic signs datasets for content. Although there are several open-source geo-datasets like nuScence (Caesar et al., 2020), KITTI (Geiger et al., 2013), they are only valid for other regions, not Cambodia, and are focused on autonomous vehicles.

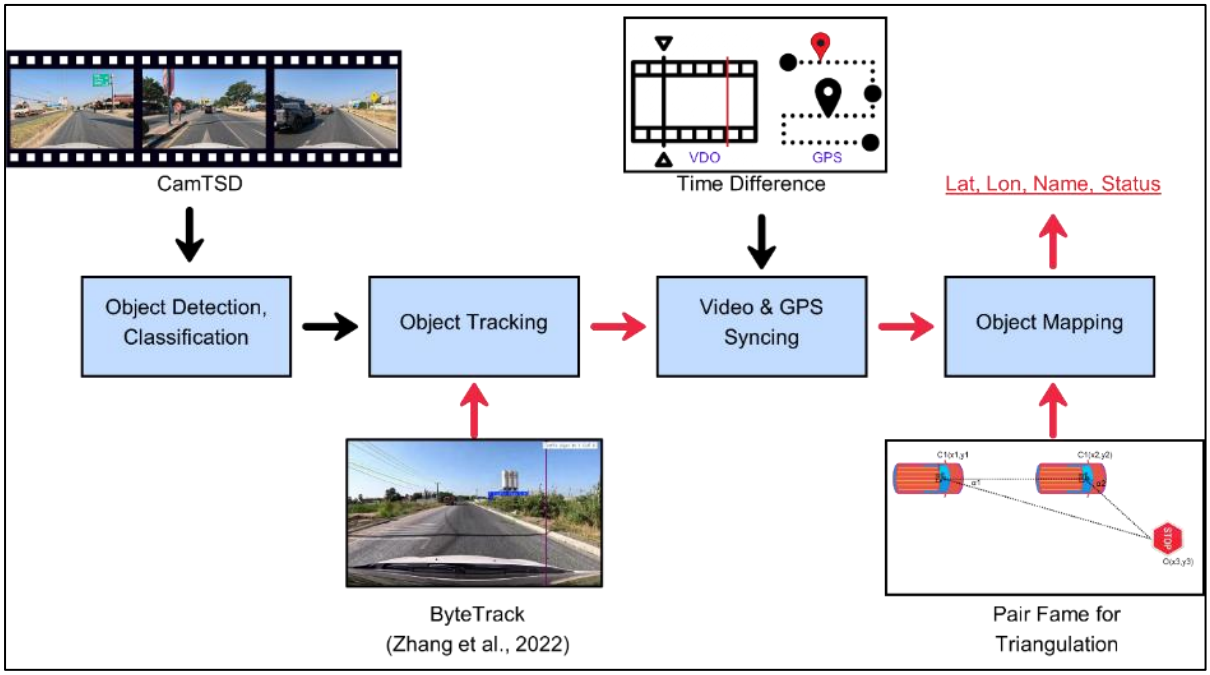



Figure 2: Method Pipeline

**Methodology**

To achieve the goal of traffic signs mapping, DL models were used along with Triangulation. There are several ways to get object location, such as using LiDAR point cloud, structure from motion, and triangulation using camera images, GNSS, and camera metadata (Kunsági-Máté et al., 2025). However, in triangulation, some authors have been using it with certain limitations, which depend on a known size. In this research, we utilise triangulation between a

pair of frames to measure camera-object-distance and DL to detect and recognise in Right-of-Way (ROW) video. Our automated mapping process consists of four steps as follows (Figure 2):

1. The traffic signs detection and recognition models are used to obtain the bounding box position in the image and perform sign name and status recognition.
2. The object tracking model is used to get a unique ID of the sign to avoid duplication.
3. Video and GPS are aligned based on the timestamp for locating the camera or frame coordinate.
4. Object mapping is performed to determine the object coordinate based on a pair of frames to form triangulation.

a. **Equipment use:**

While other methods rely on high-cost technology, our method uses only one action camera and a GNSS device, which are compatible with vehicles and portable. Everyone can easily set up in any car. The GoPro Hero 12 Black offers high-quality resolution up to 4K and Hyper Smooth 6.0 Vibration correction, and U-blox ANN-MB-00 listens to L1 and L2 bands in a multi-constellation satellite with QZNEO receiver developed by Core Group Japan (Figure 3). This GNSS device can provide a stable sub-meter positioning in single frequency mode and sub-centimetre positioning in Real-time kinematic positioning (RTK) mode (Core Group).


Figure 3: (Left) GoPro12 and GNSS mounted on the car. (Right) U-Blox ANN-MB-00 and QZNEO

b. **Traffic Signs Detection and Recognition:**

Traffic signs detection means the sign is detected with a bounding box in the image frame; this implies the sign location is identified in the 2D image plane. Traffic signs recognition means

the sign's content is identified after being detected in cropped images. There are several methods to achieve this by DL or image processing. In this paper, we use You Only Look Once version 8 (YOLOv8) (Jocher et al., 2023) for object detection and classification, since it is a single-stage detection model that works quickly while achieving high accuracy. Among the various model sizes available, we selected the YOLO small model because it is lightweight and allows for fast inference on standard computers.


Figure 4: Image Annotation with VIA Tool

First, the Cambodia Traffic Signs Dataset (CamTSD) has been developed to account for regional variation in traffic signs (see Figure 1). The dataset includes selected frames captured by a GoPro 12 Black action camera mounted on the roof of a car. The video frames are recorded in standard video mode with a 16:9 aspect ratio, using a linear digital lens, HyperSmooth stabilisation, and in 4K resolution. The CamTSD consisted of 2,700 images and 10,200 annotations. The images are in RGB (Red, Green, Blue) format with a 4K resolution (3840x2160). The annotations were done using the VIA tool (Dutta & Andrew 2019) and included 1 type of road object (traffic sign), 7 classes, 48 traffic signs names, and 5 statuses for damage signs (see Figure 4). The bounding box was drawn carefully around the traffic signs and labelled in every image. After the annotation process was completed, a JSON annotation file was exported. Then the JSON file was converted to the YOLO label structure format for an object detection and classification dataset using a Python script. In classification datasets, only 26 signs were selected to avoid class imbalance. This allows our model to detect 26 recognisable traffic signs, while others are treated as "Not Recognised Sign."

Then, the datasets were fed into the model training process using the Ultralytics framework with a pretrained YOLOv8n model (Jocher et al., 2023). Ultralytics allows the program to run smoothly in training and running the model with just a few lines of code, making it convenient and user-friendly for non-technical users. Additionally, it is open source, which is beneficial for researchers. The trained models are for traffic signs detection, traffic signs recognition, and sign status classification. The detection model was trained on 640×640 images for 80 epochs. The two classification models were trained on 64×64 images for 80 epochs. Table 2 shows the model's metrics.

Table 1: Traffic signs detection metrics

| Model | Precision | Recall | mAP@50 | mAP@50-95 |
|---|---|---|---|---|
| Traffic Signs Detection | 0.96 | 0.95 | 0.97 | 0.87 |

**c. Object Tracking:**


Figure 5: Tracking Inference with Predefined Region

Second, the object tracking was performed based on the bounding boxes detected by the object detection model to predict the next location of the objects and maintain consistent IDs to prevent duplicate sign captures. There are two available trackers in Ultralytics BOT-SORT and ByteTrack. In this paper, the ByteTrack (Zhang et al., 2022) was used because it is fast, lightweight, and can handle partial occlusions by using both high- and low-confidence detections.

ByteTrack is a high-performance, real-time multi-object tracker that improves upon Simple Online and Realtime Tracking (SORT) by utilising low-confidence detections to reduce missed tracks. The process involves three main steps: First, a detector like YOLOv8 identifies objects

in each frame and assigns a confidence score. Second, we match both high- and low-score detections using Intersection over Union (IoU) and Kalman Filter prediction. Third, the Kalman Filter predicts the object motion in the next frame.

Before running tracking inference, the predefined lines in the frame were set (Figure 5). Whenever the tracked sign crossed the lines, a pair of frames was captured corresponding to frame numbers in the video. These pairs of frame data included the bounding box of the object, frame number, and track ID.

**d. Video and GPS Syncing:**

Third, to obtain the location of each camera frame, we need to synchronise the video data with the GPS data (GPS and GNSS terms are interchangeable here). Although both devices share a common timestamp, there is a slight deviation between the GPS time and the camera time. Accurate synchronisation can be achieved by photographing the time displayed on the GPS unit during data collection (Exiftool). Because the GPS device does not provide the time value directly, we developed an alternative method to estimate the time deviation. To achieve this, we will identify a certain landmark in the video and on the GPS route in the map and then compare their timestamps. Thus, the video and GPS data are interconnected. This step ensured seamless integration for triangulation.

**e. Object Mapping:**

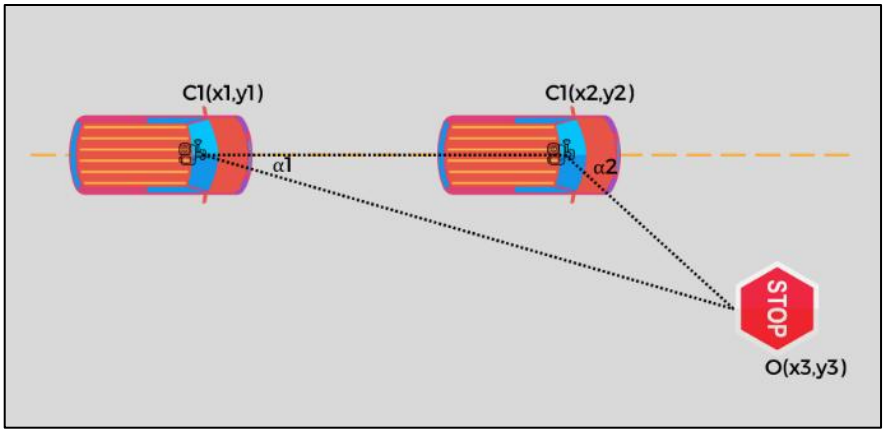



Figure 6: Triangulation based on a Pair Frame

Finally, object mapping is processed, creating two frames and identifying the object as a triangle. The position of the object in the image is converted into real-world coordinates (Figure 6). Since directly calculating the GPS coordinates of signs is challenging, the object-to-camera distance is first calculated, and the angle of the sign is measured. Then, by adding the offset distance and the angle, the sign's actual GPS coordinates could be found. There are many ways

to get camera-object-distance, as mentioned in the literature review, but they rely on other sensors and sources. In our paper, we utilise a simple method that relies on triangulation between a pair of frames. This work can be done when the distance between the two cameras is known.

First, the difference between the GPS locations and the camera ($D_1$) was defined. There was a slight distance of about 10 cm from the camera to the GPS locations, as those devices could not be mounted on top of each other. However, this slight distance was ignored. The other consideration was different in the different camera vertical field of view ($VFOV$). If the $VFOV$ is 90°, the correct point of the landmark (rumble strip) could be exactly set, so a different distance was needed to calculate using Equation (1), where $H_C$ is the camera height.

$$D_1 = H_C \cdot \tan\left(90 - \frac{VFOV}{2}\right) \quad (1)$$

Second, the distance between the camera ($D_2$) in frame 1 ($C_1$) and frame 2 ($C_2$) was calculated based on the Haversine formula in Equation (2-4), which provides an accurate approximation at close distances. It is denoted as follows, where $c$ is the relative distance, φ is latitude, λ is longitude, and $R$ is the mean Earth's radius to 6,371 km. This distance is also called the camera movement distance.

$$a = sin^2(\frac{\Delta\varphi}{2}) + cos(\varphi_1) \cdot cos(\varphi_2) \cdot sin^2(\frac{\Delta\lambda}{2}) \quad (2)$$

$$c = 2 \cdot atan2(\sqrt{a}, \sqrt{1-a}) \quad (3)$$

$$D_2 = R \cdot c \quad (4)$$

Third, camera-object-distance ($D_3$) was calculated based on the bounding box parameters from the object detection model. Let $\alpha_1$ and $\alpha_2$ be the angles formed between the line of cameras and the line of the target object $O$. By using trigonometry, the distance D3 was calculated using the tangent function. The total distance ($D_4$) was founded by addition $D_1$ to $D_3$.

Finally, the real-world object coordinates were calculated based on the total distance ($D_4$) and the azimuth angle ($\theta$) of the object in frame 1 to the camera's line using the Haversine and trigonometry. The ($C_1 = \varphi_1,\ \lambda_2$) and ($C_2 = \varphi_2,\ \lambda_2$) are camera coordinates in frames 1 and 2. The angle of the detected object was measured by the centre point x of the bounding box to the camera's line of sight, which is related to the focal length in pixels ($f\_px$) and image width using Equation (5). The azimuth angle of the camera line was calculated using Equation (6) and then added to the object angle ($\theta_O$). By giving the distance (D) and azimuth angle of the object ($\theta$), the coordinates were calculated using Equations (9) and (10). The process showed

the viability of using a mono camera to identify the object location without using high-cost equipment and heavy computational processing.

$$\theta_O = \text{atan}\left(\frac{object_x - 0.5 \cdot image\ width}{f\_px}\right) \quad (5)$$

$$D = \frac{D_4}{R} \quad (6)$$

$$\theta_L = \text{atan}\left(\frac{\sin(\varphi_2 - \lambda_1) \cdot \cos(\varphi_2)}{\cos(\varphi_1) \cdot \sin(\varphi_2) - \sin(\varphi_1) \cdot \cos(\varphi_2) \cdot \cos(\lambda_2 - \lambda_1)}\right) \quad (7)$$

$$\theta = \theta_L + \theta_O \quad (8)$$

$$\varphi_O = \text{asin}(\sin(\varphi_1) \cdot \cos(D) + \cos(\varphi_1) \cdot \sin(D) \cdot \cos(\theta)) \quad (9)$$

$$\lambda_O = \lambda_1 + atan2(\sin(\theta) \cdot \sin(D) \cdot \cos(\varphi_1), \cos(D) - \sin(\varphi_1) \cdot \sin(\varphi_O)) \quad (10)$$

**Results**

Most researchers use high-cost hardware, DL, depend on Google SVI, and rely on known-sized objects to estimate object position from the camera. Yet, our method used only a single camera and a compact GPS device, which can achieve accurate detection and position with detailed traffic signs information.

**Case Study 1**

To validate our model in a real scenario, we experimented on provincial road PR.261, specifically from PK.29 to PK.15, located in Srok Moukh Kampoul, Kandal Province, Cambodia. The road has a width of 10 m with two lanes, without a median strip, and all the signs are on the right-hand side of single poles. To get the actual ground truth of the signs, the field as-built data was collected using a Tersus DGPS device. While the other researchers compare data collected by handheld GPS (Wang et al., 2010) and (Krylov et al., 2018) or map-based selection (Wilson et al., 2022).

**a. Video and GPS Syncing:**

To begin, the video and GPS were synchronised by identifying and correcting the time deviation between the two devices. First, the video creation time was extracted using the ffprobe tool (FFmpeg), which returned a timestamp of '2025-05-09T06:58:35.000000Z'. Second, the GPS data were extracted from a NMEA 0183 (National Marine Electronics Association) file. The UTC (Coordinated Universal Time) time, local time, latitude, and longitude were retrieved from the NMEA file by converting it into a Python script. Third, the video time and GPS time were compared. For the sake of evidence, any important road marking or structure in the GPS map was selected, and then the playback video paused at the same point.

The selected GPS point in QGIS showed the detailed timestamp of GPS, which was '09/05/2025 13:59:43.9 (UTC+7:00)', equal to '06:59:43.900 UTC' in Figure 8. The video playtime with the landmark occurred at '1:06:868', which corresponds to '06:59:41.868' in Figure 7. Thus, the time deviation between the camera and the GPS was 00:00:02.032', about 2 seconds. Finally, the 2.032 seconds were added to the video file for syncing with the GPS data. This shows the time in the camera was about 2 seconds slower than the time in the GPS. A complete Python script was also written to ensure the synchronisation.

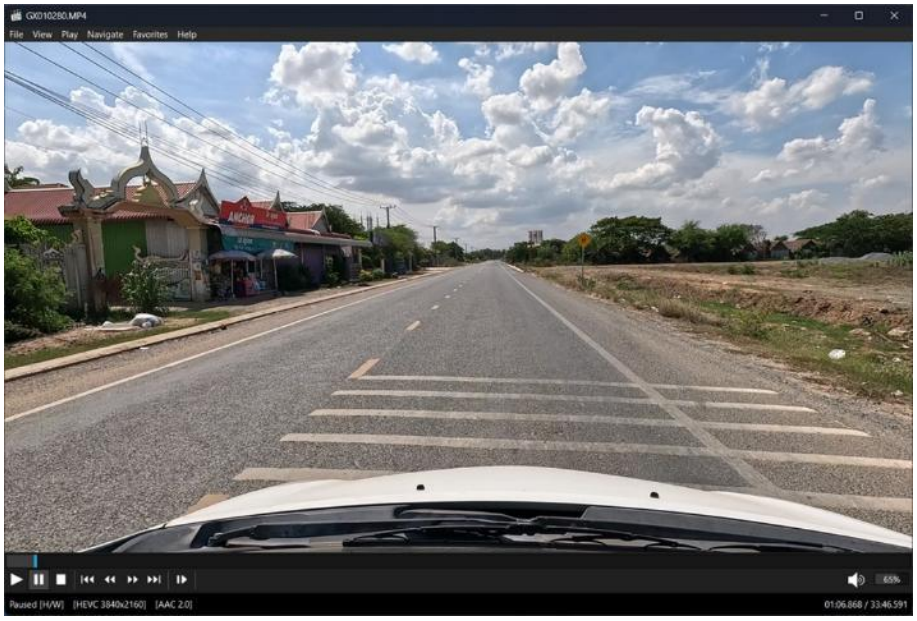


Figure 7: Video playback time paused at rumble strip

**b. Object Tracking:**

The video was recorded by a GoPro 12 in standard video mode with a 16:9 aspect ratio, linear digital lens, hyper smooth stabilisation enabled, and 4K resolution. The footage was processed with a tracking algorithm in Python. The model achieved a precision of 0.96 (Table 2). Point A (Figure 10) was fully occluded and therefore not considered a false negative (FN). Another point B was partially occluded, and the tracking algorithm could only detect it for a few frames; as a result, it could not be captured as it crossed the region line.

Table 2: Object tracking detected results in the test area. True Positive (TP), False Positive (FP), False Negative (FN)

| Case Study | Extent (km) | #Actual | #Detected | TP | FP | FN | Precision | Recall |
|---|---|---|---|---|---|---|---|---|
| 1 | 30 | 50 | 50 | 48 | 2 | 1 | 0.960 | 0.980 |
| 2 | 10 | 60 | 63 | 59 | 4 | 0 | 0.938 | 1.000 |

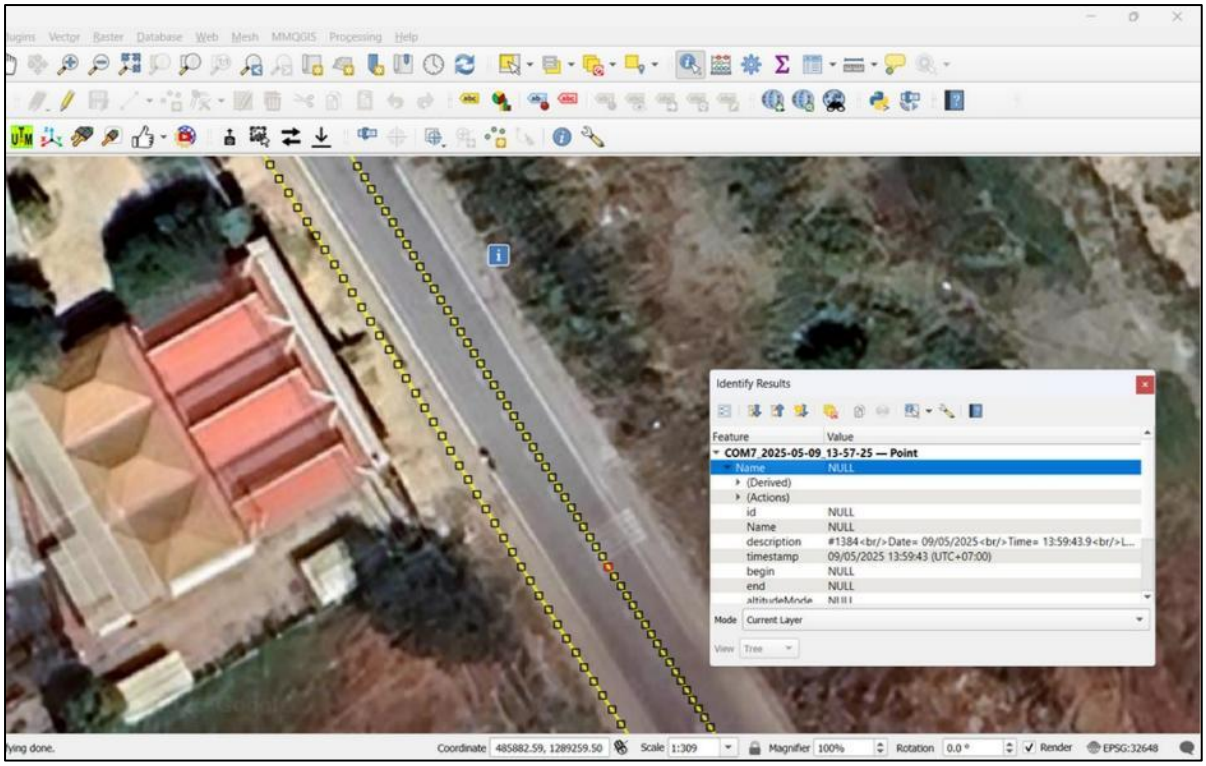



Figure 8: Check tracking point detail in QGIS

### c. Object Mapping:

After implementing the outlined method, the results are presented, showing the track ID, estimated object latitude, longitude, sign name, and status (Table 3). Some camera parameters, like height, horizontal field of view (HFOV), and vertical field of view (VFOV), need to be set. From GoPro Support, the HFOV and VFOV were obtained, and the camera height was measured during data acquisition.

Table 3: Object mapping result in Case 1

| ID | Latitude | Longitude | Name | Status |
|---|---|---|---|---|
| 427 | 11.661157 | 104.871273 | ['NO ENTRY TRUCK'] | ['Good'] |
| 17 | 11.662793 | 104.870314 | ['CHILDREN CROSSING'] | ['Good'] |
| 424 | 11.663805 | 104.869466 | ['CHILDREN CROSSING'] | ['Good'] |
| 20 | 11.664629 | 104.869159 | ['ROAD JUNCTION ON THE RIGHT'] | ['Good'] |
| 21 | 11.664806 | 104.869043 | ['SLOW DOWN'] | ['Faded'] |

For demonstration purposes, only five detected signs were selected for inclusion in the table and on the mapped signs. Figure 10 illustrates the mapped signs, where blue dots represent the

model's estimated positions and red dots (with a 5 m buffer) indicate the as-built positions. Points 2 and 5 were located on a different section of the road and hence have not been considered. While points A and B were shown on the two FN signs due to occlusion.

This information is crucial for road sign inspections, as it also monitors the names and conditions of the signs, aiding in prioritising maintenance. This data is valuable for engineers, allowing for easy and efficient identification of sign locations, names, and statuses. Figure 9 depicts the mean error at 4.633 m, with 50% of the signs having errors of less than 5 m.

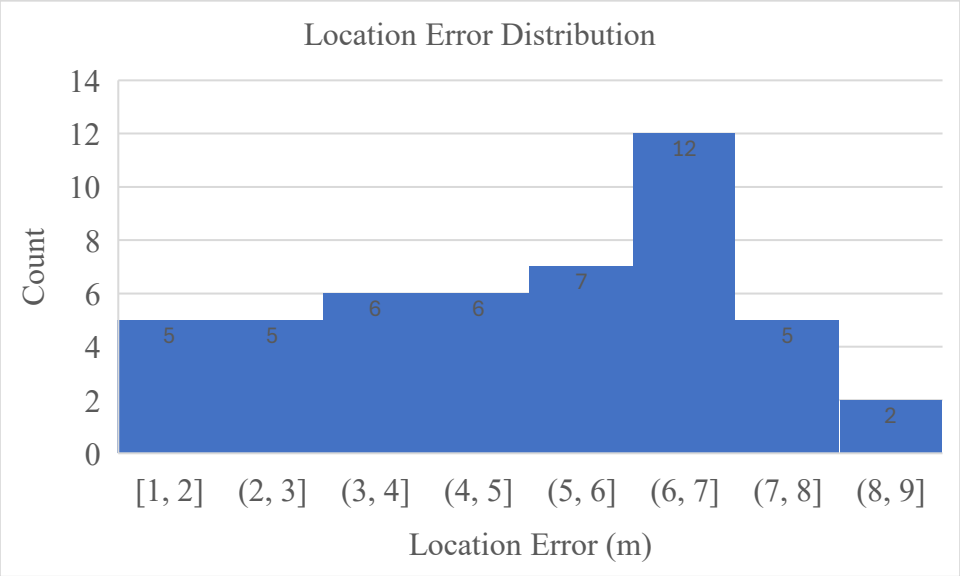



Figure 9: Location error distribution in Case 1 with a mean error of 4.633 m

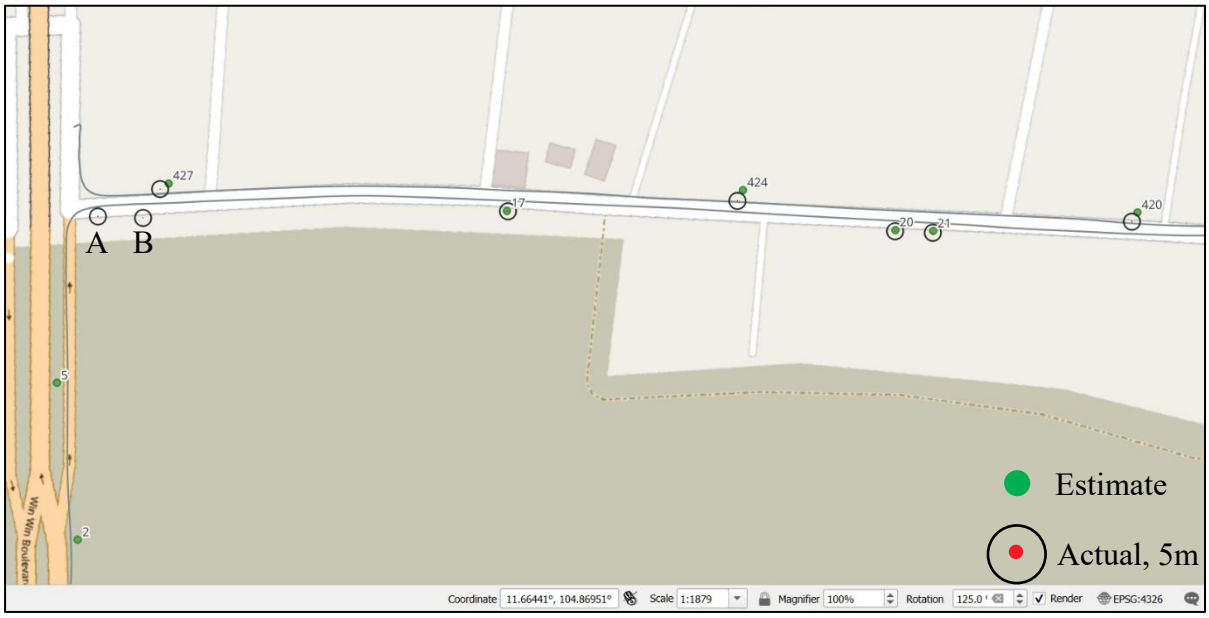



Figure 10: Map of estimated and actual signs in case 1

### d. Ablation Study

To optimise the result, several experiments were conducted. First, camera calibration was performed, and a pair of frames was undistorted using the camera calibration parameters before applying the object mapping process. In addition, the known-object size method was performed where the actual width of the signs was manually input into the data to determine the object distance from the camera (depth). Table 4 shows the comparison within our method pipeline. Interestingly, a pair of frames without any calibration produces less error in depth mean absolute error (MAE) and position.

Table 4: Comparison within our method

| Nº | Method | Depth Mean Error | Depth MAE | Position Mean Error |
|---|---|---|---|---|
| 1 | Pair Frames | -0.433 | 2.498 | 4.633 |
| 2 | Pair Frames with Calibration | 0.455 | 2.699 | 4.979 |
| 3 | Single Frame with Known-Object Size | 0.083 | 2.555 | 4.696 |

**Case Study 2**

To verify the performance on the suburban median road, we deployed the model on national road NR.3, from PK.36 to PK.46, right side driven only, in Srok Kong Pisey, Kompong Speu Province, Cambodia. The road width is four lanes, with a median, and signs are located on the right side or centre, while some are on shared poles. The ground truth has been manually selected from the traffic signs inventory recorded by the Department of Public Works and Transport in Kompong Speu Province. The accuracy of the position varies because this data was collected several years ago using handheld GPS devices. As a result, comparing locations was more challenging than assessing the recall and precision of the detected signs.

First, the video and GPS synchronisation was performed as in Case Study 1. The time deviation was found to be 2 seconds, and the synchronised data were prepared as a reference in the data sheet. Second, object tracking was applied to the video, which was recorded at 1080p to eliminate the computational overhead during tracking. The inference could reach up to 70 fps with ByteTrack on a Nvidia GTX1060Ti 14GB GPU. Table 2 shows a slightly lower precision of 0.93, due to the false detections of billboard or commercial signs in the complex environment. One sign could not be detected since it was missing at the time of recording.

Third, the object mapping was conducted. The result of the traffic sign's location, name, and status was generated, and then the histogram was plotted as in Figure 11. This stated the estimated location mean error at 4.199 m, with 67% of the signs having errors less than 5 m. Figure 12 shows the estimated points imported into QGIS along with the sign names. This visualisation provides engineers with an effective tool for monitoring sign locations and planning maintenance activities.

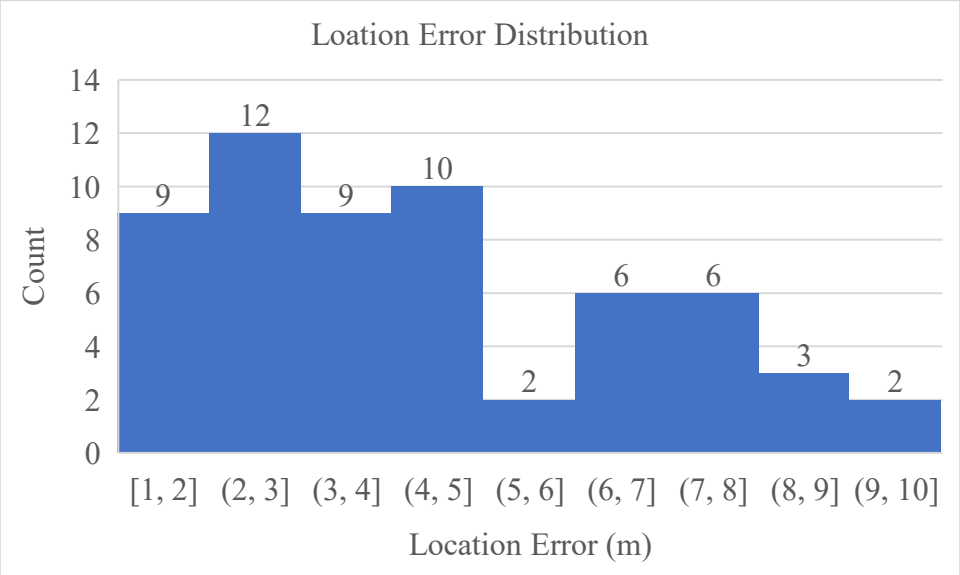



Figure 11: Location error distribution in Case 2 with a mean error of 4.199 m

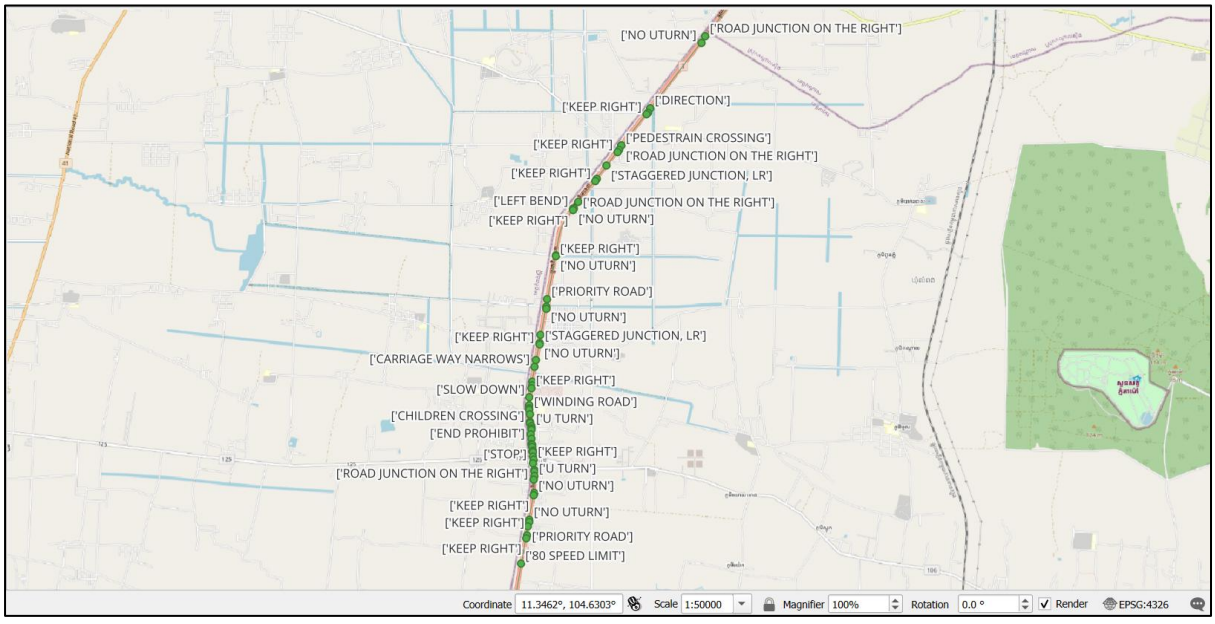



Figure 12: Map of estimated and actual signs in case 2. Green dots are the estimated result with the sign's name label.

**Comparison to other authors**

It is quite difficult to compare the location accuracy with other research, as their methods rely on external sensors, a mapping platform, DL, and different ground truths. Thus, only similar cases are selected for comparison. In Table 5, our method achieved a mean error of 4.63 meters, outperforming Wilson et al. (2022) (5.81m) and Wang et al. (2010) (5-19m). This comparison demonstrates the viability of mono-camera setups for cost-effective traffic signs mapping.

Table 5: Location error comparison with other authors

| Author | Method | Mean Error |
|---|---|---|
| Wilson et al. (2022) | GPS RetinaNet, on Mono Camera | 5.81 m |
| Wang et al. (2010) | PCA, SVM, on Stereo Camera | 5-19 m |
| Our study | Object Detection, Pair Frames, on Mono Camera | 4.63 m |

**Discussion**

The object mapping achieves a mean error of less than 5 m, based on GPS data and the detection model. Several factors contribute to inaccuracies, including GPS positioning, synchronisation between video and GPS data, object detection results, and the visibility of objects in the scene. Firstly, the GNSS receiver used in this study employs a stand-alone positioning method, which does not provide sub-centimetre accuracy like some commercial-grade systems or alternative modes. This limitation can be addressed economically by using RTK positioning or network RTK mode in conjunction with other receivers, as well as through sensor fusion with an Inertial Measurement Unit (IMU) and odometry. Secondly, aligning video and GPS data poses a challenge. Since the two devices are not interconnected, manual adjustments may introduce some discrepancies. Accurate alignment could be achieved using a USB camera module that overlays real-time GPS data onto camera frames, resulting in a video format similar to that of a dashcam. Thirdly, the object detection model cannot guarantee absolute precision in real-world scenarios. The model may exhibit various gaps depending on the environment in which it is used. These gaps need to be analysed and addressed using appropriate datasets and hyperparameters during training. Finally, objects may become invisible to the camera due to occlusion. For example, a sign may be blocked by street vendors' umbrellas (Case Study 1) or temporarily obscured when a vehicle overtakes a truck near a sign. Such scenarios highlight the importance of considering occlusions in real-world applications. In summary, inaccuracies in object positioning can be mitigated by integrating additional sensors, overlaying GPS data

directly onto video frames, expanding the dataset, utilising advanced DL techniques, and considering driving patterns.

## Conclusion and Recommendation

In conclusion, this study addressed the challenge of building a traffic signs inventory, which is an expensive, labour-intensive, and time-consuming task. Our approach provides a cost-effective and efficient alternative, delivering not only sign names and locations but also information on sign condition, which is valuable for maintenance prioritisation. Our result achieves a mean error of 4.633 m, which is in general alignment with a previous study (Wilson et al., 2021), who used DL to detect and predict the GPS location of traffic signs in the US. This work advances remote sensing by minimising the time required for object mapping and by proficiently combining DL, computer vision, and GNSS to improve data collection. Moreover, it is tailored for road or highway engineers to update the sign inventory regularly, helping to save lives by preventing traffic accidents caused by missing signs.

Nevertheless, some limitations remain. Processing time during the tracking step is a key constraint which could be addressed by leveraging the model to work in real-time processing with a USB camera module and an updated detection and tracking model architecture. Future work should focus on improving the model's robustness, running it on an edge device, and sharing it with all stakeholders. More datasets would be collected for other road assets. This scalable model could be developed to map other road facilities such as manholes, streetlights, guardrails, bridges, or culverts. Embedding GPS and camera modules into edge devices would allow real-time inference, making the tool scalable for deployment by road authorities and public works departments. Such inventories are essential for analysing road maintenance and development planning.

## References

Almutairy, F., Alshaabi, T., Nelson, J., & Wshah, S. (2021). ARTS: Automotive Repository of Traffic Signs for the United States. *IEEE Transactions on Intelligent Transportation Systems*, *22*(1), 457–465. https://doi.org/10.1109/TITS.2019.2958486

Balado, J., González, E., Arias, P., & Castro, D. (2020). Novel Approach to Automatic Traffic Sign Inventory Based on Mobile Mapping System Data and Deep Learning. *Remote Sensing*, *12*(3), 442. https://doi.org/10.3390/rs12030442

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., & Beijbom, O. (2020). *nuScenes: A multimodal dataset for autonomous driving* (No. arXiv:1903.11027). arXiv. https://doi.org/10.48550/arXiv.1903.11027

Campbell, A., Both, A., & Sun, Q. (Chayn). (2019). Detecting and mapping traffic signs from

Google Street View images using deep learning and GIS. *Computers, Environment and Urban Systems*, *77*, 101350. https://doi.org/10.1016/j.compenvurbsys.2019.101350

Chaabane, M., Gueguen, L., Trabelsi, A., Beveridge, R., & O'Hara, S. (2021). *End-to-end Learning Improves Static Object Geo-localization in Monocular Video* (No. arXiv:2004.05232). arXiv. https://doi.org/10.48550/arXiv.2004.05232

Chen, T., & Ren, J. (2023). MFL-YOLO: An Object Detection Model for Damaged Traffic Signs. *ArXiv*. https://arxiv.org/abs/2309.06750

Core Group. (n.d.). Cohac∞ QZNEO, Michibiki compatible multi-GNSS compact dual-frequency receiver. Retrieved February 10, 2025, from https://www.core.co.jp/service/industrial/gnss/receiver/qzneo

Dutta, A., & Zisserman, A. (2019). The VIA Annotation Software for Images, Audio and Video. *Proceedings of the 27th ACM International Conference on Multimedia*, 2276–2279. https://doi.org/10.1145/3343031.3350535

Exiftool. (n.d.). Geotagging with ExifTool, Tip Time Synchronization. Retrieved February 5, 2025, from https://exiftool.org/geotag.html

FFmpeg. (n.d.). FFmpeg: The ultimate multimedia framework, https://ffmpeg.org

Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, *32*(11), 1231–1237. https://doi.org/10.1177/0278364913491297

GoPro Support. (2023). HERO12 Black: Digital Lenses FOV Information, Retrieved June 10, 2025, from https://community.gopro.com/s/article/HERO12-Black-Digital-Lenses-FOV-Information?language=en_US

Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., & Igel, C. (2013). Detection of traffic signs in real-world images: The German traffic sign detection benchmark. *The 2013 International Joint Conference on Neural Networks (IJCNN)*, 1–8. https://doi.org/10.1109/IJCNN.2013.6706807

Jocher, G., Chaurasia, A. and Qiu, J. (2023) YOLO by Ultralytics. https://github.com/ultralytics/ultralytics

Krylov, V. A., Kenny, E., & Dahyot, R. (2018). Automatic Discovery and Geotagging of Objects from Street View Imagery. *Remote Sensing*, *10*(5), 661. https://doi.org/10.3390/rs10050661

Kunsági-Máté, S., Pető, L., Seres, L., & Matuszka, T. (2025). *Accurate Automatic 3D Annotation of Traffic Lights and Signs for Autonomous Driving* (No. arXiv:2409.12620). arXiv. https://doi.org/10.48550/arXiv.2409.12620

Mapillary. (n.d.). Map Explore with show traffic sign. Retrieved July 20, 2025, from https://www.mapillary.com/app/?lat=11.443260813342093&lng=104.80581132205634&z=14.293191627843967&trafficSign=all

MPWT, Ministry of Public Works and Transport (1995). Appendix on road traffic signs, https://mpwt.gov.kh/index.php/kh/documents/other/86

Ning, H., Li, Z., Yu, M., & Yin, W. (2025). SIM: A mapping framework for built environment auditing based on street view imagery. *ArXiv*. https://arxiv.org/abs/2505.24076

Wang, K. C. P., Hou, Z., & Gong, W. (2010). Automated Road Sign Inventory System Based on Stereo Vision and Tracking: Automated road sign inventory system. *Computer-Aided Civil and Infrastructure Engineering*, *25*(6), 468–477. https://doi.org/10.1111/j.1467-

8667.2010.00657.x

Wilson, D., Alshaabi, T., Van Oort, C., Zhang, X., Nelson, J., & Wshah, S. (2022). Object Tracking and Geo-Localization from Street Images. *Remote Sensing*, *14*(11), 2575. https://doi.org/10.3390/rs14112575

Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., & Wang, X. (2022). *ByteTrack: Multi-Object Tracking by Associating Every Detection Box* (No. arXiv:2110.06864). arXiv. https://doi.org/10.48550/arXiv.2110.06864

Zhao, G., Ma, F., Qi, W., Zhang, C., Liu, Y., Liu, M., & Ma, J. (2025). *TSCLIP: Robust CLIP Fine-Tuning for Worldwide Cross-Regional Traffic Sign Recognition* (No. arXiv:2409.15077). arXiv. https://doi.org/10.48550/arXiv.2409.15077

Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., & Hu, S. (2016). Traffic-Sign Detection and Classification in the Wild. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2110–2118. https://doi.org/10.1109/CVPR.2016.232