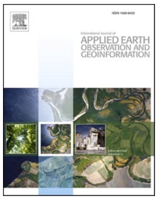
Contents lists available at ScienceDirect

International Journal of Applied Earth Observations and Geoinformation

journal homepage: www.elsevier.com/locate/jag



Low-cost mobile mapping system solution for traffic sign segmentation using Azure Kinect

Zhouyan Qiu [a,b,*], Joaquín Martínez-Sánchez [a], Víctor Manuel Brea [c], Paula López [c], Pedro Arias [a]

[a] CINTECX, Universidade de Vigo, Applied Geotechnology Group, Vigo 36310, Spain
[b] ICT & Innovation Department, Ingeniería Insitu, Vigo 36310, Spain
[c] Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela, Santiago de Compostela 15782, Spain

ARTICLE INFO

Keywords:
Time of Flight camera
Traffic sign detection and segmentation
Multi sensor system
Mobile mapping system

ABSTRACT

The mobile mapping system (MMS) could become the foundation of digital twins and 3D modeling, and is widely applicable in a variety of fields, such as infrastructure management, intelligent transportation systems, and smart cities. However, data collected by MMS is extensive and complex, making data processing difficult. We present a novel method for segmenting urban assets (specifically in this case study traffic signs) with a lower-cost Azure Kinect and automatic data processing workflows. First, it was necessary to verify the reliability of this approach using the Time of Flight (ToF) camera from Azure Kinect to detect road signs outdoors. Using the data generated by the ToF camera, we then extracted the Region of Interest (ROI) quickly and efficiently. After transforming the ROI to the RGB image, we obtained the traffic sign area through a hybrid color-shape based method. In addition, we calculated the distance between the traffic sign and Azure Kinect based on the depth image. The Coefficient of Variation c_v averaged 1.1%. It is thus evident that Azure Kinect is reliable for outdoor traffic sign segmentation. Our algorithm has been compared with deep learning algorithms. According to our analysis, our algorithm has an accuracy of 0.8216, while the accuracy of deep learning is 0.7466, which indicates that our solution is more flexible and cost-effective.

1. Introduction

Since the first Mobile Mapping System (MMS) was developed at the Ohio State University in the 1990s, it has become one of the most important topics in 3D geospatial data acquisition. Typically, mobile mapping systems consist of three components: mapping sensors, positioning and inertial systems, and time-referencing units (Puente et al. (2013)). In MMS, a portable imaging sensor attached to a moving device, such as a car, train, backpack, robot, or drone, is able to capture 2D or 3D geometric environmental information. MMS plays a very significant role in 3D modeling and Digital Twins, which is an initial step toward their many applications such as infrastructure management systems, intelligent transportation systems, and smart cities.

In a mobile mapping system, laser scanners and cameras are usually the main exteroceptive sensors. Studies over the past two decades have focused on image-based (Cavegn et al. (2015),Cavegn and Haala (2016)), laser-based (Jaakkola et al. (2008),Puentea et al. (2011), Serna and Marcotegui (2013)), and hybrid image/laser multi-sensor mobile mapping system (Paparoditis et al. (2012)). Most research on MMS has focused on limited sensors such as RGB cameras, 2D and 3D laser scanners. Up to now, little attention has been paid to the outdoor use of the Time-of-flight (ToF) camera. This is primarily due to the fact that they were designed for indoor environments.

A ToF camera is able to create a depth image, each pixel of which encodes the distance to the corresponding point in the scene by measuring the phase-shift of reflected infrared (IR) light (Hansard et al. (2012)). So far, only a few outdoor applications have been investigated. Elfiky et al. (2015) made use of the Kinect V2, which provides the information of color, depth, and intensity of reflectance, to model outdoor apple trees. Moreover, fruit detection and localization is a major outdoor application of ToF cameras. Fu et al. (2020) have summarized the related researches and challenges. De Cubber et al. (2011) analyzed the outdoor terrain traversability for robot navigation using a CamCube2 ToF camera. Other outdoor applications of the ToF camera include parking assistance (Steinbaeck et al. (2018),Peláez et al. (2019)) and on-street parking statistics (Nebiker et al. (2021)). In this study, we develop a new understanding of the outdoor capability of ToF camera and investigate the results of traffic sign segmentation using ToF camera.

[*] Corresponding author.
E-mail address: zhouyan.qiu@uvigo.es (Z. Qiu).

https://doi.org/10.1016/j.jag.2022.102895
Received 29 April 2022; Received in revised form 7 June 2022; Accepted 24 June 2022

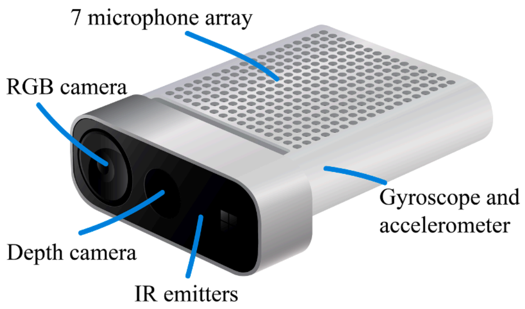


Fig. 1. Azure Kinect DK hardware specifications.

Table 1
Azure Kinect DK sensor specifications (Microsoft (2022)). **NFOV** indicates Narrow field-of-view depth mode; **WFOV** indicates Wide field-of-view depth mode; **FPS** indicates Frames-per second. By implementing 2 × 2 binning modes, the depth camera can extend the measuring range compared to the corresponding unbinned modes. However, binning reduces image resolution.

Sensor	Details	FPS	Resolution
Audio	7-mic circular array	-	-
Motion sensor (IMU)	LSM6DSMUS: 3-axis accelerometer, 3-axis gyroscope	-	-
RGB Camera	OV12A10 12MP CMOS sensor rolling shutter sensor	0, 5, 15, 30	3840 × 2160; 2560 × 1440; 1920 × 1080; 1280 × 720; 2048 × 1536
		0, 5, 15	4096 × 3072
Depth camera	NFOV unbinned	0, 5, 15, 30	640 × 576
	NFOV 2 × 2 binned (SW)	0, 5, 15, 30	320 × 288
	WFOV 2 × 2 binned	0, 5, 15, 30	512 × 512
	WFOV unbinned	0, 5, 15	1024 × 1024

Automatic traffic sign detection and recognition (TSDR) plays an important role in the development of intelligent transportation systems (ITS) and advanced driver assistance systems (ADAS). A number of studies have been conducted over the past two decades that have provided significant insight into this topic. According to recent literature, the most widely-used methods for traffic sign detection are color-based, shape-based, and machine-learning-based (Wali et al. (2019),Ellahyani et al. (2021)). Such approaches, however, have only focused on the RGB camera. A recent survey by Zou et al. (2019) summarized the difficulties and challenges in TSDR including illumination changes, motion blur, bad weather, and real-time detection. To address these problems, we combine IR and depth images gathered by the ToF camera with RGB images in our research.

Furthermore, data processing for MMS data is challenging because of the large amount and complexity of the data, which leads to high-cost solutions (Sairam et al. (2016)). In the data preprocessing stage, one of the solutions to consider is down-sampling (Rashdi et al. (2022)), which reduces computational complexity (ElRafey and Wojtusiak (2017)). In our research, we present a novel approach that utilizes the connection between RGB images, depth images, and IR images to reduce the operational area through logical operation, thereby allowing for higher efficiency.

Given the above background, the overarching goal of the paper is to provide a low-cost MMS solution for traffic sign segmentation based on Azure Kinect. By utilizing the RGB, depth, and IR images, our segmentation algorithms are able to both improve the quality and the speed of the process. Moreover, we compared our approach with the deep learning object detection methods.

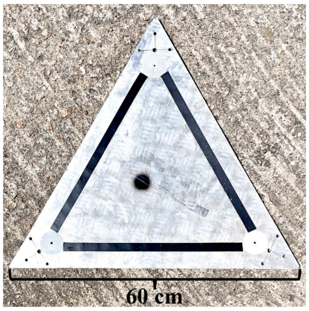


Fig. 2. This calibration board is an equilateral triangle with a side length of 60 cm and is highly reflective.

The remaining part of the paper proceeds as follows. Section 2 introduces the sensor and datasets we use. Section 3 proposes the method of data pre-processing and traffic sign segmentation. In Section 4, we shows the experiment results and compares the results with Deep learning methods. Section 5 presents a summary of the major contributions of this study.

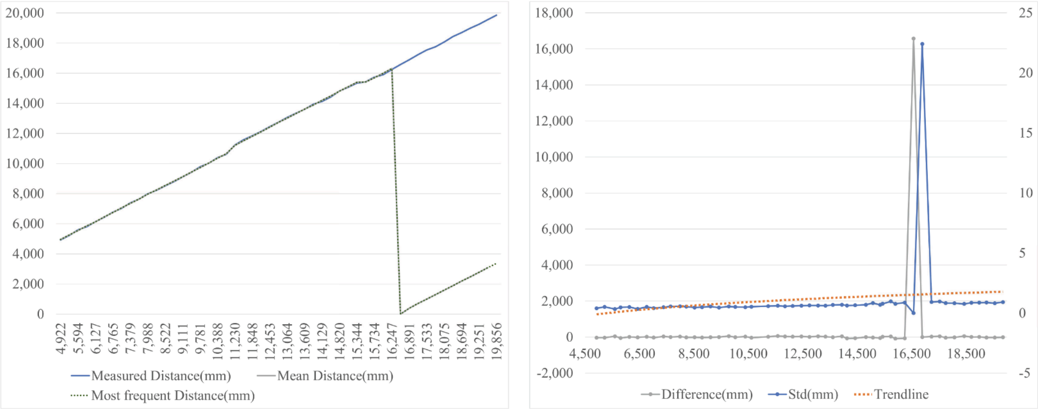
2. Sensors and datasets

2.1. Sensors

Azure Kinect provides a multi-sensor platform with advanced sensors (Fig. 1). The price of Azure Kinect is $399, making it a cost-effective solution. The main Azure Kinect DK sensor specifications are listed in Table 1. In this study, the chosen mode is NFOV 2x2 binned (depth camera resolution: 320 × 288 pixels), and RGB camera resolution is 2048 × 1536 pixels.

Tölgyessy et al. (2021) evaluated color and material effects on sensor performance and the outdoor environment performance of Azure Kinect. There is a correlation between standard deviation and reflectance, that is, materials with lower reflectance have a higher standard deviation. Besides, from the official online document (Microsoft (2022)), the operating range depends on the object's reflectivity. In outdoor cases, if we focus only on high-reflectivity targets, such as traffic signs, the measuring distance will be longer than described in the document, i.e., $0.5 - 5.46m$ for NFOV 2x2 binned mode. Tölgyessy et al. (2021) also tested the performance of Azure Kinect in the outdoor environment and mentioned that there is much noise around the testboard. They concluded that direct sunlight itself will not cause a large amount of noise and the NFOV mode provides better results in the outdoor environment. In this regard, focusing on high-reflectivity objects could produce interesting findings about the functionality of traffic sign detection and segmentation.

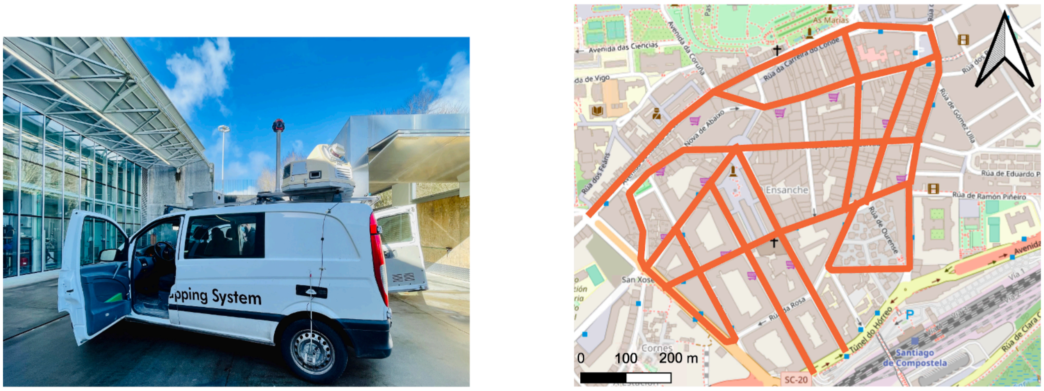
A simple statistical analysis was performed in order to investigate the relationship between the distance and the reliability when measuring high-reflectivity objects. Our calibration board (Fig. 2) is made from high reflective material, similar to traffic signs, and it was positioned upright on the ground. We started measurements from $5,000mm$ and took 300 depth images every approximately $300mm$ while simultaneously measuring the distance between the camera and the calibration board with a Electronic Distance Measurer(EDM). Then, we calculated the distances from the camera to the calibration board and compared those distances with those determined by the distance meter. The results of the correlational analysis are summarized in Fig. 3. What is striking about the values in Fig. 3a is the abrupt drop in depth values measured from about $16,500mm$. It is caused by the principle of ToF cameras. ToF cameras obtain depth information by measuring the phase shift between the emitted and reflected signals. In most commercially available ToF cameras, the phase ϕ is computed assuming that it is within the range $[0, 2\pi]$. Therefore, the maximum measuring range of a ToF camera without ambiguity is



(a) Blue: Distances measured by EDM. Grey: Mean distances. Dark green dots: Most frequent distances.

(b) The statistics after phase unwrapping. Gray: Distance between the measured value and the most frequent value. Blue: Standard deviation of depth values captured at each point. Orange: Standard deviation trendline.

Fig. 3. The comparison between the measured distances, the mean distances and the most frequent distances. Fig. 3a compares the three values at each point. Fig. 3b indicates the difference between the measured distances and the most frequent distances, as well as the standard deviation of the depth measurements at each point.



(a) The mobile mapping system used for data acquisition

(b) Map of van driving trajectory

Fig. 4. Acquisition platform and trajectory map.

$$d_{\max} = \frac{c}{2f} \tag{1}$$

where c is the speed of light, f is the modulation frequency.

If the measured distance is greater than d_{max}, the resulting distance d is much smaller than its actual distance $d + n \times d_{max}$, where n is the number of wrappings. This is called **phase wrapping**. The actual distance is calculated by Eq. 2 (Hansard et al. (2013)):

$$\mathbf{X}_p(n_p) = \frac{d_p + n_p d_{\max}}{d_p}\mathbf{X}_p \tag{2}$$

where the measured distance d_p equals $\|\mathbf{X}_p\|$, $\mathbf{X}_p(n_p)$ is the unwrapped 3D point, and n_p is the number of wrappings. Therefore, when $n = 1$, $\mathbf{X}_p(n_p) = d + n \times d_{max}$. This explains the sudden fall at value $16,500mm$ in Fig. 3. We assumed that d_{max} is approximately $16,500mm$, and the statistics in Fig. 3b are based on phases unwrapping. It is apparent from Fig. 3b that the measurements fluctuate wildly between $16,200mm$ and $17,000mm$. What is interesting about the data in Fig. 3b is that the difference and standard deviation tend to be consistent again after the fluctuation. As a result, the variation between $16,200mm$ and $17,000mm$ can be attributed to the unstable measurements between d and $d + d_{max}$. Once the distance exceeds $17,000mm$, the measured value tends to stabilize again. Since the distance meter was placed in front of the Azure Kinect, there is a difference of a few centimeters in the value they provide, but this is to be expected. The trend (Fig. 3b) in the standard deviation of the depth value measured by the ToF camera indicates that, despite the fact that the standard deviation gradually increases with the distance, it is always less than 1 mm. The results are therefore consistent. As a result, the depth values of traffic signs measured by Azure Kinect are considered reliable. Because of perspective, the traffic sign at a great distance appears small in the picture. Since we do not consider the case for distances larger than $16,200mm$, phase unwrapping is not relevant to our study.

2.2. Datasets

The measurement was done in Santiago de Compostela, Spain. We collected the Santiago dataset using our mobile mapping system (Fig. 4a). Sensors attached to the van are listed as follows:

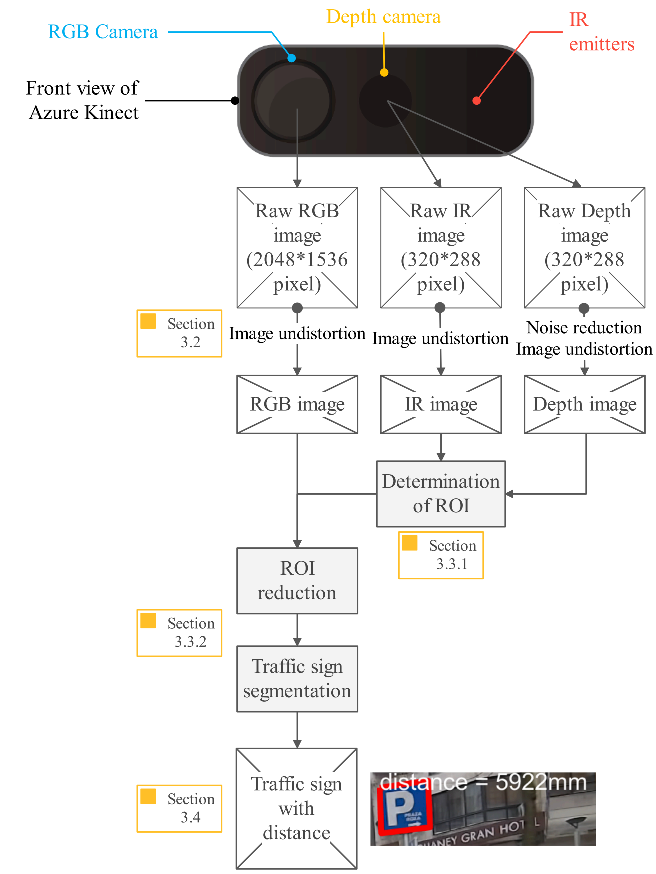


Fig. 5. Overall workflow of this study.

- Lynx Mobile Mapper^{TM} mobile mapping system
- Ladybug5 of six Sony® ICX655 CCD sensors
- Microsoft Azure Kinect

Only data captured by Azure Kinect were used in this study. Fig. 4b provides the map of the driving trajectory. The driving distance is around 10 km and the recording time was 28 min and 15 s with the rate of 15 FPS, which means the total number of images is 25, 425. During that period the weather changed from cloudy to showery.

3. Methods and results

3.1. Workflow

The objective of our method is to segment the traffic sign area using a combination of depth images, IR images, and RGB images. There are three main steps to the proposed method (Fig. 5): First, pre-process the data, including noise reduction, camera calibration and image registration. Second, detect the traffic sign area and extract the border of the traffic sign. Finally, calculate the distance from camera to the traffic sign.

3.2. Step 1: Data pre-processing

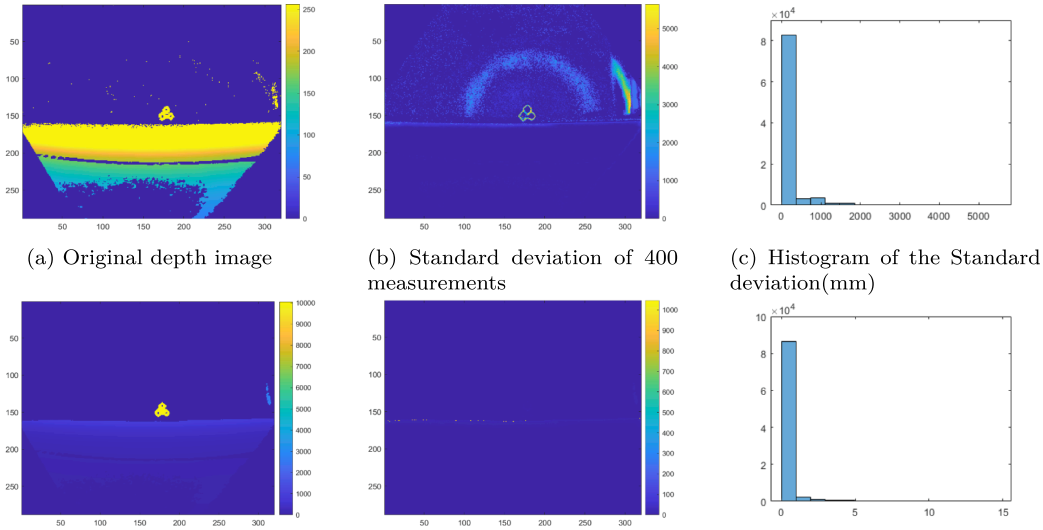
Data pre-processing includes noise reduction (only for depth images), camera calibration and image registration.

3.2.1. Noise reduction

As can be seen from Fig. 6a, the depth images captured by ToF have a lot of noise. A large proportion of the noise is due to a false phase shift introduced by the differences in distances inside a solid angle, which is called flying pixels (Lindner et al. (2010)). Given that flying pixels are randomly distributed in each frame, we propose a simple yet effective method using the relationship between two frames, which is presented in Algorithm 1.

Algorithm 1. Denoising by removing flying pixels

```
Data: $n$ indicates the number of depth images.
      $D = \{D_1, D_2, D_3, \dots, D_n\}$ is the depth image series sorted by
      time. $i, j$ represent the coordinates of the pixels in the depth
      image, $w$ is the width of the depth image, and $h$ is the height
      of the depth image. $DN = \{DN_1, IDN_2, DN_3, \dots, DN_n\}$ is
      initialized as a zero three-dimensional matrix with the same
      dimension as $D$.
Result: $DN = \{DN_1, DN_2, DN_3, \dots, DN_n\}$ now is the depth image
        series after noise reduction.
for $nn = 1$ to $n - 1$ do
    for $i = 1$ to $w$ do
        for $j = 1$ to $h$ do
            while $D_{nn}(i,j) \& D_{nn+1}(i,j)$ do
                $DN_{nn}(i,j) \leftarrow D_{nn}(i,j)$
            end
        end
    end
end
```



(a) Original depth image

(b) Standard deviation of 400 measurements

(c) Histogram of the Standard deviation(mm)

(d) Depth image after denoising

(e) Standard deviation of 399 denoised images

(f) Histogram of the Standard deviation after denoising(mm)

Fig. 6. Comparison before and after noise reduction.

For the purposes of testing the reliability of our algorithm, 400 measurements were made between the Azure Kinect and the calibration board (Fig. 2) at a distance of $10,000mm$. This experiment was performed at noon with clear clouds to simulate the situation with the most flying pixels. Fig. 6 compares the image and standard deviation before and after denoising. It is apparent from Fig. 6a and 6b that the original depth image has a high degree of noise, resulting in a high standard deviation. After processing, most of the noise is removed and the calibration board is easily discernible. A comparison of the standard deviation before and after processing can clearly be observed from the histogram (Fig. 6c and 6f). In addition, the mean standard deviation before and after the process is $130.8076mm$ and $0.2877mm$, respectively.

3.2.2. Camera calibration and registration

Although Microsoft provides the camera parameters of Azure Kinect, as noted by Kettelgerdes et al. (2021), camera calibration parameters of ToF cameras are influenced by temperature. Therefore, re-calibration before measurement is necessary. Following camera calibration, we derived the intrinsic and undistortion parameters of the two cameras, after which we obtained the extrinsic rotation and translation between the two cameras. ArUco Markers (Garrido-Jurado et al. (2014)) are used in these steps (Fig. 7).

The calibration relies on the Brown Conrady distortion model (Brown (1971)). Through Camera Calibration function provided by OpenCV (2022), we were able to obtain the intrinsic and distortion coefficients of the depth camera and RGB camera:

The intrinsic parameter k_{rgb} and the distortion matrix $dist_{rgb}$ of depth camera are shown in Eq. (3) and (4):

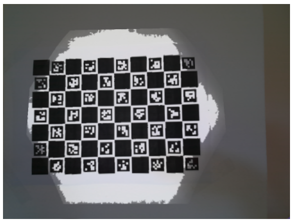


Fig. 7. The calibration board used for camera calibration and registration.

$$k_{rgb} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 88.19450402 & 0 & 103.52517722 \\ 0 & 87.87608339 & 77.67123891 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

$$\begin{aligned} &dist_{rgb} = [K_1, K_2, P_1, P_2, K_3, K_4, K_5, K_6] = \\ &[0.98674232, -3.21647782, -0.00100323, -0.00071279, \\ &1.73283007, 0.86655331, -3.04126408, 1.65822397] \end{aligned} \tag{4}$$

Here, f_x and f_y are camera focal lengths and c_x and c_y are optical centers expressed in pixels coordinates. $[K_1, K_2, K_3, K_4, K_5, K_6]$ is the radial distortion coefficient and $[P_1, P_2]$ is the tangential distortion coefficient.

Similarly, the intrinsic parameter k_{depth} and the distortion matrix $dist_{depth}$ of depth camera are shown in Eq. (5) and (6):

$$k_{depth} = \begin{bmatrix} 28.50048332 & 0 & 15.38805035 \\ 0 & 28.38939742 & 17.10743697 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

$$\begin{aligned} &dist_{depth} = [10.70222604, -21.84932743, -0.00680356, 0.0078872367, \\ &14.37924308, 10.97386330, -18.62264273, 9.47844533] \end{aligned} \tag{6}$$

After undistortion, we can calculate the rotation matrix and translation vector of the two coordinate systems by utilizing the extrinsic parameter matrix of the camera:

$$\begin{aligned} P_{depth} &= R_{depth}P + T_{depth} \rightarrow P = R_{depth}^{-1}P_{depth} - R_{depth}^{-1}T_{depth} \\ P_{rgb} &= R_{rgb}P + T_{rgb} \\ &= R_{rgb}R_{depth}^{-1}P_{depth} + T_{rgb} - R_{rgb}R_{depth}^{-1}T_{depth} \\ &= RP_{depth} + T \end{aligned} \tag{7}$$

Here, P is the coordinates of in the world coordinate system,P_{rgb} is the spatial coordinates in the RGB camera coordinates, P_{depth} is the spatial coordinates in the depth camera coordinates. Therefore, the rotation matrix R and translation vector T can be estimated as follows:

$$\begin{aligned} R = R_{rgb}R_{depth}^{-1} &= \begin{bmatrix} 0.99993838 & 0.00991586 & -0.00499079 \\ -0.00926742 & 0.99315444 & 0.11644041 \\ 0.00611124 & -0.11638698 & 0.99318514 \end{bmatrix} \\ T = T_{rgb} - RT_{depth} &= [-0.03711103, -0.00334417, -0.03943639] \end{aligned} \tag{8}$$

After calculating the extrinsic parameters, the depth images can be projected to the RGB image through the equation.

$$
\begin{aligned}
&p_{depth} = k_{depth}P_{depth} \rightarrow P_{depth} = k_{depth}^{-1}p_{depth} \\
&p_{rgb} = k_{rgb}P_{rgb} = k_{rgb}\left(RP_{depth} + T\right) = k_{rgb}\left(Rk_{depth}^{-1}p_{depth} + T\right)
\end{aligned}
\tag{9}
$$

where p_{rgb} is the projected coordinates on the 2D RGB image coordinates. p_{depth} is the projected coordinates on the 2D depth image coordinates.

As can be seen from Fig. 1, the depth camera generates both depth images and IR images. Therefore, the transformation of IR images are the same as that of depth images.

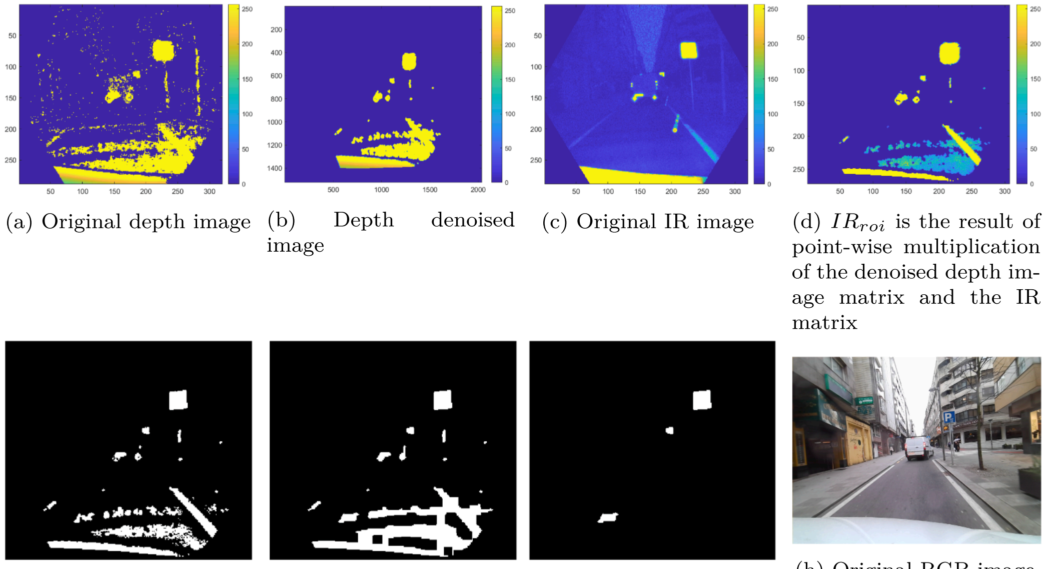
3.3. Step 2: Traffic sign detection and segmentation

3.3.1. Determination of Region of Interest(ROI)

It can be seen from Fig. 8c and 8b that the traffic sign is clearly identified on the denoised depth map and infrared image. So, if we combine depth map and IR map, we are able to get the Region of Interest (ROI) following the procedure detailed in Algorithm 2. Intermediate steps and results are set out in Fig. 8.

Algorithm 2. Determination of Region of Interest(ROI)

```
Data: $I_{Depth}$ is the depth image matrix after denoising and
      undistortion, $I_{IR}$ is the corresponding IR image matrix.
Result: $ROI = \{ROI_1, ROI_2, ROI_3, \ldots, ROI_n\}$ is the ROI series, n
        is the number of ROI.
/* Function BsrMatrix constructs Block Sparse Row matrixs
   with dense sub matrices, which can speed up
   calculations (Scipy [29]).                              */
$BSR_{Depth} \leftarrow$ BsrMatrix($I_{Depth}$);
/* Function .Multiply indicates point-wise multiplication
   */
$IR_{roi} \leftarrow BSR_{Depth}$.Multiply($I_{IR}$);
/* Function AdaptiveThreshold applies aaptive image
   threshold using **Gaussian weighted mean** in the 5 × 5
   neighborhood.                                           */
$BW_{roi} \leftarrow$ AdaptiveThreshold($IR_{roi}$);
/* Closing is defined as dilatation followed by erosion
   with the same structuring element used in both
   operations (Rafael C. Gonzalez [30]). The kernel we
   use here is a square with a side length of 10 pixels.
   */
$kernel[0:10, 0:10] \leftarrow 1$;
$BW_{roi\_close} \leftarrow (BW_{roi} \oplus kernel) \ominus kernel$;
/* Function FindContours is to find white objects from
   black background. Result $C = \{C_1, C_2, C_3, \ldots, C_i\}$ is the
   detected contours. $i$ is the number of contours.        */
$C \leftarrow$ FindContours($BW_{roi\_close}$);
$n \leftarrow 1$;
/* Remove anomalous regions by region properties. **Area**
   refers to the area of the contour, and **Extent**
   represents the ratio of the area of the contour to
   that of the bounding rectangle.                         */
for $k = 1$ to $i$ do
    if $100 \leq C_k.Area \leq 500 \& C_k.Extent \geq 0.5$ then
        $ROI_j \leftarrow C_k, n \leftarrow n + 1$;
    end
end
```



(a) Original depth image (b) Depth denoised image (c) Original IR image (d) IR_{roi} is the result of point-wise multiplication of the denoised depth image matrix and the IR matrix

(e) BW_{roi} is IR_{roi} after binarization using adaptive threshold (f) BW_{roi_close} is BW_{roi} after morphological closing operations (g) ROI image (h) Original RGB image

Fig. 8. An example of RGB, depth and IR images captured simultaneously. The intermediate results explained in Algorithm 2 are also presented here.

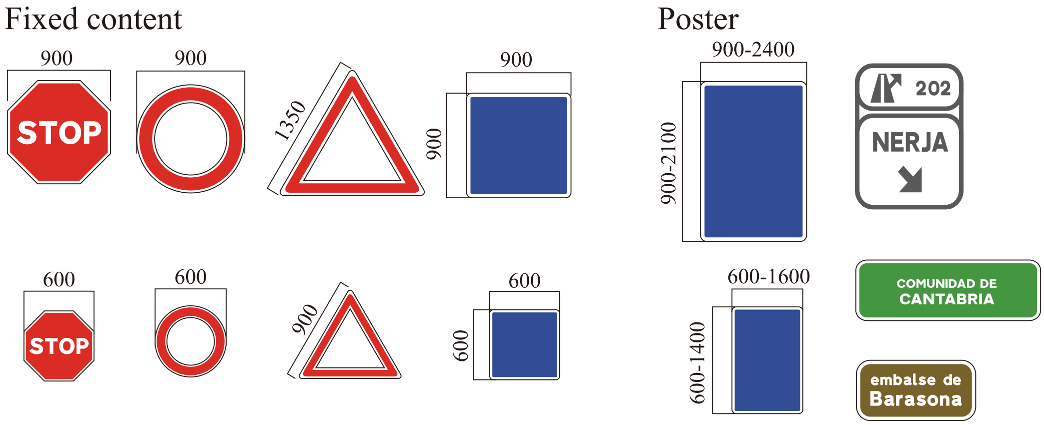


Fig. 9. Categories of traffic signs in Spain (units in mm).

3.3.2. *Region of Interest(ROI) reduction and traffic sign segmentation*

In this step, we took advantage of the unique properties of traffic signs to reduce the ROI area. There are different traffic sign regulations in different countries. As shown in Fig. 9, vertical signs in Spain are categorized as triangular (warn of danger, fix, fixed content), round or regular octagon (prohibit or require, fixed content), square or rectangular (inform or guide, fixed content or poster) (del Estado (2014)).

We applied both color-based and shape-based methods to the area in the red bounding box for traffic sign segmentation. The main workflow is demonstrated in Fig. 10. From Algorithm 2, we obtained ROI on IR images. For each ROI region, we located it on the RGB image and got the RGB information of the ROI area. Following the conversion of the red, green, and blue values into hue, saturation, and value (HSV) values, the main tone of ROI can be determined by calculating the histogram of Hue values in the ROI area. According to the different colors, we can divide traffic signs into three categories:

1. **Red.** As can be seen from Fig. 10, the size and shape of the red traffic sign are fixed, including regular octagon (Stop Sign), round and triangular. We applied an automatic method for stop sign segmentation introduced by Han et al. (2021). After binarization and noise removal on the Hue image, we extracted the edge of the region, and then determined the shape by calculating the roundness property of the region (See Eq. 10).

$$Roundness = \frac{4\pi \times area}{perimeter^2}\begin{cases} = [0.9, 1] \rightarrow Round \\ = [0.55, 0.65] \rightarrow Triangular \end{cases} \tag{10}$$

2. **Blue, Green, Brown.** It can be seen from Fig. 10 that traffic signs in these color are square or rectangular. In general, the procedure can be performed using three steps: contour detection, edge estimation, and traffic sign segmentation. The detailed steps are summarized in Algorithm 3 and an example is given in Fig. 11.
3. **Others.** Other colors will be excluded from the ROI regions.

Algorithm 3. Traffic sign segmentation

2. **Blue, Green, Brown.** It can be seen from Figure 10 that traffic signs in these color are square or rectangular. In general, the procedure can be performed using three steps: contour detection, edge estimation, and traffic sign segmentation. The detailed steps are summarized in Algorithm 3 and an example is given in Figure 11.
3. **Others.** Other colors will be excluded from the ROI regions.

```
Data: $ROI_H$ indicates the Hue value of the ROI region. $ROI_{IR}$ is
      IR value of the ROI area.
Result: $i = [i_1, i_2, i_3, i_4]$ is the four corners points of rectangle.
/* Functions are introduced in Algorithm 2.                        */
$ROI_{HIR} \leftarrow ROI_H$.Multiply($ROI_{IR}$);
$ROI_{BW} \leftarrow$ AdaptiveThreshold($ROI_{HIR}$);
$kernel[0:5, 0:5] \leftarrow 1$;
$ROI_{close} \leftarrow (ROI_{BW} \oplus kernel) \ominus kernel$;
/* Functions Fill fills holes in the binary region                 */
$ROI_{fill} \leftarrow$ Fill($ROI_{close}$);
/* Functions Boundary traces the region boundaries in the
   binary region                                                   */
$ROI_B \leftarrow$ Boundary($ROI_{fill}$);
/* Functions RANSAC estimates edges of the boundaries              */
$l \leftarrow$ RANSAC($ROI_B$);
if In $l$, two sets of parallel lines can be detected then
    $i \leftarrow$intersection of two sets of parallel lines;
end
```

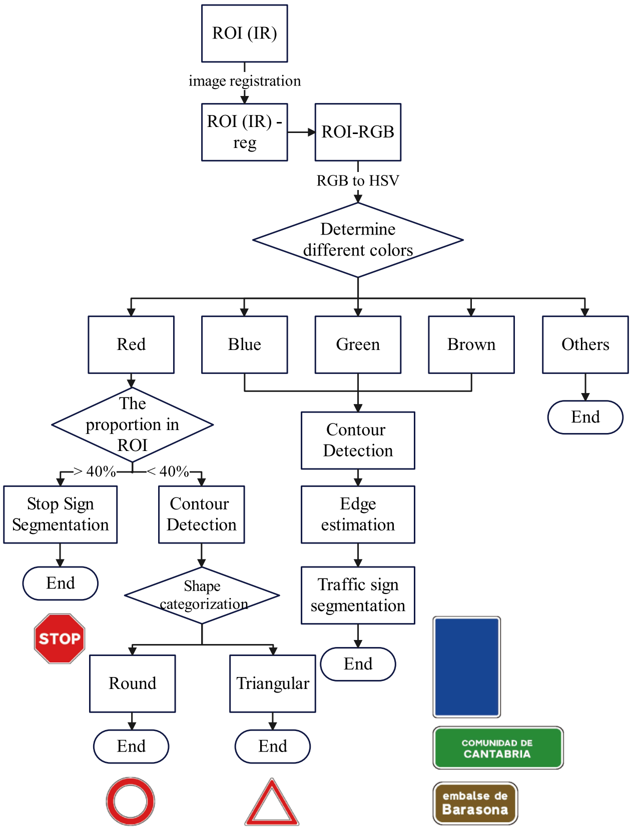


Fig. 10. Workflow of traffic sign segmentation.

3.4. Step 3: Distance calculation

Having segmented the traffic sign, we located the corresponding area on the depth map and determine the most frequent value based on statistical analysis (Fig. 5). We took the peak value of the histogram of the depth map in the traffic sign area as the distance between the camera and the sign. For example, in Fig. 11, the most frequent depth value is 5, 922*mm*.

4. Evaluation and discussion

4.1. Evaluation

In the study, hardware and software specifications are summarized in Table 2.

Some examples are set out in Fig. 12. Table 3 provides a quantitative analysis of this measurement. The ground truth for the entire dataset is determined manually. Closer inspection of Table 3 shows satisfactory results in terms of accuracy. Most false positives (commissions) occurred due to highly reflective logos and ground signs. There were many missed detections (omissions), because traffic signs parallel to the roadside are generally difficult to detect. It also accounts for why all the stop signs (regular octagon) were detected, since all of them were facing the camera.

In addition, we paid attention to the reliability of the distances measured by the depth images. While we used the most frequent depth value as the distance, we also calculated the average depth within the segmented area. Therefore, the coefficient of variation c_v estimated by the following equation:

$$c_v = \frac{\sigma}{\mu} \tag{11}$$

where σ is the standard deviation, μ is the mean value (Everitt and

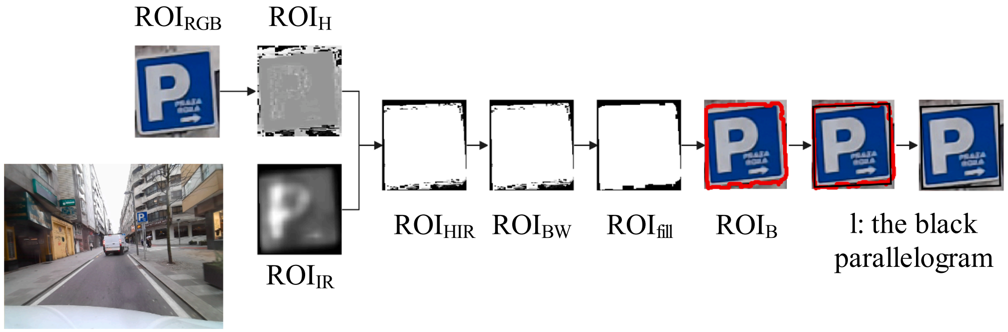


Fig. 11. An example of Algorithm 3.

Table 2
Experimental platform configuration

Processor	Intel(R) Core(TM) i7-10870H CPU @ 2.20 GHz 2.21 GHz
RAM	16.0 GB
GPU	Tesla P100 16G (Deep learning Process)
Coding Language	Python + Matlab

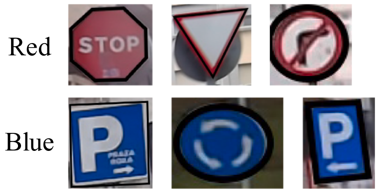


Fig. 12. Example of traffic signs segmentation with different colors and shapes. There are no brown and green traffic signs in the Santiago dataset.

Skrondal (2010)). c_v is a dimensionless number which measures the dispersion of a probability distribution. It can be used as an indicator of reliability since it assesses the stability of data. A low c_v value therefore indicates a more reliable measurement (Shechtman et al. (2013)). In this experiment, c_v averaged 1.1%. This proves the reliability of the distance value.

4.2. Comparison to deep learning methods

In the past ten years, some classic deep learning object detection algorithms such as Faster RCNN (Ren et al. (2015)) and YOLOX (Ge et al. (2021)) have been applied to the detection task of traffic signs. Arcos-García et al. (2018) evaluated the performance of different deep learning networks for traffic sign detection. To demonstrate the advantages of our method, we used YOLOX, a well-known object detection network, to test on our dataset by training on the open-source BDD100K dataset (Chen et al. (2018)). Fig. 13 provides the comparison of YOLOX and other state-of-the-art object detectors, which shows that YOLOX has a simpler design but better performance.

Table 3
Comparison of accuracy, precision and recall of traffic signs with different shapes on the Santiago dataset. The numbers of images in different categories are also provided in the table.

	Rectangle	Round	Triangle	Octagon	*SUM*
Ground Truth	3454	2024	1686	325	7489
True Positive	3008	1433	1473	325	6239
False Alarms	15	37	0	0	52
Missed Detections	461	628	213	0	1302
Precision(%)	86.71	69.52	87.36	100	82.73
Recall(%)	99.50	97.48	100	100	99.17
Accuracy(%)	**86.33**	68.30	**87.37**	**100**	82.16

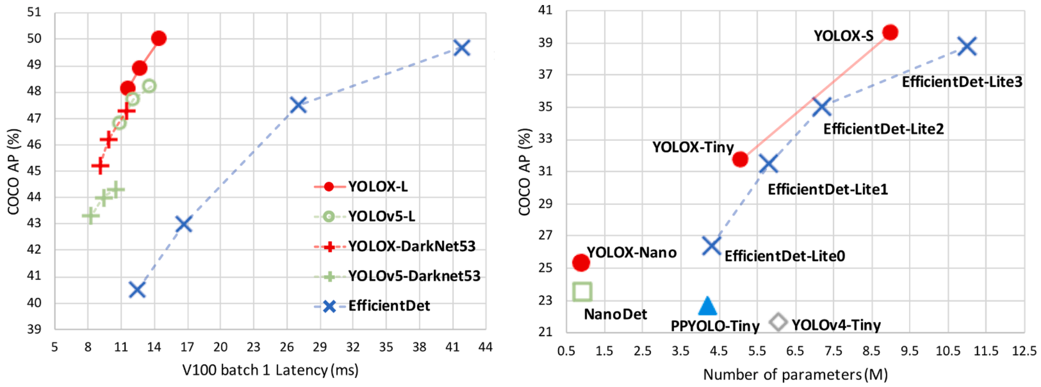
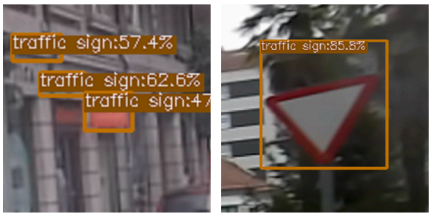


Fig. 13. Trade-off between speed and accuracy of accurate models (left) and size-accuracy curves for lite models on mobiles (right) for YOLOX and other advanced object detectors (Ge et al. (2021)).

Table 4
The evaluation of YOLOX-s on BDD100K dataset

Model	Backbone	Size	AP(%)	AP_{50}(%)	AP_{75}(%)	AP_S(%)	AP_M(%)	AP_L(%)
YOLOX-s	Modified CSP v5	640	37.6	69.6	35.3	29.8	54.8	72.7



(a) False alarm (b) Wrong location of the bounding box

Fig. 14. Examples of the traffic sign detection problems using YOLOX-s model.

4.2.1. *Dataset*

BDD100K dataset is a diverse driving Dataset (Chen et al. (2018)) contains 100 k images of 1280 × 720 pixels, including 70 k training images, 10 k validation images and 20 k testing images.

4.2.2. *Training and testing*

We adopt the SGD optimization algorithm with a momentum of 0.9 to optimize our model. The batch size is set to be 32, and total training epochs is 300. The initial learning rate is set to 0.001 in the front 100 k iterations and divided by ten from 150 to 200 k.

It took us 6 days and 11 h to train the model. It is worth mentioning that we used the small YOLOX-s model, and we only trained one traffic sign class. The evaluation on the BDD100K dataset is summarized in Table 4.

There were obvious false alarms and missed detections when we applied the traffic sign model trained on the BDD100K data set to our own data set. In addition, the locations of the bounding box sometimes were wrong. Some examples are shown in Fig. 14. Besides, no traffic sign was detection by YOLOX-s in Fig. 8h.

4.2.3. *Comparison*

In order to compare the performance of our algorithm and YOLOX, we randomly selected 1000 images from the Santiago dataset and calculated the accuracy, precision and recall. The comparison are shown in Table 5.

4.2.4. *Discussion*

From Table 5, we compared our method with deep learning methods:

- Deep learning is a data-driven approach, so the accuracy depends greatly on the dataset. Moreover, traffic signs that do not exist in the training dataset are not detectable. In the experiments we used public datasets for training and tested on our own datasets. Certainly, the accuracy would improve if we trained with our own dataset, but it would take a long time to accomplish.
- Although inference using deep learning is faster than our algorithm, deep learning needs to spend a lot of computing resources to train the model. Our method does not require pre-training.
- Due to hardware constraints, usually the training dataset needs to be down-scaled (Lu et al. (2018)), while our algorithm processes high-resolution images directly.
- Currently, the traffic sign class provided by the public datasets is only labeled for object detection. Instead, we perform instance segmentation, a more involved process than object detection.
- Deep learning algorithms are more sensitive than ours, however they produce many false positives as a consequence.

4.3. *Limitations*

We identify three major limitations of our method as follows:

- When de-noising, we use two consecutive depth images which assume that the position where two frames were captured does not change. In reality, the car is moving, therefore the positions of the objects in the two frames are also changing. This results in the size of the traffic sign on the depth image after denoising being smaller than the actual one. This test was conducted with a vehicle speed of about 10–20 km/h and a capture frame rate of 15 FPS. In this case, we can compensate for the loss of denoising by enlarging the ROI area. Meanwhile, if the frame rate is 30 frames per second, the corresponding vehicle speed is limited to 30–40 km/h.
- As discussed in Section 2.1, our method did not take into account the case where distance between the traffic sign and the camera exceed 16.2 meters so as to avoid the phase wrapping problem.
- The precision is affected by camera position, and the sensitivity of the side-facing traffic sign is relatively low.

5. Conclusion

Mobile Mapping Systems are an essential component of development of digital twins, intelligent transportation systems, and smart cities. We presented a novel low-cost mobile mapping solution for traffic sign segmentation using Azure Kinect. A comparison was made with deep learning techniques. Our results demonstrate that our method is both reliable and effective for segmenting traffic signs. This study provides a new understanding of the outdoor applications of ToF cameras, which has a number of implications for future practice. There are several sensors commonly used in MMS. The monocular camera has the disadvantage of scale uncertainty, and depth information can only be retrieved through motion; the binocular camera is too computationally intensive and not very reliable; and LiDAR is cost-prohibitive. We proposed a low-cost solution for improving classical MMS, in particular using the depth camera provided by Azure Kinect to directly obtain the distance information. Further study will focus on the following aspects:

1. Exploring the possibility of applying deep learning methods for traffic sign recognition only in the ROI regions. It will not only improves efficiency of segmentation, but also increases accuracy and robustness.
2. There are also motion sensors provided in the Azure Kinect, which were not used in the study. In addition to this, Azure Kinect does not

Table 5
Comparison of our algorithm and YOLOX deep learning algorithm.

	Presion(%)	Recall(%)	Accuracy(%)	Speed
Our method	82.73	**99.17**	**82.16**	15FPS
YOLOX	93.22	78.94	74.66	~50FPS(GPU) ~30FPS(CPU)

interact with other sensors on the Van. Future research will enable Azure Kinect to play a more significant role in mobile mapping. The distance information provided by Azure Kinect can also be used to position the traffic sign on the trajectory map and enhance the result of mobile mapping.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 860370.

This work has also received funding from the European Regional Development Fund: from the Xunta de Galicia-Consellería de Cultura, Educación e Ordenación Universitaria Accreditation 2019–2022 ED431G-2019/04 and Reference Competitive Group Accreditation 2021–2024, GRC2021/48.

This work has been supported by Gobierno de España: Ministerio de Ciencia, Innovación y Universidades through the Grant PID2019-108816RB-I00 funded by MCIN/AEI/ 10.13039/501100011033.

References

Arcos-García, Á., Alvarez-Garcia, J.A., Soria-Morillo, L.M., 2018. Evaluation of deep neural networks for traffic sign detection systems. Neurocomputing 316, 332–344.

del Estado, B.O., 2014. Norma 8.1-ic señalización vertical, 2014. URL: http://www.carreteros.org/normativa/s_vertical/8_1ic_2014/indice.htm.

Brown, D., 1971. Close-range camera calibration, photogrammetric engineering. Eng. Remote Sens. 37, 855–866.

Cavegn, S., Haala, N., 2016. Image-based mobile mapping for 3d urban data capture. Photogram. Eng. Remote Sens. 82, 925–933.

Cavegn, S., Haala, N., Nebiker, S., Rothermel, M., Zw'lfer, T., 2015. Evaluation of matching strategies for image-based mobile mapping., ISPRS Annals of Photogrammetry, Remote Sensing & Spatial. Inf. Sci. 2.

Chen, X.W.W.X.Y., Darrell, F.L.V.M.T., Yu, F., Chen, H., 2018. Bdd100k: A diverse driving dataset for heterogeneous multitask learning, p. 1805.04687 arXiv preprint arXiv.

De Cubber, G., Doroftei, D., Sahli, H., Baudoin, Y., 2011. Outdoor terrain traversability analysis for robot navigation using a time-of-flight camera. In: Proc. RGB-D Workshop on 3D Perception in Robotics, Vasteras, Sweden, Citeseer, 2011.

Elfiky, N.M., Akbar, S.A., Sun, J., Park, J., Kak, A., 2015. Automation of dormant pruning in specialty crop production: An adaptive framework for automatic reconstruction and modeling of apple trees. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 65–73.

Ellahyani, A., El Jaafari, I., Charfi, S., 2021. Traffic sign detection for intelligent transportation systems: A survey. In: E3S Web of Conferences, volume 229, EDP Sciences, p. 01006.

ElRafey, A., Wojtusiak, J., 2017. Recent advances in scaling-down sampling methods in machine learning. Wiley Interdiscip. Rev.: Comput. Stat. 9, e1414.

Everitt, B., Skrondal, A., 2010. The Cambridge Dictionary of Statistics. Finance professional collection. Cambridge University Press. URL: https://books.google.es/books?id=nmu5zQEACAAJ.

Fu, L., Gao, F., Wu, J., Li, R., Karkee, M., Zhang, Q., 2020. Application of consumer rgb-d cameras for fruit detection and localization in field: A critical review. Comput. Electron. Agric. 177, 105687.

Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Marín-Jiménez, M.J., 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recogn. 47, 2280–2292.

Ge, Z., Liu, S., Wang, F., Li, Z., Sun, J., 2021. Yolox: Exceeding yolo series in 2021, arXiv preprint arXiv:2107.08430.

Han, Y., Liu, Y., Paz, D., Christensen, H., 2021. Auto-calibration method using stop signs for urban autonomous driving applications. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). IEEE, pp. 13179–13185.

Hansard, M., Lee, S., Choi, O., Horaud, R.P., 2012. Time-of-flight cameras: principles, methods and applications. Springer Science & Business Media.

Hansard, M., Lee, S., Choi, O., Horaud, R., 2013. Disambiguation of time-of-flight data. Time-of-Flight Cameras, Springer 29–43.

Jaakkola, A., Hyyppä, J., Hyyppä, H., Kukko, A., 2008. Retrieval algorithms for road surface modelling using laser-based mobile mapping. Sensors 8, 5238–5249.

Kettelgerdes, M., Böhm, L., Elger, G., 2021. Correlating intrinsic parameters and sharpness for condition monitoring of automotive imaging sensors. In: 2021 5th International Conference on System Reliability and Safety (ICSRS). IEEE, pp. 298–306.

Lindner, M., Schiller, I., Kolb, A., Koch, R., 2010. Time-of-flight sensor calibration for accurate range sensing. Comput. Vis. Image Underst. 114, 1318–1328.

Lu, Y., Lu, J., Zhang, S., Hall, P., 2018. Traffic signal detection and classification in street views using an attention model. Comput. Visual Media 4, 253–266.

Microsoft, Azure kinect dk hardware specifications, 2022. URL: https://docs.microsoft.com/en-us/azure/Kinect-dk/hardware-specification, date accessed: 04-04-2022.

Nebiker, S., Meyer, J., Blaser, S., Ammann, M., Rhyner, S., 2021. Outdoor mobile mapping and ai-based 3d object detection with low-cost rgb-d cameras: The use case of on-street parking statistics. Remote Sensing 13, 3099.

OpenCV, Camera calibration, 2022. URL: docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html.

Paparoditis, N., Papelard, J.-P., Cannelle, B., Devaux, A., Soheilian, B., David, N., Houzay, E., 2012. Stereopolis ii: A multi-purpose and multi-sensor 3d mobile mapping system for street visualisation and 3d metrology. Revue française de photogrammétrie et de télédétection 200, 69–79.

Peláez, L.P., Recalde, M.E.V., Muñóz, E.D.M., Larrauri, J.M., Rastelli, J.M.P., Druml, N., Hillbrand, B., 2019. Car parking assistance based on time-or-flight camera. 2019 IEEE Intelligent Vehicles Symposium (IV), IEEE 2019, 1753–1759.

Puente, I., González-Jorge, H., Martínez-Sánchez, J., Arias, P., 2013. Review of mobile mapping and surveying technologies. Measurement 46, 2127–2145.

I. Puentea, H. González-Jorgea, P. Ariasa, J. Armestoa, 2011. Land-based mobile laser scanning systems: a review, International archives of the photogrammetry, remote sensing and spatial information sciences 38.

Rafael, R.E.W., Gonzalez, C., 1992. Digital image processing. Addison-Wesley.

Rashdi, R., Martínez-Sánchez, J., Arias, P., Qiu, Z., 2022. Scanning technologies to building information modelling: A review. Infrastructures 7, 49.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Adv. Neural Inform. Process. Syst. 28.

Sairam, N., Nagarajan, S., Ornitz, S., 2016. Development of mobile mapping system for 3d road asset inventory. Sensors 16, 367.

Scipy, Scipy sparse bsr_matrix, 2022. URL: docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.bsr_matrix.html.

Serna, A., Marcotegui, B., 2013. Urban accessibility diagnosis from mobile laser scanning data. ISPRS J. Photogram. Remote Sens. 84, 23–32.

Shechtman, O., 2013. The coefficient of variation as an index of measurement reliability. In: Methods of clinical epidemiology, Springer, 2013, pp. 39–49.

Steinbaeck, J., Druml, N., Tengg, A., Steger, C., Hillbrand, B., 2018. Time-of-flight cameras for parking assistance: a feasibility study. In: 2018 12th International Conference on Advanced Semiconductor Devices and Microsystems (ASDAM). IEEE, pp. 1–4.

Tölgyessy, M., Dekan, M., Chovanec, L., Hubinskỳ, P., 2021. Evaluation of the azure kinect and its comparison to kinect v1 and kinect v2. Sensors 21, 413.

Wali, S.B., Abdullah, M.A., Hannan, M.A., Hussain, A., Samad, S.A., Ker, P.J., Mansor, M. B., 2019. Vision-based traffic sign detection and recognition systems: Current trends and challenges. Sensors 19, 2093.

Zou, Z., Shi, Z., Guo, Y., Ye, J., 2019. Object detection in 20 years: A survey arXiv preprint arXiv:1905.05055.