EasyChair Preprint

№ 1448

# Pre-Robotic Navigation Identification of Pedestrian Crossings & Their Orientations

Ahmed Farid and Takafumi Matsumaru



September 1, 2019

# Pre-Robotic Navigation Identification of Pedestrian Crossings & Their Orientations

Ahmed Farid and Takafumi Matsumaru

**Abstract** This paper describes an off-line (i.e. pre-navigation) methodology for machines/robots to identify zebra crossings and their respective orientations within pedestrian environments, for the purpose of identifying street crossing ability. Not knowing crossing ability beforehand can prevent motion trajectories from being accurately planned pre-navigation. As such, we propose a methodology that sources information from internet 2D maps to identify the locations of pedestrian zebra crossings. This information is comprised of road networks and satellite imagery of street intersections, from which the locations and orientations of zebra crossings can be identified by means of trained neural networks and proposed verification algorithms. The methodology demonstrated good capability in detecting and mapping zebra crossings' locations, while also showing good results in verifying them against falsely detected objects in satellite imagery. Orientation estimation of zebra crossings, using a proposed line-scanning algorithm, was found to be within an error range of 4 degrees on a limited test set.

## 1 Introduction

The role of mobile robotics is bound to increase in the future across many fields, such as human assistance, package delivery, and surveillance. One important setting for such robots' operation is pedestrian environments. The annual Tsukuba Challenge held in Japan [1] addresses problems and tasks in such pedestrian settings using mobile robots. In such context, one important task is that of street crossing; robots must be able to recognize street crossings [2, 3] and to determine the safety of crossing [4] during on-line operation.

We argue for the need to identify the locations of street crossings prior to navigation (i.e. prior to on-line operation). Consider the two cases in Fig.1; assuming

Both authors are with Waseda University, Graduate School of Information, Production & Systems, 2-7 Kitakyushu, Fukuoka, Japan, e-mail: ahmed.farid@asagi.waseda.jp, matsumaru@waseda.jp

a mobile robot with a controller logic for reducing the distance between the robot and a destination, in both cases (a) and (b) the options for motion would increase that distance. Case (a) can be solved if the controller logic was modified, so that the robot can use explicitly marked zebra crossings at the same street intersection to eventually move towards the destination. This would also eventually solve case (b), but no prior-knowledge would mean there is a probability that the robot would take a longer route. What we wish to highlight here is that prior-knowledge of the environment affects both the controller logic of a robot and time cost to reach a destination. Task efficiency and reducing energy/time costs are especially important for the applications of package delivery and the guidance of visually impaired people.

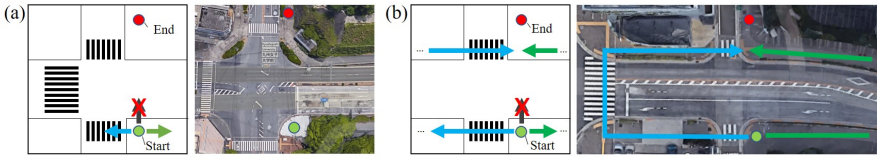


**Fig. 1** Two cases with their real-life equivalents in which there were no direct street crossing to the goal. This is to demonstrate the problems of controller logic and time cost to reach a given destination. (Satellite images are from Google Maps©)

In this paper, we describe a methodology to identify zebra crossings pre-navigation to benefit map construction and path planning purposes. Specifically, we propose a method to identify a zebra crossing's location and orientation in the world frame (i.e. GPS-wise). We are interested in the paradigm of using 2D internet maps (e.g. Google Maps, OpenStreet Maps) instead of on-line map recording techniques such simultaneous localization and mapping (i.e., SLAM) and structure from mapping (i.e., SfM) algorithms. In this methodology, we utilize satellite imagery of street intersections for the reason that not all 2D digital maps have street crossing information. For the identification process, we trained an object detector neural network on a self-prepared dataset of zebra crossings. We will introduce two methods for verifying the correctness of a zebra crossing detection, followed by an algorithm to estimate the orientation. We estimate orientation as a means of identifying both possible motion trajectories and explicit linkages between one city block to another.

## 2 Related Works

For robots to be able to create path plans and navigate in such environments, a topological understanding, or a map, is required. One method suggested by the literature is by using on-line map recording techniques such as SLAM [5, 6] and SfM [7, 8]. Kummerle et al. [5] demonstrated a mobile robot utilizing a SLAM-based 3D map for navigation in a densely populated area. Although some implementations can detect the presence and location of street crossings, time and effort are needed for the

on-line recording trips made by robots. Additionally, such robot-recorded maps are not widely available (i.e. no global scale coverage), which means that the preparation of such maps demands time and effort to provide global map coverage.

To overcome the global map availability problem, research has been made into navigation using 2D internet maps. Hentschel and Wagner [9] utilized OpenStreet Maps to implement path planning and navigation by an autonomous vehicle. Their system, however, depended on the accuracy of map encoded way-points. Irie, Sugiyama and Tomono [10] implemented localization by applying edge matching between a 2D map and a real-world RGB camera view. In a similar manner, other research [11, 12] implemented localization but by using 3D point clouds instead. Hosoda et al. [13] devised a technique to generate a simplified map structure by using an input Google Map image. In the aforementioned works, although a mobile robot was utilized as a testing platform, the locations of street crossings were not explicitly considered; a general practical problem with 2D internet maps is that explicit data on street crossings is not always available. This problem affected our own previous work [14, 15]; we developed an off-line path planning framework to attempt addressing sidewalk trajectories and street-crossing maneuvers, but the accuracy of some test cases was influenced by missing street crossing information in the utilized maps.

As such, we decided to address the identification of the possible locations for street crossings to improve path planning accuracy and overcome the prior-knowledge challenges previously mentioned in section 1. The identification process was designed to utilize 2D internet maps for their wide availability and for not requiring on-line map recording by a robot beforehand. Owing to the fact that 2D maps do not always have such street crossing information in practice, locations of street intersections were utilized to directly obtain satellite imagery of such intersections, from which zebra crossings can be visually detected by a trained neural network (i.e. if not visually obstructed by a bridge or overpass). The output of the proposed work is the GPS-wise locations street-crossings, which can be combined with previous works [10, 15, 16] to enhance localization and navigation mobile robots in pedestrian settings.

## 3 Methodology Description

### *3.1 Sourcing Street Maps and Satellite Imagery*

Map retrieval, in principle, requires the knowledge of its GPS coordinate bounds. In our implementation, we used GPS bounds for desired locations to download .OSM map files from OpenStreet Maps. The .OSM file[1] contains descriptions of road networks and geometries of buildings. Using such descriptions, we extracted two important elements: (1) A list of all road intersections occurring in the map, and (2)

[1] Further description of an .OSM file's structure can be found in [9]

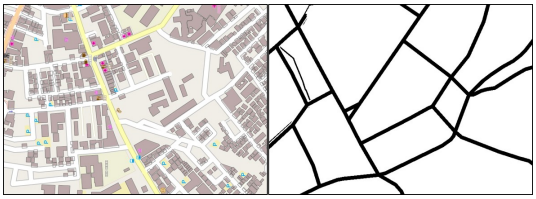

**Fig. 2** An example of an urban map converted to a binarized image form. This was done by using the Maperitive software tool [17].

a graphical representation of the road networks. In our implementation, we utilized the Maperitive software [17] to generate custom-styled maps; in Fig.2, an example stylized map is shown, which would be later utilized in section 3.2.3. The list of intersections, comprised of GPS coordinates of street intersections, were then used to download satellite imagery directly using Google Maps' API. Such imagery are close-up images of street intersections, which would then act as a source for detecting the location of zebra crossings.

## *3.2 Zebra Crossing Detection*

### 3.2.1 Training of an Object Detection Neural Network

As mentioned previously in section 2, information on street crossings is not always present within downloadable maps in practice. Despite maps showing a traffic signal icon as in Fig.3 (left), it is not indicative of exact locations of street crossings. By utilizing satellite imagery, their existence and locations can be identified. To implement this, we trained an object detection neural network to detect zebra crossings within a given image. We prepared a dataset of satellite imagery which includes labels for the locations of zebra crossings. Table 1 provides a summary on the properties of the dataset and the performance of the trained network.

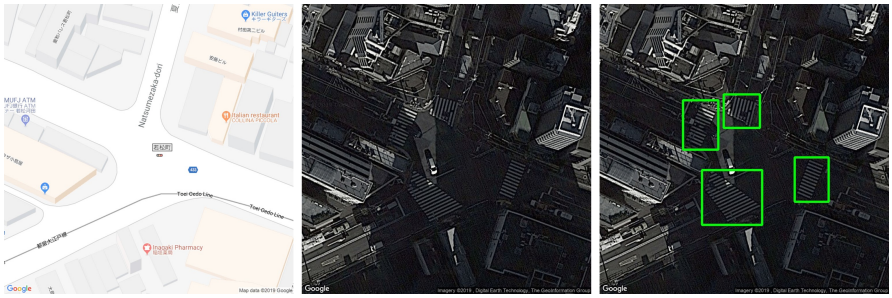

**Fig. 3** Zebra crossing detection at a given intersection. Left is a Google Maps© image of the intersection, showing only a traffic signal. Middle is a satellite image of the intersection. Right shows zebra crossings detected by the trained neural network.

**Table 1** Object Detector Properties

| Property | Value |
|---|---|
| Training set | 115 satellite images (435 crossings) |
| Testing set | 40 satellite images (115 crossings) |
| Image size | 640x640 |
| Augmentation | Rotation, flipping, and random darkening |
| Neural network | Fast Inception V2 |
| Batch size | 1 |
| Learning rate | $2 \times 10^{-6}$ |
| Training steps | 50,000 steps |
| Graphics hardware | Nvidia GTX1080Ti |
| Evaluation accuracy (mAP) | 89.792% |

Object detectors only provide location information within the pixel space of an input image, and not in that of the world-space (i.e. GPS). To obtain GPS-wise locations, a mapping from the local pixel space of a given satellite image to the global GPS space is needed. Given the GPS-wise[2] and pixel-wise bounds of an input image, mapping functions to convert pixel positions to GPS latitudes and longitudes were utilized as shown in equations 1 and 2. Equations 3 and 4 are the inverse of 1 and 2 (i.e. conversion from GPS space to pixel space).

**Table 2** Abbreviations for GPS Coordinates-to-pixel-space functions Eqs.1 and 2

| Property | Value |
|---|---|
| Image width | imW |
| Image height | imH |
| Input longitude | inLng |
| Output longitude | outLng |
| Left longitude of bounding box | leftLng |
| Right longitude of bounding box | rightLng |
| Input latitude | inLat |
| Output latitude | outLat |
| Top latitude of bounding box | topLat |
| Bottom latitude of bounding box | botLat |
| X pixel position in image | X |
| Y pixel position in image | Y |

$$outLng = \frac{X * (rightLng - leftLng)}{imW} + leftLng \tag{1}$$

[2] Using the resolution of an image, the aerial zoom level and the GPS center (i.e. the GPS location of a given street intersection), estimation of the GPS bounds is achievable by means of Mercator projection calculations [18]

$$outLat = \frac{(imH - Y) * (topLat - botLat)}{imH} + botLat \tag{2}$$

$$X = imW * \frac{inLng - leftLng}{rightLng - leftLng} \tag{3}$$

$$Y = imH - \frac{imH * (inLat - botLat)}{topLat - botLat} \tag{4}$$

The trained neural network showed great performance in detecting zebra crossings. However, there were cases where false positives were detected (Fig.4). Due to the critical nature of street crossings in a real-world scenario, very high accuracy is desired. In the next two subsections, we will introduce two output verification techniques and will afterwards evaluate their performance.

### 3.2.2 Classifier Neural Network-based Verification

One idea for verification is to feed the output of the object detector into a classifier neural network; we utilized the fact that the classifier was trained not only on zebra crossings but also on examples of true negatives. In essence, the output of the classifier, based on a confidence threshold, would be binary (e.g. whether a zebra crossing is true or not). An illustration is shown in Fig.4, training and evaluation properties are shown in table 3.

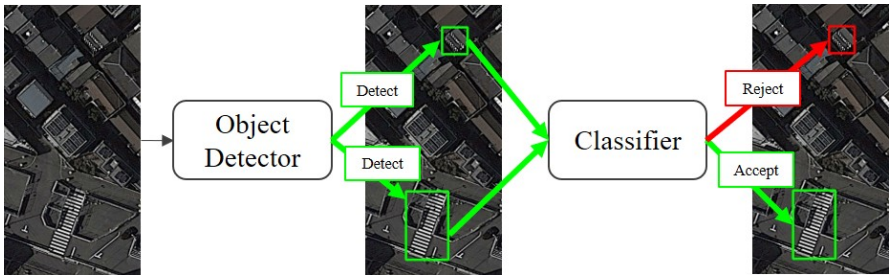


**Fig. 4** Illustration of the classifier neural network based verification approach. Satellite imagery were obtained from Google Maps©.

### 3.2.3 Location-wise Algorithmic Verification

We present an alternative algorithmic method to verify the output of the trained object detector. The reason is that zebra crossings have two real-life properties that can be utilized: (1) they exist on a street, and (2) they exist between two city blocks (i.e. used to cross from one city block to another). Utilizing these properties, we de-

**Table 3** Classifier Properties

| Property | Value |
|---|---|
| Dataset size | 474 positives, 65 negatives |
| Train/Test split | 70/30 |
| Augmentation | Rotation, flipping, and random darkening |
| Neural network | Inception V3 RCNN |
| Batch size | 100 |
| Learning rate | $1 \times 10^{-5}$ |
| Training steps | 15000 steps |
| Graphics hardware | Nvidia RTX Titan |
| Evaluation accuracy | 92% |

signed an algorithm that received the positions of the detected zebra crossings, and then calculated the pixel-wise distances between these positions and surrounding city blocks in a map. The city blocks were obtained by applying contour detection on the binary map image retrieved in section 3.1. To achieve this, the GPS positions of detected zebra crossings are mapped into the pixel-space of the binary map image using equations 3 and 4.

**Algorithm 1** Location-wise algorithm for zebra crossing detection verification

```
for each zebra crossing: z do
    initialize distances array d
    for each city block touching main roads: b do
        distance = get distance between z and b
        if distance <road width then
            d.append(b,distance)
        end if
    end for
    sort d based on distance index
    if d has more than one element then
        block1 = block index of the first element of d
        block2 = block index of the second element of d
        add z to the list of valid zebra crossings
    end if
end for
```

### 3.2.4 Accuracy of Detection Verification Approaches

In this subsection, accuracy of detection verification approaches is presented. As an experiment, a map for an urban location was used as a test example[3]. By scanning

[3] The urban location on which the test was conducted is a random area within the Shinjuku ward in Tokyo, Japan. This area is bound between the GPS coordinates (i.e. latitude and longitude) of [35.6994726204,139.718178435] and [35.6975273796,139.716821565].

through this map at each street intersection, the trained detector made a total of 82 detections at confidence threshold 0.7. Within those 82 detections, there were 76 true positives and 6 true negatives. In Fig.5, the precision and recall of both the classifier (at thresholds ranging between 0.1 and 0.9) and algorithmic-based approaches are shown.

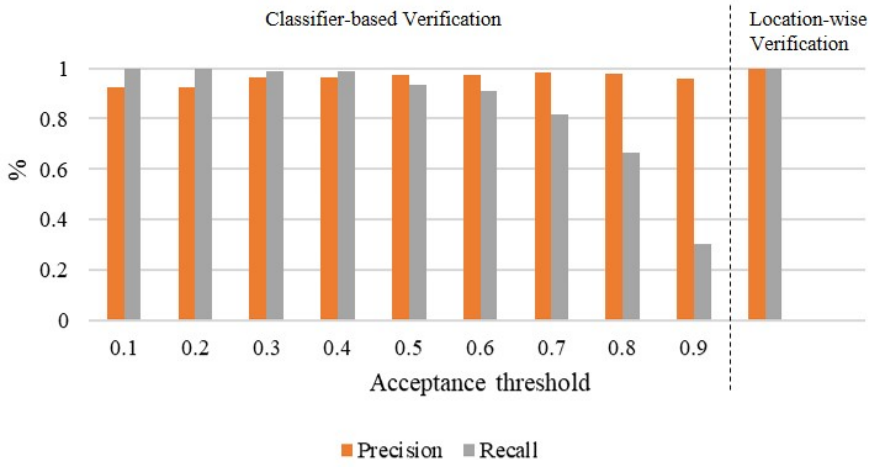


**Fig. 5** Precision and recall results for the classifier-based (left) and location-wise algorithmic (right) verification approaches.

Although the classifier achieved relatively high accuracy, the algorithmic approach was better. This is owing to the positional properties of zebra crossings that were utilized by the algorithmic approach. On the other hand, the classifier approach largely depended on its trained dataset for its ability to differentiate between true zebra crossings and false ones. As a side note, the high precision in the graph is indicative of the good ability to find true positives by the trained object detector. Our main motivation from the comparison between classifier-based and algorithmic-based approaches is to demonstrate that not all problems should be solved solely by neural networks.

## *3.3 Orientation Estimation*

### 3.3.1 Estimation Approach

A technique for estimating orientations was developed to identify motion trajectories, which would be useful in linking city blocks to one another in a map construction application. As shown in Fig.6, orientation estimation is motivated by presence of street crossings that are diagonal in trajectory (i.e. with respect to the street). Such

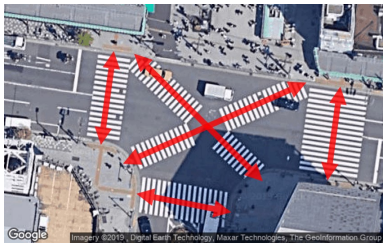


**Fig. 6** An example satellite image from Google Maps© showing that some streets have diagonal street crossings.

street crossings, if identified prior to navigation, can be utilized in a path planning process to provide shorter paths.

The main idea of the proposed technique is to attempt to fit a line between the zebra stripes, then obtaining the tangent angle of that line to estimate the orientation of the street crossing. In our proposed technique, the first step was to convert the input zebra crossing image to grayscale, then binarizing it; instead of using a simple threshold function, we utilized K-means clustering (k=2) to binarize. This was done because of lighting conditions on a given zebra crossing; owing to the real-world aspect of satellite images captured during day time, some zebra crossings can be partially or fully shaded by a nearby building. As a result, we used K-means clustering to implicitly determine the binarization threshold by dividing the bright colors (e.g. the white stripes) from the dark street color. Next, a line was drawn across the binarized image and was rotated in a range from 0°to 179°, assuming 1°rotation resolution. Per each degree rotation, two properties were measured: the number of zebra stripes intersected (S) and the total intersected length (L). As such, we form a 180x2 matrix of these measurements as in equation 5. An example of the line scan approach is shown in Fig.7.

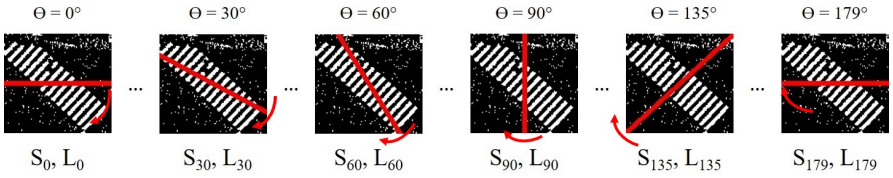


**Fig. 7** Example of the rotating line scan approach for orientation estimation. Starting from $\theta = 0°$, a line is rotated until it reaches $\theta = 179°$ (assuming 1° resolution for clockwise rotation). Per each angle rotation, the number of zebra stripes intersected (S) and the total intersected length (L) are measured.

$$Score = \begin{bmatrix} S_0, L_0 \\ S_1, L_1 \\ \vdots \\ S_{179}, L_{179} \end{bmatrix} \tag{5}$$

Within this matrix, We searched for a minimum combination; we prioritized the least number of stripes intersected, then followed by the intersected length. Upon finding the minimum, the row at which this minimum was found would denote the angle of the fitted line, from which the tangent angle is computed (i.e. fitted line angle plus 90°). This tangent angle was considered the orientation angle of the zebra street crossing.

### 3.3.2 Exemplary Results for Orientation Estimation

Orientation estimation for five cases of zebra crossings are shown in Fig.8. The five cases have different properties and thus were used to demonstrate the estimation technique under different circumstances: case (1) is partially faded, case (2) is completely shaded, and case (3) is partially shaded. Cases (4) and (5) were artificially shaded to experiment with different shading ratios and patterns. In all of the five cases, the approach managed to estimate the general direction of the input zebra crossing, with the most error difference being 4 degrees. Although such error gap is relatively small, more work is needed to improve upon the accuracy of the algorithm.

## 4 Conclusion

In this paper, a methodology for identifying zebra crossings prior to robotic navigation was demonstrated. Given the relatively high accuracy, we believe that the next step is to integrate the presented methodology into other works (e.g. robot localization/navigation). However, we find room for further improvement of the orientation estimation algorithm to reduce the error margin further. Additionally, we envision an application in which zebra crossing locations can be identified by an artificial agent, then stored in an internet database (e.g. OpenStreet Maps). This way, not only robots can benefit from understanding street crossing ability prior to navigation, but also humans with visual impairments can be guided by either a smartphone application or even a mobile guidance robot that utilize internet maps.

**Acknowledgements** This study was supported by Waseda University Grant for Special Research Projects Number 2018A-047. The study was also partially supported by Waseda University - Graduate Program for Embodiment Informatics Research Grant. For both grants, we wish to express our sincere gratitude.

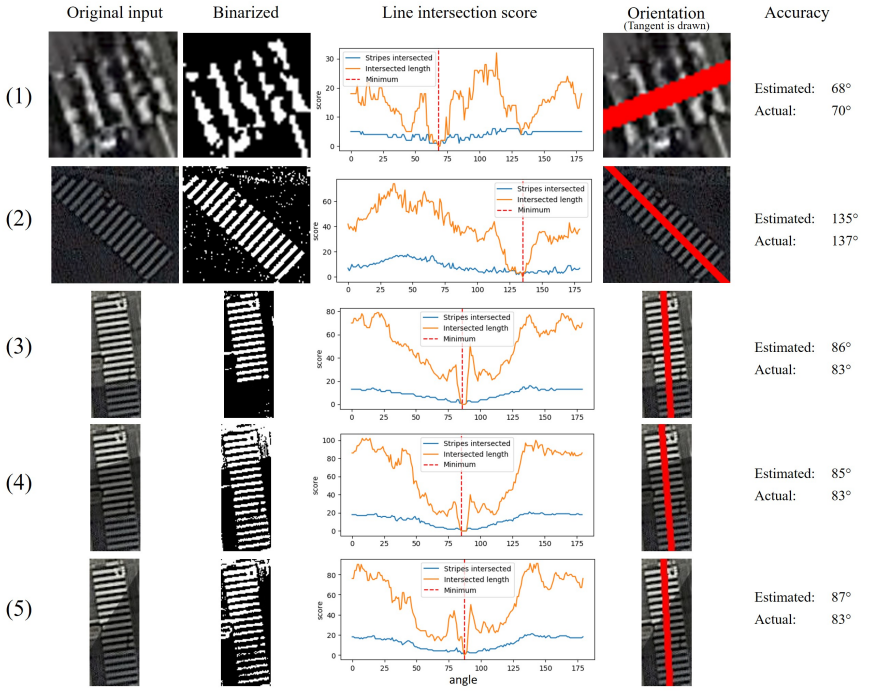


**Fig. 8** Exemplary results on orientation estimation for detected zebra crossings. Conditions vary based on real-life fading of the stripes and lighting conditions. Original input zebra crossings were originally from satellite imagery from Google Maps©.

## References


1. Tsukuba Challenge, Web page http://www.tsukubachallenge.jp/ [Accessed: March, 2019]
2. A. Chand and S. Yuta, "Navigation strategy and path planning for autonomous road crossing by outdoor mobile robots", 2011 15th International Conference on Advanced Robotics (ICAR), 2011.
3. A. Chand and S. Yuta, "Road-Crossing Landmarks Detection by Outdoor Mobile Robots", Journal of Robotics and Mechatronics, vol. 22, no. 6, pp. 708-717, 2010.
4. N. Radwan, W. Winterhalter, C. Dornhege and W. Burgard, "Why did the robot cross the road? Learning from multi-modal sensor data for autonomous road crossing", 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2017.
5. R. Kummerle, M. Ruhnke, B. Steder, C. Stachniss and W. Burgard, "Autonomous Robot Navigation in Highly Populated Pedestrian Zones", Journal of Field Robotics, vol. 32, no. 4, pp. 565-589, 2015.
6. A. Carballo, S. Seiya, J. Lambert, H. Darweesh, P. Narksri, L. Morales, N. Akai, E. Takeuchi and K. Takeda, "End-to-End Autonomous Mobile Robot Navigation with Model-Based System Support", Journal of Robotics and Mechatronics, vol. 30, no. 4, pp. 563-583, 2018.
7. S. Song, M. Chandraker and C. Guest, "High Accuracy Monocular SFM and Scale Correction for Autonomous Driving", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 4, pp. 730-743, 2016.
8. M. Suraj, H. Grimmett, L. Platinsky and P. Ondruska, "Predicting trajectories of vehicles using large-scale motion priors", 2018 IEEE Intelligent Vehicles Symposium (IV), 2018.

9. M. Hentschel and B. Wagner, ”Autonomous robot navigation based on OpenStreetMap geodata”, 13th International IEEE Conference on Intelligent Transportation Systems, 2010.
10. K. Irie, M. Sugiyama and M. Tomono, ”A dependence maximization approach towards street map-based localization”, 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2015.
11. C. Landsiedel and D. Wollherr, ”Global localization of 3D point clouds in building outline maps of urban outdoor environments”, International Journal of Intelligent Robotics and Applications, vol. 1, no. 4, pp. 429-441, 2017.
12. G. Floros, B. van der Zander and B. Leibe, ”OpenStreetSLAM: Global vehicle localization using OpenStreetMaps”, 2013 IEEE International Conference on Robotics and Automation, 2013.
13. Y. Hosoda, R. Sawahashi, N. Machinaka, R. Yamazaki, Y. Sadakuni, K. Onda, R. Kusakari, M. Kimba, T. Oishi and Y. Kuroda, ”Robust Road-Following Navigation System with a Simple Map”, Journal of Robotics and Mechatronics, vol. 30, no. 4, pp. 552-562, 2018.
14. A. Farid and T. Matsumaru, ”Path Planning of Sidewalks & Street Crossings in Pedestrian Environments Using 2D Map Visual Inference”, ROMANSY 22 Robot Design, Dynamics and Control, pp. 247-255, 2018.
15. A. Farid and T. Matsumaru, ”Path Planning in Outdoor Pedestrian Settings Using 2D Digital Maps”, Journal of Robotics and Mechatronics, vol. 31, no. 3, pp. 464-473, 2019.
16. T. Suzuki, M. Kitamura, Y. Amano and N. Kubo, ”Autonomous Navigation of a Mobile Robot Based on GNSS/DR Integration in Outdoor Environments”, Journal of Robotics and Mechatronics, vol. 26, no. 2, pp. 214-224, 2014.
17. Maperitive, Web page http://maperitive.net/
18. E. Olson, Autonomous Robotics Laboratory Lecture Notes, ”Map Projections and GPS”, EECS467, University of Michigan, Winter 2014. Retrieved from: https://april.eecs.umich.edu/courses/eecs467_w14/wiki/images/5/5e/Maps_gps.pdf