Journal of Engineering and Applied Sciences 13 (7): 1859-1871, 2018
ISSN: 1816-949X

Development and Performance Analysis of Canny and Hough Transform Based Pole Detection Algorithm as Mobility Aid for Visually Impaired Person on SEE-S (Smart Environment Explorer-Stick)

[1, 2, 3]M. Yusro, [1]K. Ramli, [2]K.M. Hou and [1]D. Sudiana
[1]Department of Electrical Engineering, Universitas Indonesia (UI),
Depok, Indonesia
[2]LIMOS UMR 6158 CNRS, Blaise Pascal University,
Clermont Ferrand, France
[3]Department of Electrical Engineering, Universitas Negeri Jakarta UNJ,
Kampus Rawamangun, Jakarta, Indonesia

Abstract: This study investigated approach to improve the SEE-S (Smart Environment Explorer-Stick) function in detecting obstacle for Visually Impaired Person (VIP). The study proposed enhanced algorithm to detect the presence of poles. This algorithm uses a combination of method of distance calculation and pairing search of vertical lines obtained from optimized Canny detection. The position of the vertical lines was found using Houghlines function. Our field performance evaluation tests indicated that the proposed algorithm works satisfactorily. Two dominant variables that significantly affect the ability of SEE-S in detecting poles were identified. They were the location of the test and the colour of poles. Two pole colours are used, namely black and silver. At indoor the algorithm performance was worse than at outdoor by 0.085 times. The use of silver pole improved the performance of the algorithm by 11.757 times.

Key words: Smart stick, Canny edge, Hough transform, pole detection, algorithm, significantly

INTRODUCTION

Now a days, urban environment is getting complex due to building shapes, roads, pedestrians, poles, transportation modes and others. In the street, we are familiar with Pole-Like Objects (PLOs) standing as traffic signs, lamp posts, street signs and tree trunks which are some important details for roads (Lehtomaki *et al*., 2010; Yokoyama *et al*., 2013). Some studies and development on the algorithm to detect and classify PLOs in urban environment has been done using Mobile Laser Scanning (MLS) system (Yokoyama *et al*., 2013; Serna and Marcotegui, 2014; Cabo *et al*., 2014). The purpose of those studies are managing and mapping some urban facilities. But, pole detection for human, especially Visually Impaired Person (VIP) has not been conducted yet as further concern in order to facilitate them conveniently.

On the other side, to support the activities of the VIP, there are many studies on the designs of navigation and mobility aids (Dakopoulos and Bourbakis, 2010; Praveen and Paily, 2013). A critical issue for VIP and blind people is their ability to walk or move indoor and outdoor independently. Generally, they are able to walk independently along a route they had travelled before or studied along with a guide (Strumillo, 2010). These facts limited their daily activities significantly. On the other hand, they have a strong desire to travel independently without any help, be able to travel to new locations safely and have no worries of getting lost on their way. Mobility enables someone to interact with the outside world (Hakobyan *et al*., 2013). Human autonomous mobility is an ability to reach location B from initial location A without any help from other people.

For a high function of mobility, VIP need the ability to detect obstacles in front of them (Costa *et al*., 2012). The main obstacle often encountered by VIP upon travelling is an obstacle in the form of poles, namely street lights, traffic light poles, street mark poles and trees, which generally stands vertically. Therefore, they need the ability to detect and recognize obstacles in the form of poles with an algorithm developed from image processing system.

The target of this research is an applied algorithm for detecting robust objects, especially, poles. Therefore, VIPs are able to recognize obstacles whether they are pole or not. Furthermore, pole detection capability is an important part in SEE-S which offers a solution in mobility and navigation aids by utilizing modern technology (Yusro *et al*., 2013).

Corresponding Author: M. Yusro, Department of Electrical Engineering, Universitas Indonesia (UI), Depok, Indonesia

This study describes the research and implementation of object detection algorithm to recognize the existence of poles. This algorithm uses a combination method of distance calculation and pairing search between vertical lines obtained by optimized Canny detection. The position of vertical lines is obtained from Houghline transform function where the output is line vector with 2 coordinates, i.e., the starting point and end point of a line. Those line vectors is further formed into pairing lines and then filtered with distance categories between lines. Afterwards, only lines with more than a pair are detected as poles.

Literature review: Several studies/researches have been conducted worldwide for object detection and recognition based on image processing technology. In 2006, a straight line detection algorithm is proposed by Lee *et al*. (2006). The algorithm uses Principal Component Analysis (PCA) by separating row and column edges from edge image using the primitive shapes. PCA will make the algorithm detects straight lines and orientations which is useful for various intensive applications. The algorithm is a simple, efficient and good for an edge image that has a lot of edges. Future work for improving the performance of the algorithm is solving the problem of two or more merged straight lines.

A most popular tool for line detection is the Hough Transform (HT) due to its robustness to noise and missing data. Fernandes and Oliveira (2008) presented an improved voting scheme for the HT that allows a software implementation to achieve real-time performance even on relatively large images. They used an approach on clusters of approximately collinear pixels. The proposed approach produces a much cleaner voting map and makes HT more robust to the detection of spurious lines and also significantly improves the performance of the voting scheme.

Sharma (2010) conducted studies of an object detection by using particle swarm optimization based template matching algorithm. Implementation of this algorithm reduces the time required for object detection than conventional template matching algorithm. The algorithm can detect object in less number of iterations and hence less time and energy than the complexity of conventional template matching. In this study, the feature makes the method capable for real-time implementation.

The other study by Yu *et al*. (2010) has been developed an algorithm to reduce accident at the intersections during day and night. The traffic lights detection algorithm which is applied in a vehicle driver assistance system is created by using an image processing. The traffic light detection system includes three parts: PC, CCD camera and image acquisition card. The traffic lights can be well extracted in the complex environment by using the object extraction in RGB (Red, Green and Blue) space and the verification rule. The algorithm may result the round traffic lights as well as arrow-shaped traffic lights and give traffic signals accurately. Final experiments show that the algorithm has the characteristics of reliability and stability.

Oji (2012) did a research on image processing for objects recognition with full boundary detection by combining Affine Scale Invariant Feature Transform (ASIFT) and a region merging algorithm. In this study, ASIFT is a fully affine invariant algorithm that means features are invariant to six affine parameters namely translation (2 parameters), zoom, rotation and two-camera axis orientations. Investigational results show that the presented method is very efficient and powerful to recognize the object and detect with high accuracy.

Currently, a computer vision-based wayfinding and navigation aid can improve the mobility of VIP to travel safety and independently. Study in 2014 has developed a new framework to detect and recognize stairs, pedestrian crosswalks and traffic signals based on RGB-D (Red, Green, Blue and Depth) images (Wang *et al*., 2015). Since, both stairs and pedestrian crosswalks are featured by a group of parallel lines, they applied HT to extract the concurrent parallel lines based on the RGB channels. The depth channel is employed to recognize pedestrian crosswalks and stairs. Next, the traffic signs of pedestrian crosswalks are recognized. The results of detection and recognition demonstrate the effectiveness and efficiency of framework.

Ushma *et al*. (2014) conducted a research by using digital image processing techniques to detect the object in digital images. Principally to detect an object uses two major steps: image enhancement and edge detection. To detect an object from digital images is a major task in image processing. It has been a prior discussion all over the world to detect and classify the shape based on image processing technology.

Many research on image processing-based focusing on object detection and recognition has been carried out but the specific discussion on pole object detection has not been investigated yet which is useful for VIP mobility.

MATERIALS AND METHODS

Development of improved algorithm for pole detection: Canny method used for edge detection from an image

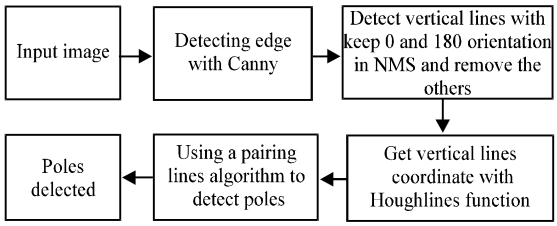

Fig. 1: The pole detection process

input. There is a process called Non-Maxima Suppression (NMS) (Neubeck and Gool, 2006) which compare pixel with their neighbour based orientation pixel or direction pixel (θ). From this step, we keep the 90° orientation only and the others orientation will be removed. The result of this process is the vertical lines that will be drawn in image. Next, we get vertical lines coordinate by using Houghlines function. The overall process in pole detection is shown in Fig. 1.

Houghlines function produces list of vertical lines that becomes the input basis for pole detection algorithm. These vertical lines are the rough contour of a pole. Pole detection algorithm further processes the lines through a series of sequence operations. First step ensures closest proximity vertical lines based on x coordinate are merged into a line, determined as the strongest edge of a pole. Next from these lines, all close lines in proximity are paired due to consideration of a pole would have close vertical edge distance. This will result on a situation where some lines pair might be originated from the same x coordinate. The final step of the algorithm is to choose the widest line pair as detected pole. Generally, a pair line algorithm to pole detection can be seen at Fig. 2.

The Canny edge detection algorithm starts by using Gaussian Filter to eliminate the noise from the image input which will produce image with minimum noise to the end result of the edge detection. The Gaussian filter will blur the image by using Gaussian function equation as follows:

$$G(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{x^2}{2\sigma^2}} \quad (1)$$

And the following 2D Gaussian equation:

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{x^2+y^2}{2\sigma^2}} \quad (2)$$

where by σ is the standard deviation and pixel to x, y point and acquires the heaviest weight of 1. As the image data is discrete, it is necessary to make Gaussian

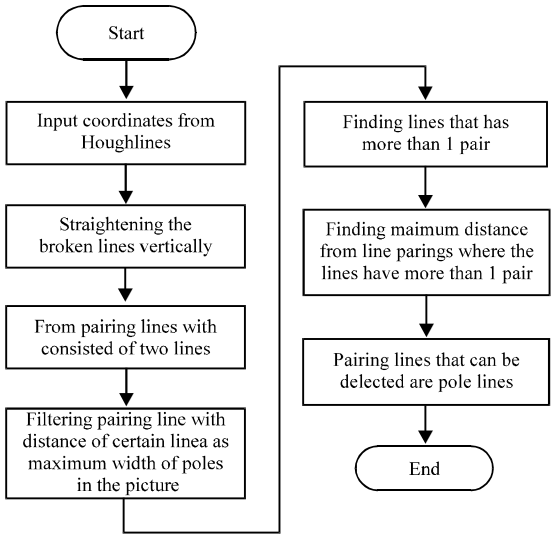

Fig. 2: A pair line algorithm

Table 1: Kernel matrix 5×5

P_0	P_1	P_2	P_3	P_4
0.003765	0.015019	0.023792	0.015019	0.003765
0.015019	0.059912	0.094907	0.059912	0.015019
0.023792	0.094907	0.150342	0.094907	0.023792
0.015019	0.059912	0.094907	0.059912	0.015019
0.003765	0.015019	0.023792	0.015019	0.003765

distribution to implement the Gaussian function to an image. To make the Gaussian distribution into a kernel mask which will be convoluted into image, the 1D or 2D Gaussian equation could be used. However, if we use the 1D Gaussian equation, we need to equate further because 1D Gaussian only use 1 coordinate that is x as distance from its distribution mean while the matrix formed image uses 2 coordinates that x and y. Therefore there will be a need to equate 1D Gaussian equation one more step for the y coordinate and the kernel mask form could be multiplied between x and y values.

In this research, the standard deviation used is $\sigma = 1$, and therefore, it is needed to determine the measurement of the kernel from its mask. At this point the kernel cut can be made so as to avoid value near/close to zero. Table 1 is the matrix from the result of the cut to become 5×5: The matrix can also alternatively be represented as integer values with a divisor such as Eq. 3:

$$\frac{1}{266}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 25 & 16 & 4 \\ 6 & 25 & 40 & 25 & 6 \\ 4 & 16 & 25 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} \quad (3)$$

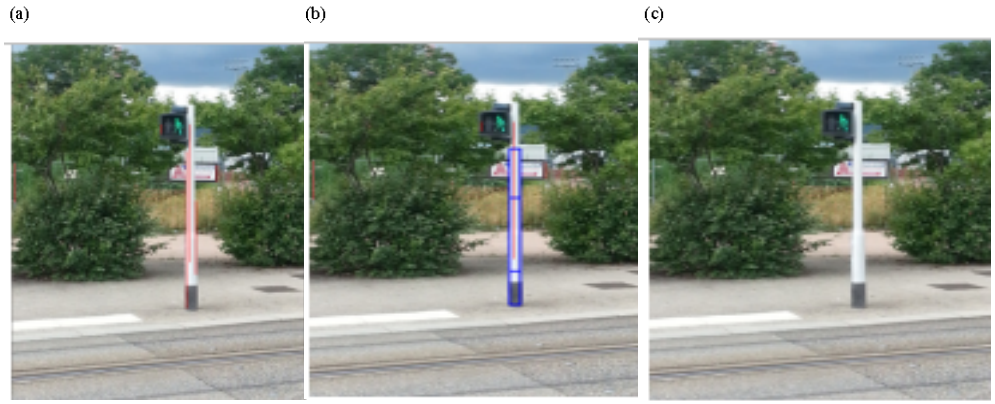

Fig. 3: Gaussian filter: a) Orignal imag; b) Grayscal image input and c) Gaussian filter results

Figure 3a-c shows the result of matrix filter towards the grayscale image input: In Canny algorithm, after being filtered using Gaussian filter gradient value from object of the image is sought. To do this, we need to use Sobel operation (Molton *et al*., 1998) that every pixel in the image are co-related with G_x and G_y by using the following matrix:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad (4)$$

$$G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (5)$$

Figure 4a, b is the result of the image determined by its gradient value from the image which has been filtered by Gaussian.

Co-relation matrix C are computed with scalar product operation, C = P, S, where P is the image pixel values. Assume P has the values as follow:

$$P = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad (6)$$

Then:

$$C_x = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} . \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (7)$$

$$= 1\times1+2\times2+3x1+4\times0+5\times0+6\times0+7x-1+8x-2+9x-1$$

$$= 1+4+3+0+0+0+(-7)+(-16)+(-9)$$

$$= -24$$

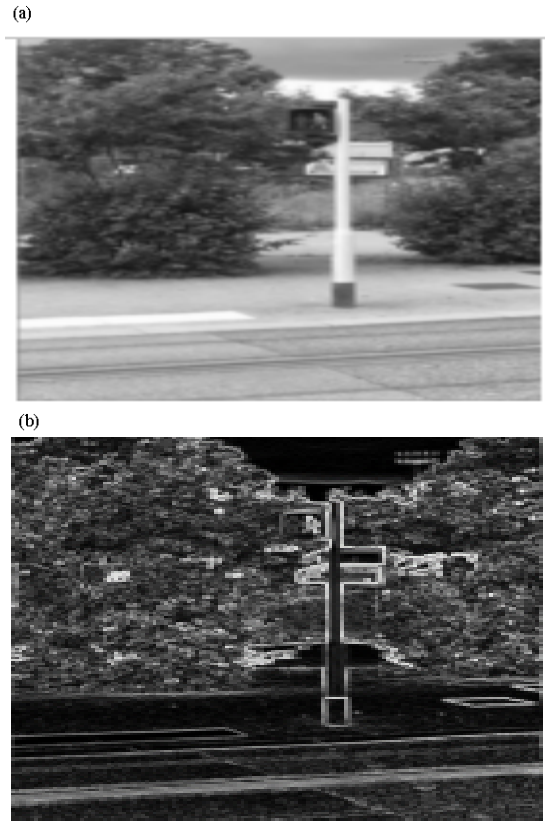

Fig. 4: Gradient from sobel operations: a) Gaussian filter results and b) Gradient results

Value -24 is placed into pixel 5. C_y is similarly computed but was co-related with G_y. The value of the gradient direction is calculated from each pixels of the image by using the following equation:

$$\theta = \tan^{-1}\frac{C_y}{C_x} \quad (8)$$

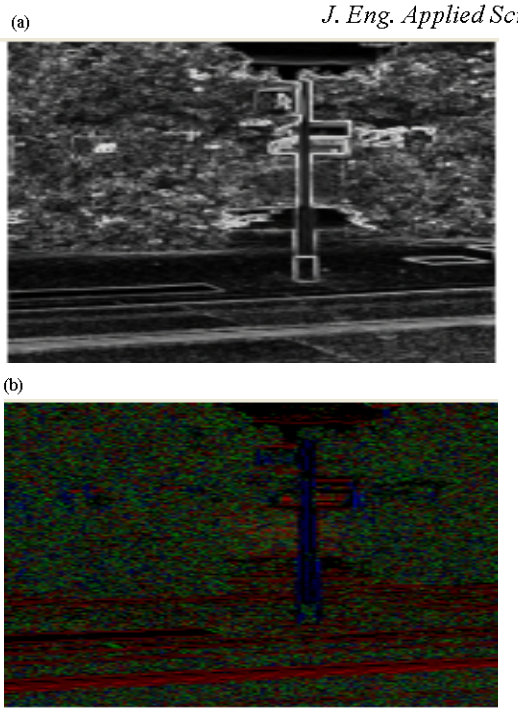

Fig. 5: Non-maximum suppression method: a) Gradient and b) NMS results

In the Canny algorithm, the θ value which would be derived will be placed into 4 groups that are close in proximity to 0, 45, 90 and 135° Table 2 for the exact interval.

In the Canny detection, the value of θ determines the threshold ratio in the gradient value G of a pixel image toward its neighbouring pixel. If the pixel value is bigger than its neighbouring pixel, then the value will be retained, otherwise it will be eliminated. The ratio direction of the pixel will follow θ value. If it is 0° then it will be compared to its left or right neighbour if 45° then it will be compared to diagonal right top and diagonal left bottom, if 90° then it will be compared to top and bottom pixel and finally if it is 135° it will be compared to left top diagonal value and bottom right diagonal value. In the Canny edge detection algorithm, this stage is called Non-Maxima Suppression (NMS) method (Juneja and Sandhu, 2009; Tang *et al*., 2014).

The NMS method is an edge thinning technique by applying ratio method the G (gradient) value from every pixel with its neighbours (Neubeck and Gool, 2006).

Figure 5a, b is the product of the image where the NMS method was executed. We can see the difference between the Sobel and NMS method where the NMS result is thinner than the previous input. The result of the implementation of the gradient calculation is that every

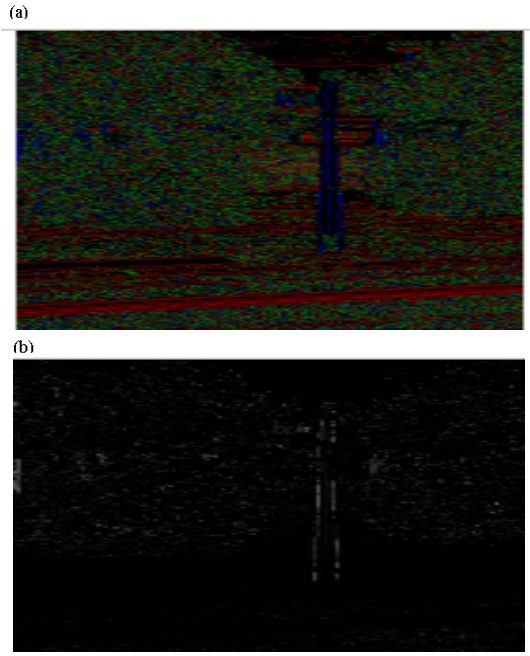

Fig. 6: Results of the elimation of the pixels: a) Results of NMS and b) Results of θ group of 0° and 180°

Table 2: Grouping of the gradient directions

Groups	Values
0 and 180°	0-22.5 and 157.5-180
45°	22.5-67.5
90°	67.5-112.5
135°	112.5-157.5

colour in Fig. 6 represents the gradient direction group such as blue represents the edges having 0 or 180°, red represents 90°, green represents 135° and yellow represents 45°. In NMS stage, this is modified and only the perpendicular line can be seen that is only 0 or 180degrees retained, the remaining lines (other than 0 or 180°) will be suppressed or eliminated. The result of the elimination of the pixels other than 0 or 180° is shown in Fig. 6a, b.

Further in Canny algorithm, the next step is hysteresis step that is eliminating further the noise using two parameter threshold (Matrella and Marani, 2011). These two parameters will be grouped into 3 parts: noise, strong pixel and weak pixel. If the pixel value is below the minimum threshold parameter, it is called noise. If the pixel value is higher than the maximum threshold parameter, it is called strong pixel and finally if the pixel value between these two parameters it will be considered as weak pixel. The noise will then be suppressed by giving pixel value equal 0. The value of the strong pixel will be increased to its maximum 255 (8 bit pixel). Then, the weak pixel will be calculated further by using 3×3 kernel. The

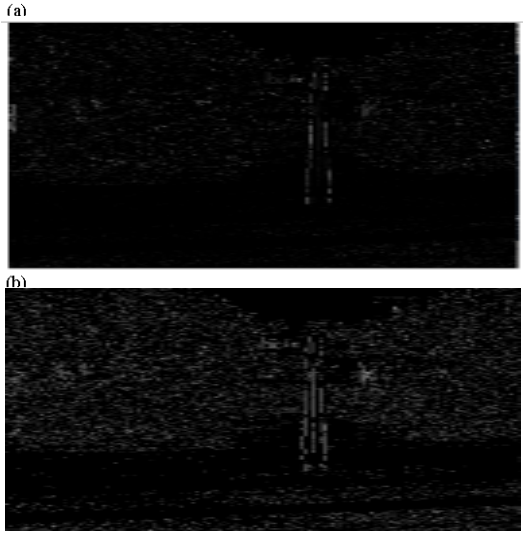

Fig. 7: Hysteresis process result: a) Results of θ group of 0° and 180° and b) Hystersis result

3×3 kernel is 3.3 matrix used as a mask to calculate the weak pixel. If the pixel enters into the weak pixel group, then this pixel will be set at 225 or compressed. This will be determined by looking at its neighbours in the 3×3 space of the weak pixel which is the central point. Table 3 shows 3×3 matrix as mask for the weak pixel.

If there is a strong pixel in P_0-P_8 then the P_{weak} will enter into the strong pixel and give the pixel value 255. On the other hand, if there is no strong pixel then the P_{weak} will be suppressed or eliminated.

This stage of hysteresis is called the confirmation stage that is there are only 2 pixel values, 225 and 0. In this stage, more edge lines can be seen and the edge noise pixel will be eliminated or suppressed. Figure 7a, b shows the example of the hysteresis result.

On completion of the modified canny detection stage that incorporated NMS until perpendicular line seen in the image, the process continues with Houghlines function which is used to detect lines formed by the filter in the form of certain length of lines. These lines are acquired from information in the forms of coordinates in the image. Figure 8 shows the result of Houghlines as seen in the original image. In Fig. 8, the lines can be seen on the post and is indicated by a red line which is the result of Houghlines from the image that had been Canny detected and optimized for vertical line detection only. Noted, we call detected lines as broken lines in this stage.

Next, there is a need for a certain algorithm that is able to detect a pole based on the broken lines produced from Houghlines. The algorithm works by forming line pairs,

Fig. 8: Houghlines result

Table 3: The 3×3 matrix as mask

P_0	P_1	P_2
P_3	P_{weak}	P_5
P_6	P_7	P_8

the coordinates of the lines are then saved in points pairing data structure where the lines are drawn from two points. A point is kept in an array (x, y) which represents coordinates of points in the cartesians space. The structure of the points pairing data is kept as matrix L:

$$L = [X_1, X_2, Y_1, Y_2]$$

Where:

X_1 = Column vector of x coordinates for first point
X_2 = Column vector of x coordinates for second point
Y_1 = Column vector of y coordinate for first point
Y_2 = Column vector of y coordinate for second point

Noted in vertical line X_1 and X_2 would have same value. Therefor, for simplification we wrote $L = [X_1, Y_1, Y_2]$. This Algorithm will process the L and produce G, which are the counterparts of the vertical lines detected as G pole measuring ×3 where m are the number of poles detected and $G = [P_1, P_2, D]$:

$$P = [p_1, p_2, ..., p_m]^T$$

Where:

P_1 = Set from first line
P_2 = Set from second line
D = Set from the distance between p_1 and p_2

This pole detection algorithm is as follows:

Pole detection algorithm:

```
1  Algorithm pole detection is
2      Input: Matrix L which store broken lines
3      Output: Matrix G which represent the detected poles
4          L2←connect_line_segment (L), to connect the vertical line is
           disconnected
5          G←detect_pole (L2)
6  Return G
```

This algorithm calls two sub-functions: connect_line_segment and detect_pole. The connect_line_segment function is used to connect broken lines into a single vertical line for adjacent x coordinate and the detect_pole function is used to find the pair of adjacent vertical line as the characteristics of the pole.

Connect line segment function: The input of the parameters from this function is matrix L and produce matrix L_2 which are connected Ls. The aim of this algorithm is to connect the vertical lines that is within the distance of the x coordinates that are closed to one another or x_1:x position on the first point, x_2:x position in the 2nd point and d: distance between x_1 and x_2, so, d: = $|x_1-x_2|$, d≤threshold.

Next the two lines which fulfil these criteria or considered to be within the x coordinates that close to one another are connected and formed into a line or y_{min}: is the lowest coordinates that are differentiated, y_{min}: is the highest that are differentiated and y_{min} = min (y_1, y_2, y_3, y_4), y_{man} = man (y_1, y_2, y_3 y_4). One line keeps two y coordinates, so that, when the two lines are differentiated then there will be four y coordinates.

Each point will be differentiated with all the points included in these points. This process is known as one sweep. Consider n:= number of lines detected, then 1 sweep will move at the value of n. So n^2 sweep will be moved to differentiate all the lines. This process can be streamlined by sorting the coordinates x before that, which will ensure the most adjacent point will be next to it so that, it only takes n time to connect the vertical lines. The algorithm is as follows:

Connect line algorithm segment function:

```
1   Algorithm connect_line_segment is
2   Input: L which keeps the line data
3          thres, threshold distance that are tolerated
4   Output: L2, is the lines that are already connected
5   Let Ls is a pair of lines that are in sequence based on the coordinates
    x
6   L←sort (L)
7   Let L2, data structure which will gather line outputs
8   Initialize L2 = [ ], as empty matrix
9     L2 (k):=L (0), keep first line in L2
10  Let k as index to L2
11  Set k:= 0
12  Set xpivot, keep x coordinate on the 1st line
13  Set ypivot, keep all y coordinates on the 1st line
14  For i:=1 to n-1
15      If |xpivot,-Lx (i)|<there then
16       Update y in position L2
17              Update L2y (k)←[min ( ypivot, Ly (i)), max (ypivot, Ly (I))]
18                                  Else
19               Update k:= k+1
20              Insert the new entry in L2
21               Insert L2y (k)←Ly(i)
22      Set L2x (k)←Lx(i)
23      Update
24      Update ypivot:= Ly(i)
```

Detect pole function: The Input parameter function is L matrix from connect_line_segment function and produces G. This algorithm has the task of pairing the closes pairs of lines or the distance of x coordinates between the two lines less than a certain thresholdto be taken maximum distance. This is done by considering that the assumptions detected as a pair of adjacent horizontal lines. Only a pair of lines that meet the criteria will be recorded as G. The algorithm is as follows:

Detect pole function algorithm:

```
1   Algorithm detect_pole is
2   Input: L which keeps the data of the lines connected
3          thres, as tolerance threshold of horizontal distance
4   Output: G, matrix which keeps the filtered pairing of lines
5      Let nas as length of L
6   Initialize G as empty matrix with 3 columns
7   Initialize D as distance matrix n×n
8     Let k as tracker index from G
9     K:= 1
10  For i:= 0 to n-1
11          For j:= i+1 to n-1
12                  Let d as x distance between 2 lines
13                  Set d:= |xi-xj|
14                  If d≤ thres and i not equal jthen
15                          D (i, j) = d
16          Initialize max:= 0
17          Initialize imax:= 0
18          Initialize ctr:= 0
19          For j:= i+1 to n-1
20                  If D (i, j)>max
21                          Update max:= D (i, j)
22                          Update imax := j
23                          Update ctr :=ctr+1
24          If ctr>1
25                Set G(k, 1) := i
26          Set G (k, 2) := j
27          Set G (k, 3) := max
28          Update k := k+1
```

RESULTS AND DISCUSSION

Log running algorithm: Figure 9 is the result of the log running algorithm and the testing of the pole detection program. To run the algorithm used MATLAB by entering the Houghlines data for each picture (Fig. 1-3) and to run the pole detection program used Netbeans by testing several images. From the results of both between the log running algorithm and the pole detection program, it is seen consistency in detecting the pole. The green

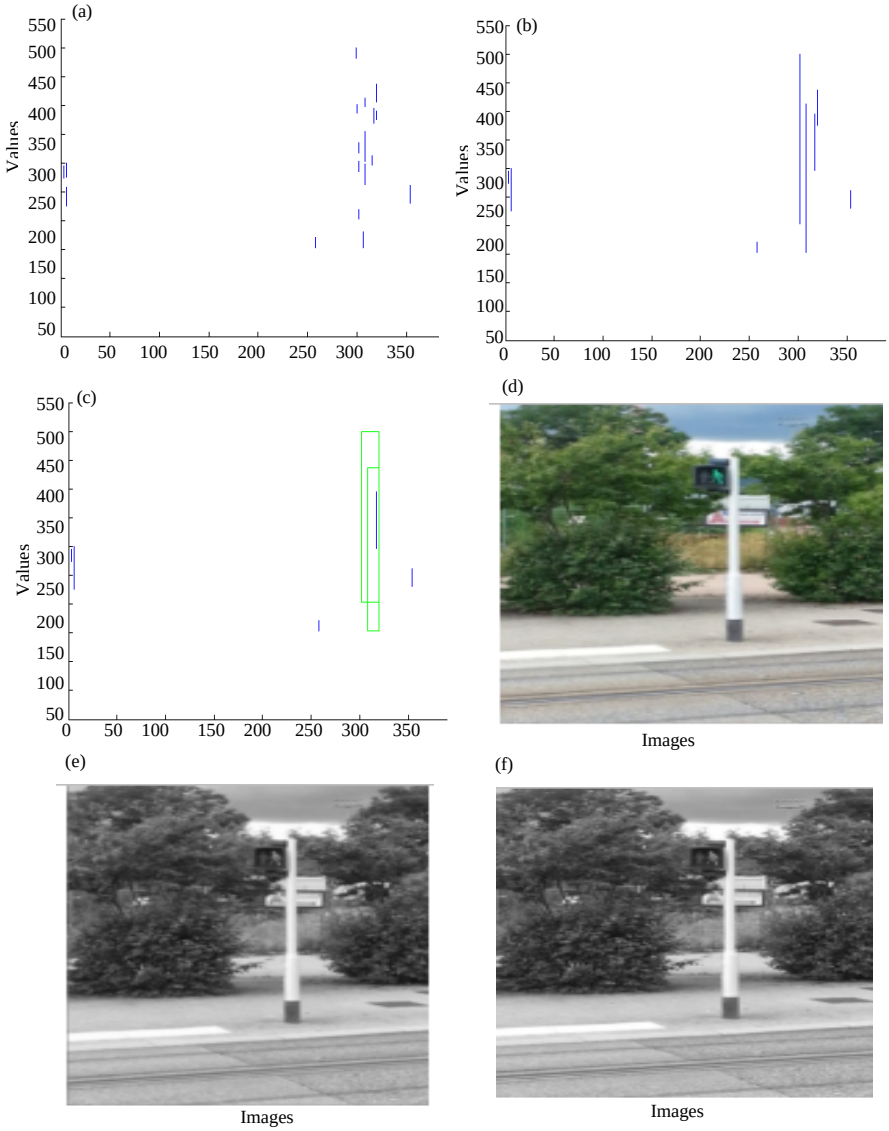

Fig. 9: Case 1 on pole deatection: a) The orignal lines; b) The connected lines; c) The pole detected; d) The orignal images; e) Vertical lines detection and f) Pole detected

rectangular box on the figure shows the pole detected in log running algorithm and the blue rectangular box on the Figure shows the pole detected in application program. In the case 1 (Fig. 9a-f), there were 2 and 3 detected poles however all of them led to the same pole. Therefore, we conclude the result is correct.

In the case 2 (Fig. 10a-f), the input image consisted of a close pole and a far pole. The Houghlines correctly produced vertical mark. However, there was only a detected pole. This was caused by although the Houghlines produce vertical mark but only one sides of the pole was correctly determined.

In the case 3 (Fig. 11a-f), the input image consisted of a series of poles following the road. Notice that, here all poles had certain vertical Houghlines mark for the closest poles both sides of the pole were correctly determined. But for the far poles, the distance between two sides of the pole was miniscule that they were eliminated in the final pole detection. The important points here were although, only the closest pole were detected but as the user walks further the distance pole would come closer. The algorithm would re-run such that the closest pole would again be detected.

Hardware specification and criteria of the real-time test: During the test on the pole detection, the SEE-Stick used consisted of the following hardware (Yusro *et al*., 2014):

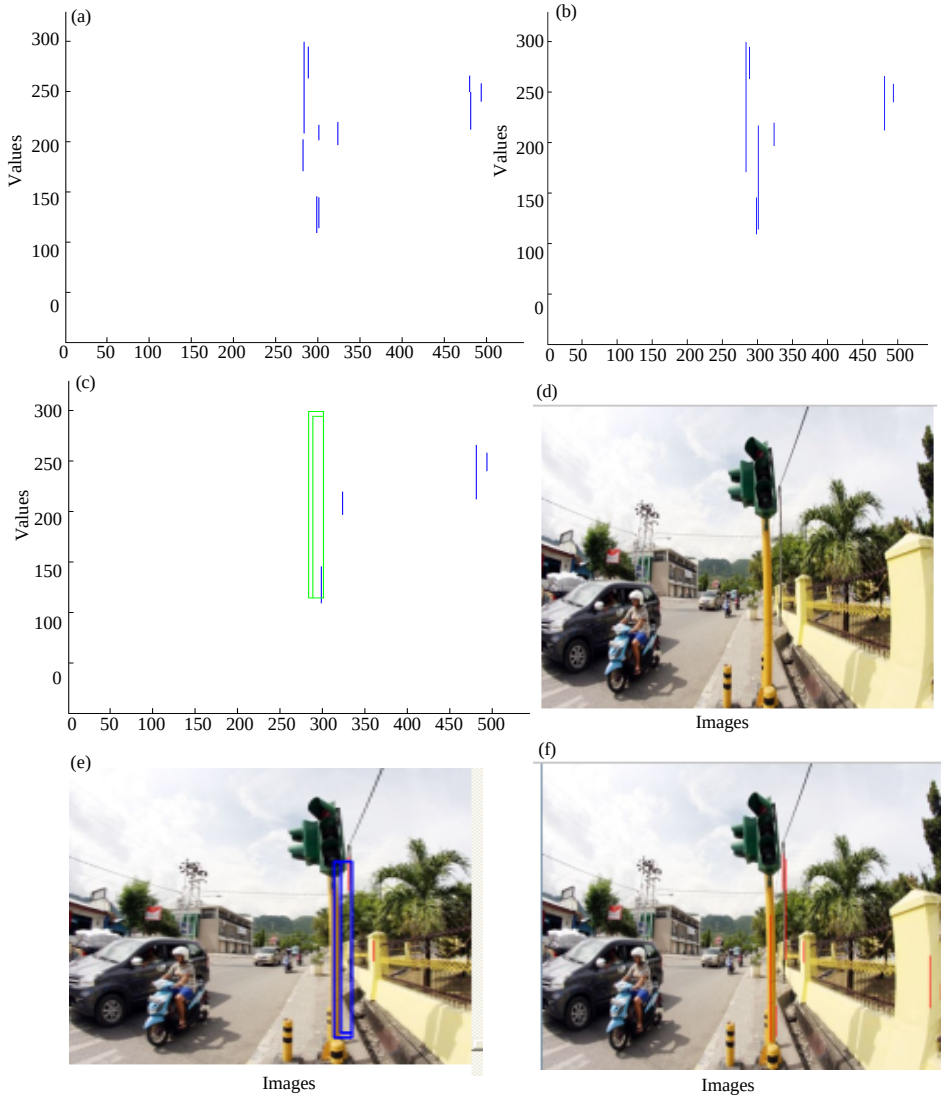

Fig. 10: Case 1 on pole deatection: a) The orignal lines; b) The connected lines; c) The pole detected; d) The orignal images; e) Vertical lines detection and f) Pole detected

- Minicomputer Raspberry Pi Model B which has the following specifications: 900 MHz quad-core ARM Cortex-A7 CPU; 1GB RAM; 4 USB ports; 40 GPIO pins; Full HDMI port and Ethernet port
- USB Wi-Fi with the following specifications: built in smart antenna; IEEE 802.11n, backward compatible with IEEE 802.11g and IEEE 802.11b; Frequency range from 2.4-2.4835 GHz; 1-13 working channel; 20 dBm (max) transmit power and natively supported by the Raspbian Wheezy Linux operating system distribution onwards
- The following types of Webcam: (Cam1 [1.3 MP], Cam2 [3.0 MP], Cam3 [5.0 MP] and Cam4 [8.0 MP]).
- Power DC Regulator: 5 V, 2.1 A

The SEE-stick is a design in the form of a stick with the length of 1.5 m which can be adjusted, 45 cm width and has 3 wheels (2 wheels in front and 1 in the back). The hardware device is placed on the stick of the SEE-stick. The physical appearance of the SEE-stick used for the object detection can be seen from Fig. 12a, b.

The tested shape of the object (pole) was a cylindrical shape pole with the height of 2.5 m and with the diameter of 3 inch shown in Fig. 13 a-c. The first thing that was done by the program was to process the information of the image acquired from the webcams. In this research, real-time detection of an object was being tested by using various models of webcams of different

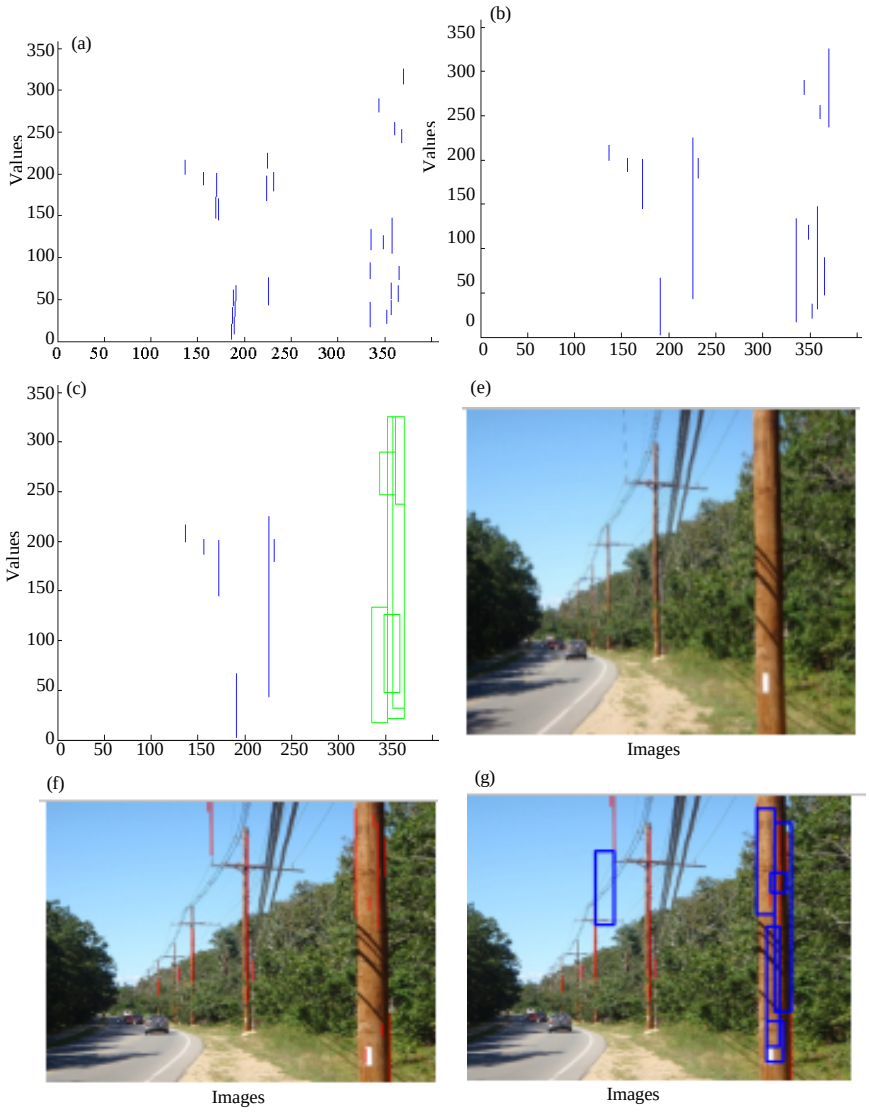

Fig. 11: Case 1 on pole deatection: a) The orignal lines; b) The connected lines; c) The pole detected; d) The orignal images; e) Vertical lines detection and f) Pole detected

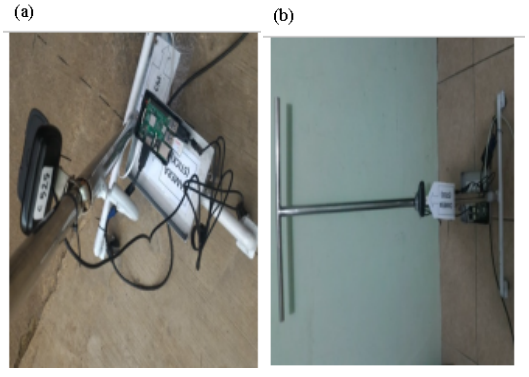

Fig. 12: The phisical of the SEE-stick: a) SEE-stick (side) and b) SEE-stick (front)

Table 4: Criteria of objects (poles)

Name	Height (m)	Diameter (inch)	Colour	Materials
Pole 1	2.5	3	Silver	Plastic (PVC)
Pole 2	2.5	3	Black	Plastic (PVC)

Table 5: Criteria of test camera

Name	Type	Image capture	Video capture	Lens
Webcam 1	Cam1	Up to 1.3 megapixels	640×480 pixels	Manual
Webcam 2	Cam2	Up to 3.0 megapixels	1280×720 pixels	Manual
Webcam 3	Cam3	Up to 5.0 megapixels	1280×720 pixels	Manual
Webcam 4	Cam4	Up to 8.0 megapixels	1280×720 pixels	Autofocus

types of resolution. These tests were made day and night, indoor and outdoor. The criteria of the object (poles) and camera used are as in Table 4 and 5.

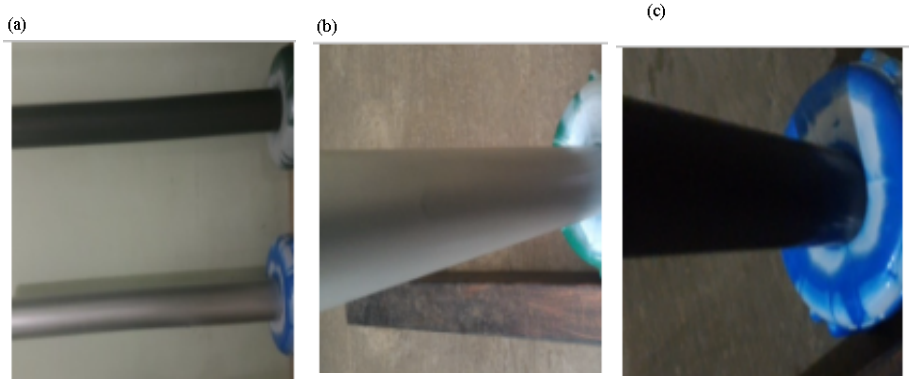

Fig. 13: The pole tested: a) Both poles; b) Silver pole and c) Black pole

Table 6: The relationship between the camera resolutions, time of test, colour of pole with the object detected up to the distance of < 300 cm

	Object detected up to the distance of <300 cm					
	Poor		Good			
Variable	N	(%)	N	(%)	OR (95% CI)	p-values
Camera resolution						
Cam1	8	100.0	0	0		
Cam2	6	75.0	2	25.0	1(0.104-9.614)	1
Cam3	5	62.5	3	37.5	1(1.104-8.614)	1
Cam4	3	37.5	5	62.5	5(0.584-42.797	0.142*
Time test						
Daytime	10	62.5	6	37.5	0.556(0.122-2.536)	0.703
Night time	12	75.0	4	25.0		
Location test						
Outdoor	8	50.0	8	50.0	0.143(0.024-0844)	0.057*
Indoor	14	87.5	2	12.5		
Colour of the poles						
Black	14	87.5	2	12.5	7(1.185-0.844)	0.057*
Silver	8	50.0	8	50.50		

Analysis of the statistical data on the result of the real-time test: The researcher performed an experimental test on the ability of pole detection system (SEE-stick) and made a statistic test to ascertain what variable had an effect on the system ability. In the data processing of the pole detection experimental test, the researcher used SPSS. The data analysis used Chi-square and logistic regression. In this experimental test, variable presumed to have an effect on the ability of pole detection are: camera resolution (Cam1 [1.3 MP], Cam2 [3.0 MP], Cam3 [5.0 MP] and Cam4 [8.0 MP]) time of test (daytime and night time), location of test (indoor and outdoor) and colour of pole (black and silver) The criteria of the experimental test are as follows:

- The camera resolution is differentiated based on the type of cameras used
- Daytime test is between 09.00-11.00 h when the sky is bright and the sun is not covered by the clouds and the night time test is 19.00-21.00 h with the aid of 25 W TL lamp
- Location of the outdoor test is an open area whereas the location of the indoor test is in a room. The location of the indoor test between daytime and night time is assumed to be same with the aid of 25 W TL lamp.
- The silver coloured poles are painted with silver doff paint while the black poles are poles which are painted with black doff paint. The shape of the object tested was a pole in the shape of cylinder with height of 2.5 m and 3 inches in diameter

The SEE-stick ability is the ability to detect the existence of the pole at the distance of 300 cm. If the system is able to detect the poles from the distances of 50 100, 150, 200, 250 and 300 cm, then it can be categorised as good but if the system cannot detect the existence of the poles at one of these distances, then it is not good. After the trial test had been performed, the results are as in Table 6.

Table 6 indicates that the use of Cam4 (which has a

resolution of 8.0 MP) was better in detecting the poles than other cameras (OR-5, p = 0.142). Daytime was better in detecting poles compared than night time (OP = 0.556, p = 0.703). Testing indoor was not as good as in detecting poles in the daytime. Detecting silver coloured poles was better than black coloured poles (OR = 7, p = 0.0576).

Table 7: The multivariate results

Variable	OR (95%CI)	p-values
Location of the test outdoor, indoor	0.085 (0.010-0.699)	0.022
Colour of the poles, black, silver	11.757 (1.430-96.634)	0.022

The value of $p<0.25$ indicated that there was a connection and interaction between the variables. So, as to determine the variables that were inter-connected and most dominantly affected the pole detection system, a multivariate test should be made to the 3 variables that were camera resolution location of the test colour of the pole. The final result of this multivariate research mode is as follows (Table 7).

The above table indicates that there are 2 dominant variables which affect the ability of SEE-stick in detecting the existence of the pole objects up to <300 cm that is the location of the test and colour of the poles ($p<0.05$). The indoor location test caused the SEE-stick not as good in detecting the pole at 0.085 times compared to the outdoor location test (OR = 0.085, p = 0.022) while the use of silver colour caused the SEE-stick 11.757 times better in detecting the existence of the pole than the black coloured poles.

The SEE-stick indoor trial test had a setback in its ability in detecting the presumed poles because of the poor lighting effect while in the outdoor test (especially in the daytime) the result was better because of a better lighting effect. In the SEE-stick trial test in detecting the pole, silver poles are better than the black poles.

CONCLUSION

The investigation on maximizing the SEE-S (Smart Environment Explorer-Stick) function in detecting obstacle was performed in this study. The study proposes an algorithm advancement to object detection to detect the presence of obstacle (pole). This algorithm uses a method of distance calculation and pairing search of vertical lines obtained from optimized Canny detection. The position of the vertical lines is found using Houghlines function. During simulation using MATLAB and Netbeans with some of pole images in a variety of conditions/ circumstances can be concluded that the algorithm worked well. The algorithm can detect and recognize the pole and differentiate it with non-pole object. The test result in real-time on the program algorithm in detecting poles using cameras with different resolutions (1.3-8.0 MP) with different colors of poles (silver and black), performed during daytime and night time and indoor and outdoor location. The algorithm works satisfactorily in detecting poles based on two dominant variables: location and colour. At indoor the algorithm performance was worse than at outdoor and the use of silver pole improves the performance of the algorithm.

Enhancement on the algorithm in detecting various objects and investigating Human Machine Interaction (HMI) in collaboration with VIP will be further concern in the near future work. In addition, the improvement of pole detection algorithm can be considered by applying superpixel and blobs technique for image segmentation.

ACKNOWLEDGEMENTS

This resaerch was supported by the Indonesian and French Governments in Double Degree Program for Research and Higher Education. Researchers wished to acknowledge Prof. Edwige Pissaloux, Mr. Imam Firmansyah and Mr. Eka Suryana for the fruitful discussions and fabulous advices during completing this project. The authors would like to thank also for some anonymous reviewers for their constructive comments and insightful suggestions that improved the quality of this study.

REFERENCES

Cabo, C., C. Ordonez, S.G. Cortes and J. Martinez, 2014. An algorithm for automatic detection of pole-like street furniture objects from mobile laser scanner point clouds. ISPRS. J. Photogramm. Remote Sens., 87: 47-56.

Costa, P., H. Fernandes, P. Martins, J. Barroso and L.J. Hadjileontiadis, 2012. Obstacle detection using stereo imaging to assist the navigation of visually impaired people. Procedia Comput. Sci., 14: 83-93.

Dakopoulos, D. and N.G. Bourbakis, 2010. Wearable obstacle avoidance electronic travel aids for blind: A survey. IEEE. Trans. Syst. Man Cybern. Part C., 40: 25-35.

Fernandes, L.A. and M.M. Oliveira, 2008. Real-time line detection through an improved hough transform voting scheme. Pattern Recognit., 41: 299-314.

Hakobyan, L., J. Lumsden, D.O. Sullivan and H. Bartlett, 2013. Mobile assistive technologies for the visually impaired. Surv. Ophthalmol., 58: 513-528.

Juneja, M. and P.S. Sandhu, 2009. Performance evaluation of edge detection techniques for images in spatial domain. Int. J. Comput. Theory Eng., 1:

1793-8201.

Lee, Y.S., H.S. Koo and C.S. Jeong, 2006. A straight line detection using principal component analysis. Pattern Recognit. Lett., 27: 1744-1754.

Lehtomaki, M., A. Jaakkola, J. Hyyppa, A. Kukko and H. Kaartinen, 2010. Detection of vertical pole-like objects in a road environment using vehicle-based laser scanning data. Remote Sens., 2: 641-664.

Matrella, G. and D. Marani, 2011. An embedded video sensor for a smart traffic light. Proceedings of the 14th Euromicro Conference on Digital System Design (DSD), August 31-September 2, 2011, IEEE, New York, USA., ISBN:978-1-4577-1048-3, pp: 769-776.

Molton, N., S. Se, J.M. Brady, D. Lee and P. Probert, 1998. A stereo vision-based aid for the visually impaired. Image Vision Comput., 16: 251-263.

Neubeck, A. and L.V. Gool, 2006. Efficient non-maximum suppression. Proceedings of the 18th International Conference on Pattern Recognition (ICPR'06), August 20-24, 2006, IEEE, New York, USA., ISBN:0-7695-2521-0, pp: 850-855.

Oji, R., 2012. An automatic algorithm for object recognition and detection based on ASIFT. Signal Image Process. Intl. J., 3: 29-39.

Praveen, R.G. and R.P. Paily, 2013. Blind navigation assistance for visually impaired based on local depth hypothesis from a single image. Procedia Eng., 64: 351-360.

Serna, A. and B. Marcotegui, 2014. Detection, segmentation and classification of 3D urban objects using mathematical morphology and supervised learning. ISPRS. J. Photogramm. Remote Sensing, 93: 243-255.

Sharma, A., 2010. Object detection in image using particle swarm optimization. Intl. J. Eng. Technol., 2: 419-426.

Strumillo, P., 2010. Electronic interfaces aiding the visually impaired in environmental access, mobility and navigation. Proceedings of the 3rd International Conference on Human System Interaction, May 13-15, 2010, IEEE, Lodz, Poland, ISBN:978-1-4244-7560-5, pp: 17-24.

Tang, S., M. Andriluka and B. Schiele, 2014. Detection and tracking of occluded people. Intl. J. Comput. Vision, 110: 58-69.

Ushma, A., P.M. Scholar and A.M. Shanavas, 2014. Object detection in image processing using edge detection techniques. IOSR. J. Eng., 4: 10-13.

Wang, S., H. Pan, C. Zhang and Y. Tian, 2015. RGB-D image-based detection of stairs, pedestrian crosswalks and traffic signs. J. Visual Commun. Image Represent., 25: 263-272.

Yokoyama, H., H. Date, S. Kanai and H. Takeda, 2013. Detection and classification of pole-like objects from mobile laser scanning data of urban environments. Intl. J. CAD. CAM., 13: 31-40.

Yu, C., C. Huang and Y. Lang, 2010. Traffic light detection during day and night conditions by a camera. Proceedings of the IEEE 10th International Conference on Signal Processing Proceedings, October 24-28, 2010, IEEE, Shenyang, China, ISBN:978-1-4244-5897-4, pp: 821-824.

Yusro, M., K.M. Hou, E. Pissaloux, H.L. Shi and K. Ramli *et al*., 013. SEES: Concept and design of a smart environment explorer stick. Proceedings of the 2013 6th International Conference on Human System Interactions (HSI), June 6-8, 2013, IEEE, Blaise Pascal, France, ISBN:978-1-4673-5635-0, pp: 70-77.

Yusro, M., K.M. Hou, E. Pissaloux, K. Ramli and D. Sudiana *et al*., 2014. Concept and Design of SEES (Smart Environment Explorer Stick) for Visually Impaired Person Mobility Assistance. In: Human-Computer Systems Interaction: Backgrounds and Applications 3, Zdzislaw, S.H., L.K. Juliusz, T. Mroczek and J. Wtorek (Eds.). Springer, Berlin, Germany, ISBN:978-3-319-08490-9, pp: 245-259.