Image Based Centre Object Calculation Using Coordinate Averaging Method for Object Following Mobile Robot

Aditya Kurniawan
Politeknik Kota Malang, Raya Tlogowaru no 3, Malang, Indonesia
Email: aditya@poltekom.ac.id

Hendrawan
Institut Teknologi Bandung, Ganesha No 10, Bandung, Indonesia
Email: hend@stei.itb.ac.id

Kholilatul Wardani
Politeknik Kota Malang, Raya Tlogowaru no 3, Malang, Indonesia
Email: kholilatul@poltekom.ac.id

Eueung Mulyana
Institut Teknologi Bandung, Ganesha No 10, Bandung, Indonesia
Email: eueung@stei.itb.ac.id

***Abstract*—Research about robot vision is growing and more focusing on using image processing algorithm to enhance the sensing ability of a robot mobile to aware about its surroundings. This research using Kinect sensor which has 2 cameras, one RGB visible light and IR camera / depth camera both with 640x480 pixels at 30fps (frame per second). An object following method is applied on a 4WD differential mobile robot chassis equipped with Kinect sensor 1.0 running at 640x480 pixels, 30fps. The method using pixel coordinate averaging which summing all the pixel coordinate of the object of interest. Each calculation (vertical and horizontal) will result in a single integer value of coordinate in i or j showing the center of the object. Two tests object using symmetrical and non-symmetrical shows that the center object only moves slightly off about 1 pixel on j axis. The result of the calculation for symmetrical object on j axis is 9 (rounded down) and for the non-symmetrical object is 8 (rounded up). with the calculation on i axis is similar (on pixel 14).**

***Index Terms*—center object, kinect, mobile robot, object follower**

I. Introduction

Robot Vision as a method of robot sensing has become more popular. More researches are focusing on enhancing robot sensing to become more “human like” and replacing the majority of vision sensing other than camera. Camera as vision sensing for robots provide more information in a high resolution all in one device. Normally the resolution range for HD camera is 1280 x 720 that adds up to 921.600 pixels to 1080p HD which has 2.073.600 pixels. Technically, that means it has over 2 million color sensor that could be used to extract various information about its environment.

Previous research [2];[5];[7] has attempted to use Kinect visible camera that that has 450 x 480 VGA with 30fps framerate, 57° horizontal and 43° vertical angular field of view [1] using both visible camera and depth camera of the Kinect sensor to localize the object for further processing. This paper is then combining previous method to locate an object with a method to calculate a center point of a target object. The object center point calculation using multiple isolated filtering method is applied as preprocessing algorithm. As a result, the previous processing will produce an i and j integer value showing the position of the target object. The method has been applied on a robot model 6WD wild thumper differential chassis using LabVIEW programming environment.

II. Kinect Visual Sensor Coverage Area

A. Spatial Resolution

The resolution for Kinect camera is 640 pixel horizontally and 480 pixel vertically, It adds up to 307.200 pixels in total. Therefore, it only has 307.200 data in RGB format (RRR, GGG, BBB) in 24 bits for every image captured [9]. The illustration of Kinect spatial resolution is given in Fig. 1.

Manuscript received July 7, 2020; revised December 2, 2020.

doi: 10.18178/joace.8.2.26-31

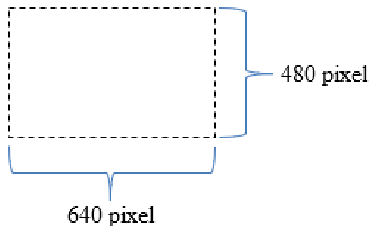

Figure 1. Kinect camera spatial resolution.

B. *Nominal Spatial Resolution*

Given the spatial resolution in pixel, then nominal spatial resolution in real world could be defined as follows:

Let

D : field of view distance from Kinect
A : coverage area of the camera
Sres : Nominal spatial resolution in mm/pixel

Then

$$Sres = \sqrt{\frac{(2d\tan 28.5°)x(2d\tan 21.5°)}{307200}}\ x\ 10^{6}\ mm/pixel \quad (1)$$

Therefore, the nominal spatial resolution depends on the distance of the field of view from the sensor camera. The more the distance, the less the nominal spatial resolution (higher value in mm/pixel).

C. *Nominal Depth Resolution*

Nominal depth resolution is the resolution of the IR camera sensing the distance of target object. The nominal dept resolution in 2m distance is 1cm, and the further the distance, the less the depth resolution [1]. The resolution graph could be seen in Fig. 2.

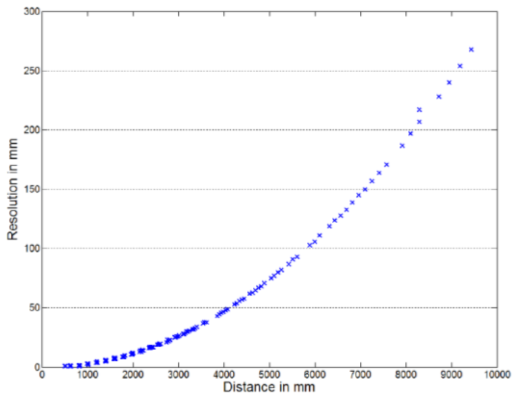

Figure 2. Depth resolution vs distance graphs

III. Robot Model

A. *Hardware*

Robot model that will be used for testing the algorithm is a 6WD wild thumper mobile robot that is attached with Kinect sensor on the top of the chassis. The Kinect sensor is used for capturing image in 640 x 480 pixels frame taken at 30 fps (frame per second). The robot model is shown in Fig. 4. This robot is equipped with an Arduino Due controller for low level dynamic movements control, and a mini computer for handling the image processing algorithm. The unique specification of the hardware model is given on Fig.3.

Hardware spesification of robot model	
Robot Weight	: 7 Kg
Max Body Work Weight	: 5 Kg
Actuators	: 6x DC motor gearbox 1:75 (160 RPM) 11Kg.Cm Torque 1x LED 1x Passive Buzzer
Sensors	: 2x Kinect Cameras 2x Infrared sensors 1x environment sensor 2x HCSR04 ultrasonic sensor Sensor AM2301 Sensor DHT21 Sensor BMP 180 Digital Pressure Sensor Gas MQ 135 Light Intensity Sensor UV Sensor
Controller	: Arduino Due & nano
Connectivity	: Bluetooth HC-06 (9600 bps), GPRS Module SIM 900 with mobile number (+62895627288014)
Battery	: 12,6 Volt max ; 5Ah 20C (Li-Po)

Figure 3. Hardware specification.

The model will be used as a prototype for developing the ITIS (Intelligent Transportation Information System) mobile robot called MRDC 1.0 which will function as a mobile station data logger for weather, pollution, traffic density and road environments that will collect data and send it to a central server. This silo data will always be updated to give road users reliable and accurate information about the real condition of the highway. The data can then be displayed on an Android-based cellphone and a website called travelersense.

Figure 4. 6WD chassis wild thumper robot.

B. *Simulation for Robot Movements*

The robot model dynamics will be simulated using mobotsim v1.0 to further understand the basic algorithm on "avoider program" in addition to object following task. The software however, has a limited simulation on robot geometry in which only rounded shape is possible. The crucial configuration of the robot on simulation are position (in cartesian), platform diameter (0.5m), distance between wheels (0.297m), wheels diameter (0.126m), wheels width (0.1m), number of ranging sensors (4 sensors), angle between sensors (60 degrees), sensors ping radius (0.2m), and radiation cone (25 degrees). The visualization of the configuration is shown on Fig. 5.

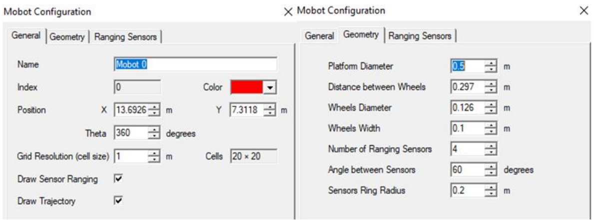

Figure 5. Robot simulation configuration on MobotSim v1.0

To maintain the accuracy of the simulation, the configuration is set in accordance to the real robot model sensors. The configuration of the sensors are 4 ranging sensors which are 2 infrared sensor with narrower sensing cone on the front and 2 HCSR04 ultrasonic sensors with wider sensing cone on the side. The visualization of the sensing range is shown on Fig. 6.

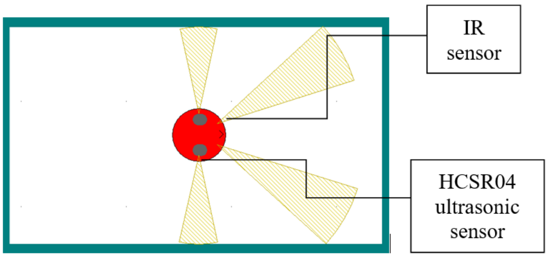

Figure 6. Sensor Configuration

The movements result of the simulation for avoider program is shown in Fig. 7. The program is simulated using 0.2 time step with 5000 iteration for each starting point, namely SP1, SP2, SP3, and SP4. Each start points (SP) is defined randomly to show the natural movements of the robot in simulation. The sensors is set to be activated if the obstacle is less than 0.5 metre from the sensor, and using 2 motion control strategies. The strategy is sharp turn by setting the speed of the opposite group of wheels by 0 (stop) and slight turn by setting the speed of the opposite group of wheels 50% less.

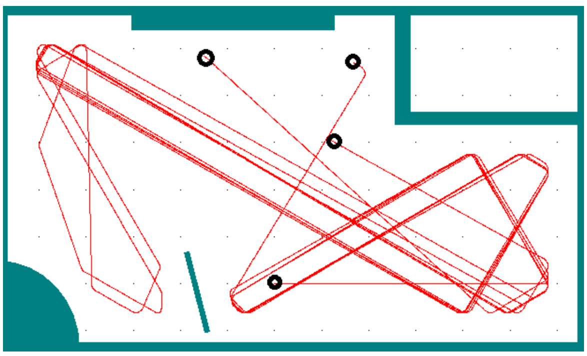

Figure 7. Movement paths result for avoider

The robot path in Fig. 7 shows that the robot can move freely inside of the wall while executing "following object" task.

C. Software and Library

To simplify the movements of the robot, a library called poltekomDuino is used. PoltekomDuino software library is a library (program function) that bridges between programmers and more complicated algorithms on Arduino Due, so that by using this library, novice programmers can perform / utilize more complicated functions with simpler commands by typing a syntax command and parameters. The example syntax is given in the example menu shown on Fig. 8.

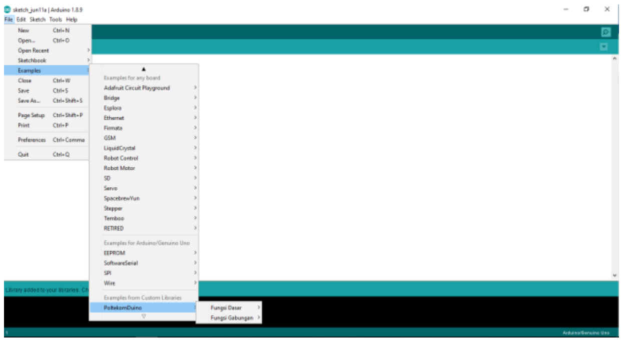

Figure 8. Library function menu used in the model

IV. Object Centre Point Calculation

A. Initial Process

This initial process using multiple localized filtering algorithm applied to an image array captured by visible camera of the Kinect sensor. This filter is a multiple masking method applied on a layer over layer using a mean masking [2];[3]. The following method will be executed at the work point or called ROI (Region of Interest). To remove the noise even further, Gaussian masking will be executed on the top of averaging mask. To isolate the region of interest, edge detection is used to segment the image area to be then processed using multiple filtering.

1) Algorithm for image segmentation

Image segmentation / isolation process is done to get the edge of the object from the object image. A point (i, j) is defined as the edges of an image if the point has high difference value with the surrounding pixel value [4];[6];[8].

Let

$P_{(i,j)}$: *pixel value on row i column j*
$P'_{(i,j)}$: *pixel value on row i column j*
$M_{(i,j)}$: *filter used for isolation*

Then

$$P'_{(i,j)} = \frac{1}{3}\, x \sum_{0}^{i=2} \sum_{0}^{j=2} M_{(i,j)}\, x\, P_{(i,j)} \quad \text{........ (2)}$$

2) Masking using multiple filtering technique

This process is done to get a smoother edge. A point / pixel that has a significance difference is assumed to be a noise. By doing the process of masking Neighborhood-Averaging (mean) and Gaussian, elimination of the noise is expected.

Let

$P'_{(i,j)}$: *pixel value on row i column j after segmentation*
$P''_{(i,j)}$: *pixel value on row i column j*
$M_{(i,j)}$: *filter used for smoothing*

Then

$$P''_{(i,j)} = \left(\frac{1}{3}\ x \sum_{0}^{i=2} \sum_{0}^{j=2} M_{(i,j)}\ x\ P_{(i,j)} \right) x\ Mm_{(i,j)}\ x\ Mg_{(i,j)} \quad .(3)$$

B. *Centre Point Calculation for Single BLOB, Symmetrical Object*

The object centre point is calculated by finding the average of every "black" pixels found in the image. the horizontal centre coordinate is found by summing all "black" pixels coordinates and divided them by the number of its pixels (average). The programming in the next section will elaborate on how to count pixel coordinate in each row and calculate the average using array count.

An image with 20x20 pixel is made to illustrate the working principal on the method.

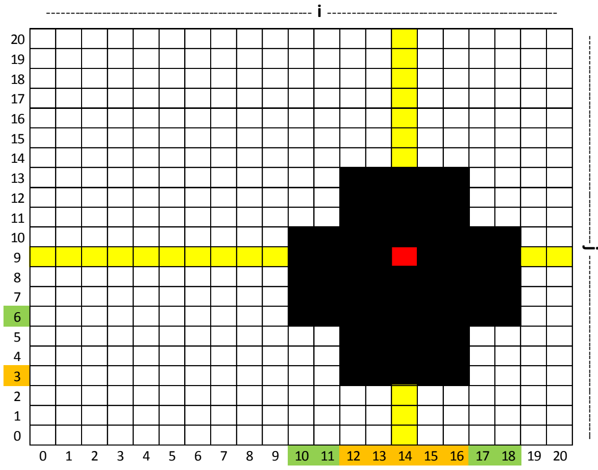

Figure 9. Ilustration of single BLOB symmetrical object in a 20x20 pixel frame

The method could be presented mathematically in a formula below:

Let:

CpH(i) = coordinate of black pixels horizontally
CpV(j) = coordinate of black pixels vertically
Cp = The number of "black" pixels in total
Hc = Centre point coordinate in horizontal axes (i)
Vc = Centre point coordinate in vertical axes (j)

Then:

$$Hc = \frac{\sum CpH(i)}{Cp} \text{ , where Hc is an integer value} \quad (4)$$

$$Vc = \frac{\sum CpV(j)}{Cp} \text{ , where Vc is an integer value} \quad (5)$$

An object illustrated on Fig. 4 has 1) CpH(12) to CpH(16) = 5 pixels with the total of 6 rows, 2) CpH(10) to CpH(18) = 9 pixels with the total of 5 rows, thus adds up to 75 pixels in total (Cp). The detail calculation is given in Table I.

TABLE I. DETAIL CALCULATION OF SINGLE BLOB SYMMETRICAL OBJECT

i	∑ CpH (i)	j	∑ CpV (j)	CP
3	70	10	40	5
4	70	11	40	5
5	70	12	40	5
6	126	13	88	9
7	126	14	88	9
8	126	15	88	9
9	126	16	88	9
10	126	17	88	9
11	70	18	40	5
12	70	19	40	5
13	70	20	40	5
Sum	1050	Sum	680	75
Hc	14	Vc	9.066666667	

Using formula 4 and formula 5, HC (Horizontal Centre) coordinate and VC (Vertical Centre) coordinate could be calculated shown below:

1) *Horizontal centre coordinate*

$$Hc = \frac{\sum CpH(i)}{Cp}$$

= 14

2) *Vertical centre coordinate*

$$Vc = \frac{\sum CpV(j)}{Cp}$$

= 9.06 (round down to 9)

A point is made up from two coordinates. These coordinates represent the horizontal position and vertical position in a spatial domain. Therefore, the centre point of the object is given by a horizontal and vertical coordinate which is P(Hc,Vc). The object centre point result is shown in Fig. 9 with red coloured pixel and the selected coordinate in i,j is highlighted in yellow colour.

C. *Centre Point Calculation for Single BLOB Non-symmetrical Object*

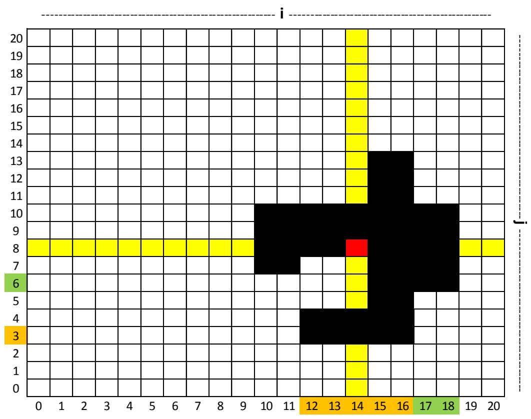

Figure 10. Ilustration of single BLOB non-symmetrical object in a 20x20 pixel frame.

A non-symmetrical object illustrated on Fig. 10 has 1) CpH(12) to CpH(16) = 5 pixels with the total of 6 rows, 2) CpH(10) to CpH(18) = 9 pixels with the total of 5 rows, thus adds up to 75 pixels in total (Cp). The detail calculation is given in Table II.

TABLE II. DETAIL CALCULATION OF SINGLE BLOB NON-SYMMETRICAL OBJECT

i	∑ CpH (i)	j	∑ CpV (j)	CP
3	70	10	34	5
4	70	11	34	5
5	31	12	34	2
6	66	13	34	4
7	87	14	34	6
8	126	15	88	9
9	126	16	88	9
10	126	17	40	9
11	31	18	40	2
12	31			2
13	31			2
Sum	795	Sum	426	55
Hc	14.4545455	Vc	7.745454545	

1) *Horizontal centre coordinate*

$$Hc = \frac{\sum CpH(i)}{Cp}$$

= 14.45 (round down to 14)

2) *Horizontal centre coordinate*

$$Vc = \frac{\sum CpV(j)}{Cp}$$

= 7.74 (round up to 8)

D. *Object Comparation*

For further discussion, on Fig. 11 is shown how the shape of the object is only affect the center point by 1 pixel off to the bottom of the object of the non-symmetrical object on the right, compared with the fully symmetrical object to the left. The result of the calculation for CpV (j) of symmetrical object is 9.06 and for the non-symmetrical object is 7.74.

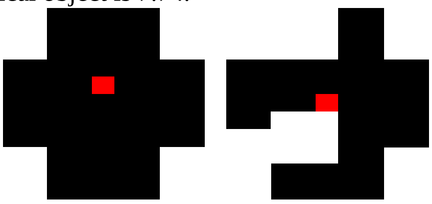

Figure 11. Object (symmetrical & non-symmetrical) comparison.

E. *Program Block of the Method on LabVIEW*

The function block of the program for centre point calculation is presented on Fig.12.

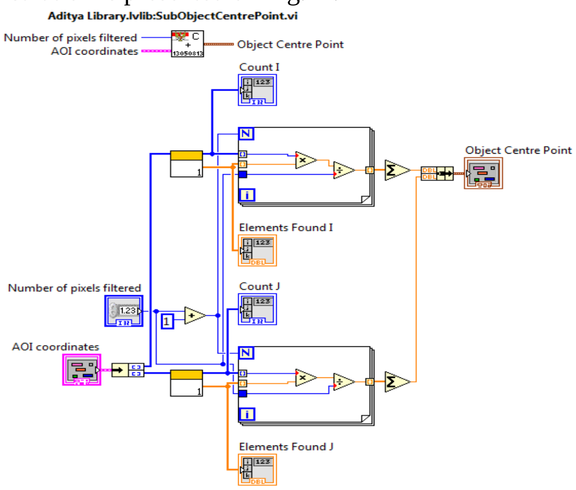

Figure 12. Program block using labview programming environment

The block of the program shown on Fig. 12 using Labview Programming environment consist of 5 blocks diagram which are :

1. AOI coordinates block
   This block is a block data taken from previous program block called "AOI" (Area Of Interest) which applying object recognition that has 3 dimensional array for index of data, horizontal coordinate pixel of the AOI image, and vertical coordinate pixel of the AOI image.
2. Number of pixel block
   This block is a block data taken from previous program block called "sum pixel filter" which calculate the sum of pixel that is filtered by the "AOI" block. This block data type is an integer.
3. HC (Horizontal Coordinate)
   This block calculate the HC using formula number (4). The output of the block is an integer value
4. VC (vertical Coordinate)
   This block calculate the VC using formula number (5). The output of the block is an integer value
5. Object Centre Point
   The final output of the program block is 2 integer value which is Hc and Vc that shows the centre point of an object in a target image.

The entire program block is then packed in 1 block called "centre point calculation" block utilizing 2 inputs (a single integer value and 3D array data type) producing 1 output (2 integer value) that could be used for further process.

V. CONCLUSION

a. The center object calculation depends on preprocessing method which could allocate and mark an object of interest
b. If the object marked has missing pixel in any part (e.g. because of uneven illumination) thus resulting a non-symmetrical BLOB, the center point will also change although the change is only 1 pixel off vertically illustrated Fig. 11.
c. An attempt to test this method using various scenario such as multiple BLOB, or scattered small object with noise should be made to further test the efficiency and accuracy.

ACKNOWLEDGMENT

The authors would like to express our great thanks to Ministry of Research and Higher Education of Indonesia (MRHE) / Direktorat Jenderal Penguatan Riset dan Pengembangan Kementrian Riset dan Pendidikan Tinggi Republik Indonesia (DRPM Kemenristekdikti) for fully funding this research.

CONFLICT OF INTEREST

The authors declare no conflict of interest

AUTHOR CONTRIBUTIONS

Aditya Kurniawan and Kholilatul Wardani conducted the research, simulating the algoritm, building the prototype of the robot and collecting the data needed ; Hendrawan and Eueung Mulyana support and guide the presentation flow on the paper, validate and process the data, all authors had approved the final version

REFERENCES

[1] M. R. Andersen, T. Jensen, P. Lisouski, A. K. Mortensen, M. K Hansen, T. Gregersen, P. Ahrendt, *Kinect Depth Sensor Evaluation for Computer Vision Applications*, 2012.

[2] A. Kurniawan and K. Wardani, "Kinect structural noise elimination technique for ITIS mobile robot data collector," *International Journal of Engineering & Technology*, vol. 7, pp. 1-5, November. 2018.

[3] K. Wardani, A. Kurniawan, E. Mulyana, and Hendrawan, "ROI based post image quality assessment technique on multiple localized filtering method on kinect sensor," in *Proc. International Conference on Telecommunication Systems, Services, and Applications (TSSA 12)*, IEEE Press, May. 2019, pp. 1-4.

[4] R. Gonzalez, R. Woods, *Digital Image Processing*, Pearson Prentice Hall, 2008.

[5] G. Bekey, G. Sukhatme, *The Path to Autonomous Robots*, New York:Springer, 2009.

[6] K. Castleman, *Digital Image Processing*, N.J:Prentice-Hall, 1979.

[7] J. Kramer, *Hacking the Kinect*, New York:Apress, 2012.

[8] M. Petrou, C. Petrou, *Image Processing*, United Kingdom:Wiley, 2010.

[9] J. St. Jean, *Kinect Hacks*, California:O'Reilly Media, 2012.

[10] J. Webb, J. Ashley, *Beginning Kinect Programming with the Microsoft Kinect SDK*, New York:Apress, 2012.

Aditya Kurniawan was born in Malang, Indonesia, in 1987. He completed the B.Eng. degree in mechatronics from Politeknik Elektronika Negeri Surabaya, Indonesia in 2009 and PGDip degree also in mechatronics from the University of Newcastle Upon Tyne, United Kingdom in 2014.

He started working as a Quality Assurance Engineer at Formulatrix Indonesia in 2010, testing microfluidic automation machine for medicine research application. In 2010 he started to became lecturer at Politeknik Kota Malang, Indonesia until recently. His current field of research includes robotics, vision sensing for robot and IoT.

Mr. Aditya Kurniawan is a member of IEEE, Indonesia section and currently undergoing various projects for South East Asia Minister Education Organization (SEAMEO).

Kholilatul Wardani was born in Malang, Indonesia, in 1988. She completed the B.Eng. degree in telecommunication from Politeknik Elektronika Negeri Surabaya, Indonesia in 2009 and M.Eng. degree in telematics from Institut Teknologi Bandung in 2014. In 2010 she started to became Junior Lecturer at Politeknik Kota Malang, Indonesia until recently. Her current field of research includes computer network, mobile network and IoT. Mrs. Kholilatul Wardani is a member of IEEE, Indonesia section and currently member of APPPI (Indonesian Association of Educators and Educational Developers)

Hendrawan was born in Bandung, Indonesia, in 1960. He received B.Eng. in Electrical Engineering from Institut Teknologi Bandung, Indonesia in 1985, M.Sc. in Telecommunication and Information Systems in 1990 and Ph.D in Electronic System Engineering in 1994 both from University of Essex, UK. He is Associate Professor at the School of Electrical Engineering and Informatics, Institut Teknologi Bandung. His current research includes Telecommunication Performance Engineering, Communucation Networks and Multimedia.

Eueung Mulyana was born in Bandung, Indonesia, in 1974. He completed Dr-Ing in communication networks from Technische Universitaet Hamburg Harburg, Germany in 2006. He is currently with the School of Electrical Engineering and Informatics at Institut Teknologi Bandung, Indonesia, as faculty member and as the Head of Telematics Laboratory. His current research includes cloud technology and software-defined infrastructure.