International Journal of Engineering & Technology, 7 (4.27) (2018) 1-5

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Kinect Structural Noise Elimination Technique For ITIS Mobile Robot Data Collector

Aditya Kurniawan[1]*, Kholilatul Wardani[2]

[1 2] Politeknik Kota Malang
**Corresponding author E-mail: aditya@poltekom.ac.id*

Abstract

A contemporary study on the Kinect sensor as a visual sensor device for robots shows that the sensor has some fundamental flaws. One of them are shadows on the object edge that will affect the process of recognition of shape (shape recognition) spatially. If the Kinect sensor is used in the robot vision navigation system, the sensor may lead to errors in the robot's decision on the shape of the object sensed by the sensor. The previous research reports a positive influence on the variation of smoothing process by using neighborhood filtering. This research will use multiple neighborhood localized filtering (MNLF) method to eliminate structural noise generated by kinect sensor IR camera. The robot model that will be used for testing is 6WD Wild Thumper Mobile Robot Chassis from Dagu Robotics. The calculation of SSI (Structural Similarity Index) calculation based on ROI between image index 0 (original image) with index 6 (image result after multiple filtering process) results SSI index with value 0.999999930515914. This indicates that multiple filtering processes do not affect the quality of images produced by Kinect sensors. The number of 0.99 can be rounded to 1 so that the conclusion based on ROI image assessment shows no differences on image quality after process.

Keywords: *Kinect sensor; Structural noise; Multiple isolated filtering.*

1. Introduction

Autonomous machinery (robot) has evolved into the area of an artificial intelligence machine lately. This can be seen from the trend of researches and machine functions that are drawn towards the human-like ability that can think, aid or assisting other human. Autonomous machines require certain algorithms (methods of thinking) to be able to act in a certain way, so that it will be called a smart machine. In addition to the algorithm, a set of sensing mechanism is also needed to turn the dead machine into a system that is aware of the environment in which it operates. Both systems (sensing mechanisms and algorithms) must work in such a way that they can work independently without human control.

At present, the need for autonomous electromechanical systems (usually called robots) has significantly increased. The term autonom means a machine capable of performing tasks in the real world without explicit human control [1]. Previous research has attempted to develop the function of mobile robots as human assistants who can move autonomously without human help [2 ~ 6]. These study focused on the use of integrated sensors that can provide comprehensive information visually, so that mobile robots have a "consciousness" of position and orientation.

A contemporary study by Anderson [2] discusses vision sensors called Kinect as a sensory device for robots. Kinect is a vision sensor consisting of a visible light camera and an infrared array sensor [7]. A visible light camera can capture light ranging from 390 nm to 700 nm (visible to humans). By measuring the various parameters depending on how the algorithm is applied, a camera can give the robot a two dimensional vision. However, this kind of camera is not able to measure the distance between the object and the sensor. Therefore, the Kinect is equipped with an additional sensor in the form of infrared arrays to measure the distance between objects (obstacles) and robot on a certain spatial field. Thus, these sensors can provide the robot distance information (depth perception) and simultaneous two-dimensional vision to the robot.

Andersen reported that Kinect has structural noise in its depth image, therefore the measurements of the object size spatially is increasing along with the distance between sensor and object. Recent research by Kurniawan reports that the effect of Isolated Neighborhood Averaging Mask (INAM) on the number of pixel noise generated by Kinect has a positive effect of eliminating the noise [3]. The cause-effect relationship between the two (INAM vs Noise) has Pearson coefficient of 0.4, and the saturation layer of Neighborhood-Averaging filter on the pixel noise generated by the Kinect is 2 layer.

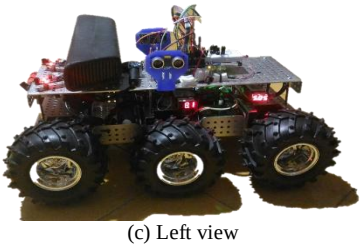
2. Robot Model Used

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 robot model is shown in Figure 1. 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.

(a) Isometric view

Copyright © 2018 Aditya Kurniawan et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

(b) Right view



(c) Left view

Fig.1: Robot model 6WD wild thumper from multiple views

3. Multiple Isolated Filtering

Multiple localized filtering is a masking applied on a layer per layer, a mean masking (average) 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.

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 (Gonzalez, 2008).

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

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

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

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

4. ROI Image Quality Assessment

ROI (Region of Interest) Image Quality Assessment is an image quality assessment model based on the SSI (Structural Similarity Index) used in certain regions of the image that is desired to be assessed. This assessment model is more valid because in the non-ROI based image assessment, the quality index that is obtained will be confused with other areas that are actually beyond the researcher's interest. (for example: to assess the quality of structural noise that incidentally occurs only on the edge of the target object)

The scoring mechanism diagram using SSIM can be seen in the Figure 2.

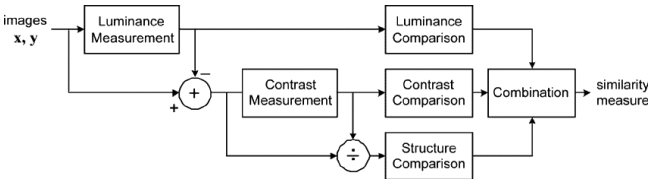


Fig. 2: ROI based quality model scoring mechanism

The algorithm used to perform the assessment based on the SSIM index is as below

$$SSIM_{(x,y)} = \frac{(2\mu_x\mu_y + (k1.L)^2)(2\sigma_{xy} + (k2.L)^2)}{(\mu_x^2\,\mu_y^2 + (k1.L)^2)(\sigma_x^2 + \sigma_y^2 + \sigma_z^2)} \quad (3)$$

By calculating the value of SSIM between two images, the quality index could be determined using the Table 1.

Table 1: ROI based image quality index meaning

Index	Meaning
1	Identical
-1	NOT Identical

5. Software Data Logger

Data logger software that is used is Visual Basic 6.0. The screenshot of the program is shown in figure 3.

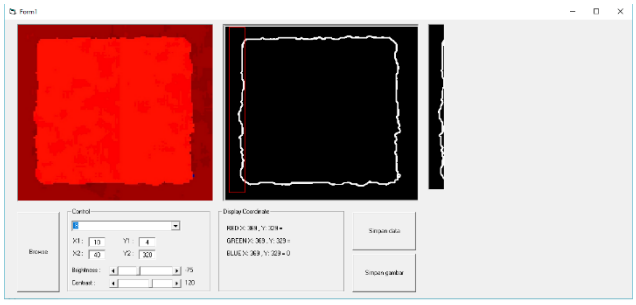


Fig. 3: Screen capture data logger software using Visual Basic 6.0

6. Multiple Isolated Filtering Process

The image data obtained from Kinect sensors shows that the edge of the object has a noise that causes errors on a spatial measurement of the shape or size of the object as shown in the graph of color element data, in the form of 2 dimensions (Refer Figure 4).

Fig. 4: Image capture Kinect sensor and edge ROI

The algorithm consist of 7 steps and 8 index data stored in a 2D arrays for each colour data element (R,G,B). The list of process

name and 2D arrays index image data where the image value is stored is shown in table 2

Table 2: Algorithm process and index data list

No	Process Name
1	Original image data
2	Brightness enhancement
3	Contrast enhancement
4	Edge detection sobel 3x3
5	Binary (edge) isolation
6	Multiple smoothing custom matrix 3x3 Vertical
7	Multiple smoothing custom matrix 3x3 Horizontal
8	Edge detection sobel 3x3
9	Binary (edge) result

7. Pre Processing Program

The Software used on robot side is Labview 7.0. The first step of the program is to capture image. In this process, it requires a driver application (Microsoft Kinect SDK) that allows Labview program to connect with Kinect sensor. The RGB camera is a camera that captures visible light and converts it to a set of 24-bit values that represent colors in RGB format. This camera will be used to distinguish colors and localize segmented objects for further processing. Because of the characteristics of raw data stored in digital format, a pre-processing algorithm is required after the capturing of this image.

Before taking further steps, sub VI of SubKinectInit.VI was created to simplify block diagrams on labview programming. sub VI consists of a setting index for RGB image format, settings for image depth format, an "InitializeKinect" block VI, and a zero constant (for selecting the Kinect index of number 0). The program block VI is shown in Figure 5.

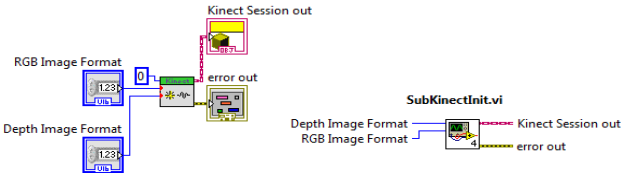


Fig. 5: SubKinectInit VI

The inputs of this sub-block are index of RGB format, and index of depth image format. The output of this block is Kinect session out, and error out. Index format for both image formats is presented below:

1. 0: default
2. 1: undefined
3. 2: RGB resolution 640 x 480 at 30 FPS
4. 3: RGB resolution 1280 x 960 at 12 FPS
5. 4: YUV resolution 640 x 480 at 15 FPS
6. 5: Raw YUV resolution 640 x 480 at 15 FPS

To get the RGB value data from the Kinect camera, the block function "GetColorFrame" in block VI which is already in the labview program will be modified.The modified program allows block VI to extract raw data out of Kinect then convert it to RGB value format. Standard block from "GetColorFrame" block VI has two inputs, Kinect session and error in and three outputs which are Kinect session out, color frame and error out Configuration minimum to retrieve data from RGB camera is presented below. Index format setting will be set automatically set to index value 0 if not defined. The output from SubKinect VI is shown in Figure 6.

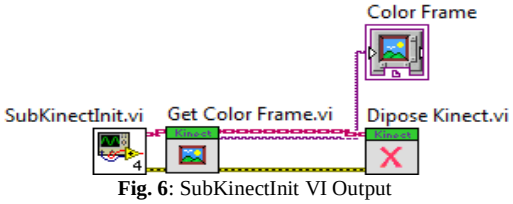


Fig. 6: SubKinectInit VI Output

The "GetColorFrame" modification is to add three more terminal outputs on block VI which are the red (R), green (G), and blue (B) values. the data coming out of this block is a 2D array that stores the RGB value of the captured image.

The calculations used for color extraction are based on the principle of digital color storage format. Therefore, for each Red element value is a multiplication of 2 ^ 0, each Green value is a multiplication of 2 ^ 8, and the Blue value is a multiplication of 2 ^ 16. The modulus method will be used to extract raw data into RGB format.

Let :

$P_{raw(i,j)}$: raw data value on row *i* column *j*

$R_{(i,j)}$: red element pixel value

$G_{(i,j)}$: green element pixel value

$B_{(i,j)}$: blue element pixel value

$P_{0(i,j)}$: RGB colour on row *i* column *j*

Then

$P_{0(i,j)} = (R_{(i,j)}, G_{(i,j)}, B_{(i,j)})$

$$P_{o(i,j)} = ((P_{(i,j)} \bmod 2^{16}) \bmod 2^{8}, \frac{P_{(i,j)} \bmod 2^{16}}{2^{8}}, \frac{P_{(i,j)}}{2^{16}}) \quad (4)$$

The result of modification of block VI for RGB extraction format is as shown in figure 7.

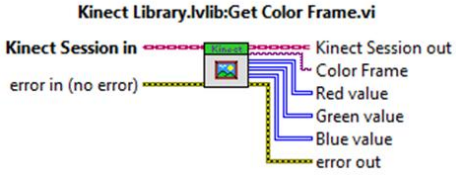


Fig. 7: KinectSession IN and OUT VI

The value obtained after color extraction is a variable in a 2D array that has 640 columns and 480 rows. To display the value after this extraction, a block VI plotter of RGB as a plotter program in Labview is created. The program construction on this block can be seen in Figure 8.

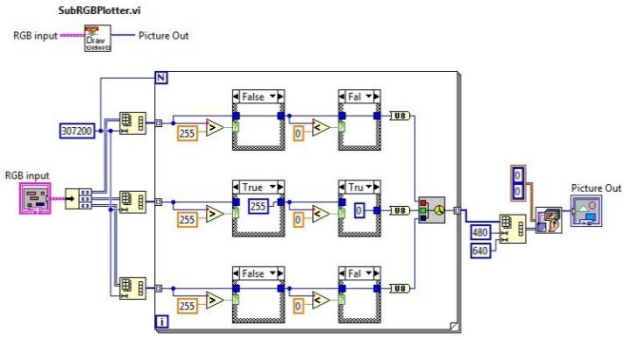


Fig. 8: SubRGB Plotter VI library

The values for each of each RGB element can be shown in the equation below.

1. For the red color element has the function:
   $R_{(i,j)} : (P_{raw(i,j)} \ mod \ 2^{16}) \ mod \ 2^{8}$
2. For the green color element has the function:
   $G_{(i,j)} : \frac{(P_{raw(i,j)} \ mod \ 2^{16})}{2^{8}}$
3. For the blue element has a function:
   $B_{(i,j)} : \frac{(P_{raw(i,j)})}{2^{16}}$

RGB color results will be processed further using a brightness enhancement to increase or decrease all pixel value and a contrast process to increase the difference of one particular pixel with its neighbor pixel. the contrast process (contrast enhancement) is the pixel value multiplication operation process to increase / decrease

the difference between pixels. This function only increases the pixel value gap without changing the RGB element structure
Let :
$P_{raw(i,j)}$: raw data on row *i* column *j*
$P_{CE(i,j)}$: pixel value after contrast enhancement
β : contrast constant
α : brightness constant

Then

$$P_{CE(i,j)} = ((((P_{(i,j)} \bmod 2^{16}) \bmod 2^{8}) + a)b, ((\frac{P_{(i,j)} \bmod 2^{16}}{2^{8}}) + a)b, ((\frac{P_{(i,j)}}{2^{16}}) + a)b \quad (5)$$

To apply the above function, a block program "SubContrast" is used. The program block is shown in the Figure 9.

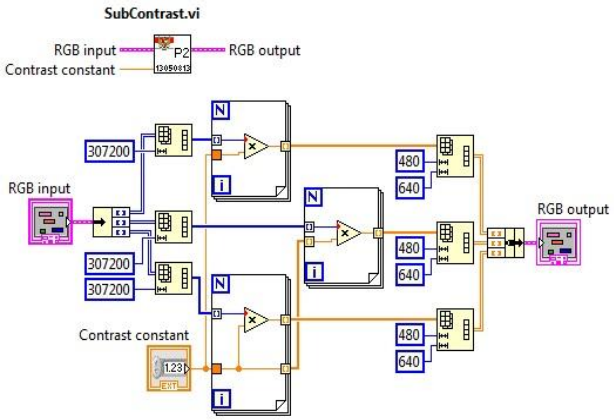


Fig. 9: SubContrast VI library

8. Result (Image Algorithm)

This stage is the core of the entire image algorithm built. Multiple smoothing filter is a noise-removing filter that is used only on the area defined by the isolated filter so that the smoothing process will not affect another image area that does not require this process. Multiple smoothing using 3x3 matrix consisting of vertical and horizontal matrix having multiplication value 20. The image result and data graph of this process is shown on Figure 10.

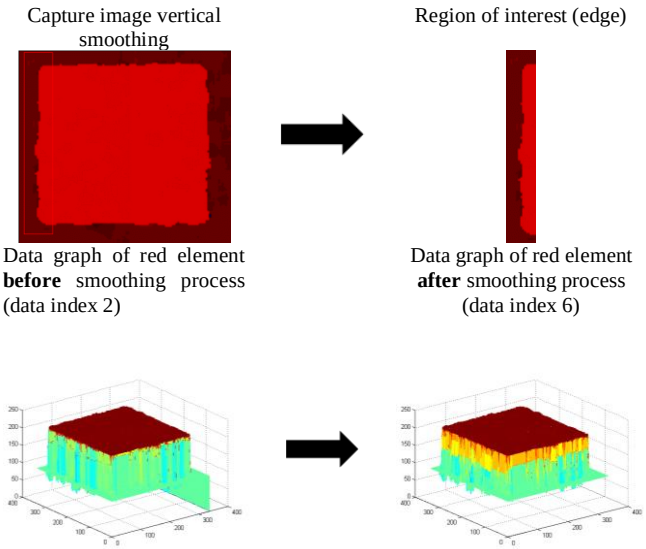


Fig. 10: Image capture and R element 3D data graph showing before and after smooting process

Smoothing process eliminates the noise on the edge of the object by calculating all pixels in the matrix range and dividing it to get the average value. In this way, pixel value that is different from the edge line will blend with the border, assuming that the different values are noise on edge. The contour graphs are shown in Figure 11.

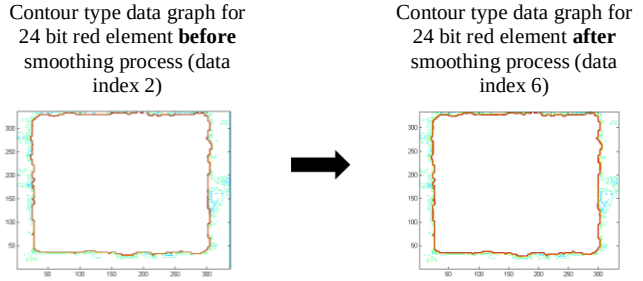


Fig. 11: R element contour type data graph showing before and after smooting process

9. Conclusion

The ROI analysis based image assessment is a SSI (Structural Similarity Index) calculation used to measure the quality of an image after multiple smoothing by comparing the images before and after the process. In this case, the index data used is index data 0 and index 6
Let:
μ_0 = average of image index 0
μ_6 = average of image index 6
σ_0^2 = variance of image index 0
σ_6^2 = variance of image index 6
σ_{06} = covariance of image index 0 and 6
k1 = constant 0,01 and k2=0,03
$L = 2^{\text{bit per pixel}} - 1$
Then

$$SSIM_{(0,6)} = \frac{(2\mu_0\mu_6 + (k1.L)^2)(2\sigma_{06} + (k2.L)^2)}{(\mu_0^2{}_2 \mu_6^2{}_2 + (k1.L)^2)(\sigma_0^2 + \sigma_6^2 + c2)} \quad (6)$$

The result of this process (index 0) to (index 6) is shown on Figure 12.

Fig. 12 : Image capture showing before and after process 0 to 6

The calculation for above image is shown below:
$SSIM_{(0,6)} = \underline{7.13055441317085^{21}} = 0.999999939515914$
7.13055490863094^{21}
The results of this research shows that the SSI value of 0.99 for the luminance, contrast, and structural section of the edge region of the specimen. The value indicates that the Multiple Localized Filtering Technique applied to the noise generated by the Kinect sensor has no effect on the image quality generated by the sensor. Both images are identical with the index value of 0.99. These are some of the suggestion for the future research work:
a) Multiple filtering processes takes long execution time due to the repetition process, therefore it would be best to limit the iteration up to 2
b) Multiple filtering in order to eliminate noise on the Kinect structural noise needs to be processed further to determine a straight line of the object edges by calculating the average point of the edge pixel position
In near future, by using the proposed method of Kinect structural noise elimination technique, it is expected that the usage can be

extended especially in area of robot vision, especially for visual robot navigation.

Acknowledgement

The authors would like to express our great thanks to Ministry of Research and Higher Education of Indonesia (MRHE) / DRPM Dikti for funding this research.

References

[1] G.A. Bekey, *Autonomous Robots: From Biological Inspiration to Implementation and Control*, MIT Press, (2005)
[2] Andersen MR, Jensen T, Lisouski P, Mortensen A K, Hansen MK, Gregersen T, Ahrendt P, *Kinect depth sensor evaluation for computer vision applications*, Aarhus University, (2012).
[3] Kurniawan A, Wardani K, "Pengaruh Isolated Neighborhood-Averaging Filters Pada Kinect Structural Noise Sebagai Sistem Navigasi Robot Wild Thumper", *Telekomunikasi Elektronika, Komputasi dan Kontrol*, Vol 3 no 1 (2017) , pp : 49-56.
[4] Kurniawan A, Wardani K, "Saturated Iteration of Neighborhood Averaging Filter Algorithm For Kinect Structural Noise Eliminator For Wild Thumper.", *Game Technology, Information System, Computer Network, Computing, Electronics, and Control* , Vol 3 no 3 (2018).
[5] Wardani K, Kurniawan A, "ROI Based Post Image Quality Assessment Technique on Multiple Localized Filtering Method On Kinect HD Sensor Mobile Robot.", *Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control* , Vol 3 No 2, (2018).
[6] Zhang H, Yan R, Zhou W, Sheng L, "Binocular Vision Sensor (Kinect)-Based Pedestrian Following Mobile Robot", *Applied Mechanics and Materials*, pp.1326-1329, (2014)
[7] Gonzalez R, Wintz, P, *Digital image processing*, Addison-Wesley, (1987)
[8] Horn B, *Robot vision*. MIT Press, (1986)
[9] Matsuda T, *Robot vision*, Nova Science Publishers, (2009)
[10] Pauli J, *Learning-based robot vision*, Springer, (2001)
[11] Sood A, Wechsler H, *Active perception and robot vision*, Springer Verlag, (1992)
[12] Yang W, "Method of image quality assessment based on region of interest", *Journal of Computer Applications*, Vol 28 No 5, (2008), pp.1310-1312