E-ISSN 2656-8039
INTERNATIONAL JOURNAL OF
ENVIRONMENT, ENGINEERING & EDUCATION

www.ijeedu.com
Vol. 1, No. 2, 2019
Research Article

Homogeneous Image Compression Techniques with the Shannon-Fano Algorithm

Fathahillah, Satria Gunawan Zain, Rismawati

Department of Electrical Engineering, Universitas Negeri Makassar, Makassar 90222, Indonesia
*Contact email: khafath@gmail.com, satria.gunawan.zain@unm.ac.id, rismajii2@gmail.com

Received: *June 28, 2019;* ***Accepted:*** *August 15, 2019;* ***Published:*** *August 28, 2019*

Abstract: Compression is a field that needed at this time where increasingly high-tech digital and computing makes it possible to process data in a size that is large enough like multimedia. Compression needed to keep pressing the storage media consumption of data and information stored on computer media. The Shannon-Fano compression algorithm is one of the well-known compression algorithms and is useful in saving data storage space. Shannon-Fano compression algorithms can be performed on text and digital images. In this study applying the Shannon-Fano method to homogeneous digital images using applications created with the MATLAB Program. The product of this research will be tested using several images taken using a webcam. The product tested to find out whether the coding used is correct or has errors. The product will be successful if it can reduce the size of the compressed image from the original image. Homogeneous images that have tested using the Shannon-Fano image compression application have an average compression ratio value of 52%. The size of the compression results reaches half of the original image. The Shannon-Fano compression algorithm detects the number of times that characters appear in each experiment, then the coding of the frequency of characters appearing in binary numbers.

Keywords: Compression, Decompression, Image Size, Lossless Compression Technique, Resolution Image

1. Introduction

Along with the development of large-capacity storage media, people no longer encounter problems if they have large files. The more so if the file we have is an image file. However, sometimes the large file size is annoying if we must manage the storage media that we have for various data [1]. Especially if we will send the file electronically, of course the file capacity becomes a problem. Images are vital and become an integral part of daily life. In specific interests, the image (picture) used as a tool to express reasoning, interpretation, illustration, representations, memorization, education, communication, evaluation, navigation, surveying, and entertainment [2].

The development of camera technology has helped humans in many ways, with the advantages provided. Currently the camera not only used to capture the moment but the current camera can also use as a safety tool [3]. This type of camera commonly referred to as a security camera. Security cameras are cameras that use for surveillance purposes in public places that considered vulnerable and dangerous such as factories, schools, supermarkets, offices, warehouses. To minimize the risk of crime and threats [4]. The results of the security camera are not only in the form of images with JPG format as usual cameras, but there are also in the form of video with AVI format. The great benefits of cameras and video besides used as a security system can also use as documentation and legal evidence [5].

The large file size is a drawback of the current camera sophistication. As a result, it takes a more extensive storage media to store an image/frame. The use of more extensive data storage media has implications for the use of higher costs. To reduce the use of more substantial costs, the file size of an image/frame should be smaller [6].

Citation this article: Fathahillah, S. G. Zain, Rismawati, "Homogeneous Image Compression Techniques with the Shannon-Fano Algorithm," *Int. J. Environ. Eng. Educ.*, vol. 1, no. 2, pp. 59–66, 2019.
DOI: https://doi.org/10.5281/zenodo.3490205

Reducing the size of image/frame or video files can be done by three things, namely, changing the number of frames per second, saving images/frames or videos when objects detected and compressing images/video frames. Changes in the number of frames per second have done to change the size of video files. The results of CCTV security camera footage will look like a video with a disjointed object motion (not smooth). In the second method, the picture/frame begins to recorded after there is a moving object, causing a change in the intensity of the image at a location. Then, the third method uses the image or video compression method. Compression Algorithm is a data compression technique in order to obtain a file with a smaller size than the original size or the process of converting a set of data into a form of code to save storage space requirements and time for data transmission [7]–[9].

To overcome the large image size, namely by making an image compression application using data compression methods. The compression method divided into two groups, namely: Static methods that use code maps that are always the same. This method requires two phases (two-pass): the first phase to calculate the probability of the appearance of each symbol/character and determine the code map and the second phase to convert the message into a set of code to transmit. For example in static Huffman, arithmetic coding such as Shannon-Fano coding [10], [11], static/dynamic/adaptive Huffman coding [12]–[15] for example in LZW and DMC algorithms and arithmetic codes [16], [17]. And then the compression technique by replacing the character/fragment in the input file with the location index of the character/fragment in a dictionary such as the Lemp-Zif-Welch (LZW) data compression technique [18]–[20].

Data compression usually obtained by making an original symbol in the data with symbols that are smaller in size. Symbols can be any character, word, phrase, or another unit that can be stored in a symbol dictionary [21]. Image compression using the Shannon-Fano method has widely studied and implemented, as done by Shanmugasundaram and Lourdusamy [22]; Kodituwakku and Amarasinghe [23]; Vaidya, Walia, and Gupta [24]. Research on the application of the Shannon-Fano method to images that have been determined at the color depth level still not studied.

2. Literature Review

Digital image processing refers to processing 2-dimensional images using a computer. In a broader context, digital image processing refers to the processing of any 2-dimensional data. Digital images are arrays that contain real or complex values represented by a particular sequence. An image can define as a function f (x, y) measuring M rows and N columns, where x and y are spatial coordinates, and the amplitude f at the coordinate point (x, y) called the intensity or gray level of the image at that point. If the value of x, y and amplitude value f limited and discrete, it can say that the image is a digital image. Values on a slice between rows and columns (at position x, y) called picture elements, image elements, and pixels. The last term (pixel) most often used in digital images. The formation of digital images (discrete) through several stages, namely Image Acquisition, Sampling, Quantization.

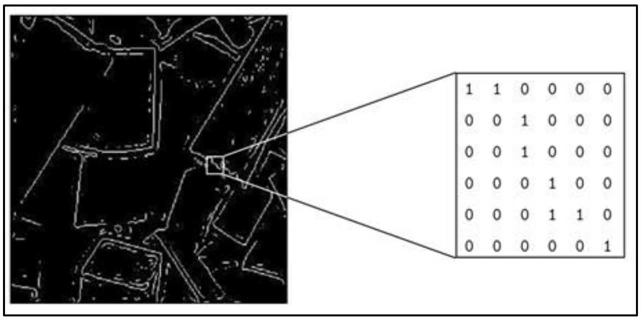
The image resolution is the level of detail of an image. The higher the image resolution, the higher the level of detail of the image. The unit in measuring image resolution can be a physical size (number of lines per mm/number of lines per inch), or it can also be a total image size (number of lines per image height). The resolution of an image can measure in various ways as follows:

a. Pixel resolution
   Pixel resolution is a calculation of the number of pixels in a digital image. Another understanding of image resolution is the result of multiplying the number of pixels in width and height, then divided by 1 million. This type of pixel resolution often found in digital cameras. An image that has a width of 2,048 pixels and a height of 1,536 pixels will have a total pixel of 2,048 × 1,536 = 3,145,728 pixels or 3.1 megapixels.
b. Spatial Resolution
   Spatial resolution shows how close each line is to the image. The distance depends on the system that created the image. Spatial resolution produces the number of pixels per unit length.
c. Spectrum Resolution
   A digital image distinguishes intensity into several spectra. Multi-spectrum images will provide a better spectrum or wavelength that will use to display colors.
d. Temporal Resolution
   Temporal resolution is related to video. A video is a collection of static frames in the form of sequential images and displayed quickly. The temporal resolution gives the number of frames that can display every second in units of frames per second (fps).
e. Radiometric Resolution
   This resolution provides the value or level of smoothness of the image that can display and usually displayed in units of examples 8-bit images and 256-bit images. The higher the radiometric resolution, the more significant the difference in intensity displayed.

2.1. Image

The value of a pixel has a value in a specific range, from the minimum value to the maximum value. The range used varies depending on the type of color. However, in general, the range is 0 - 255. Here are the types of images based on pixel values:

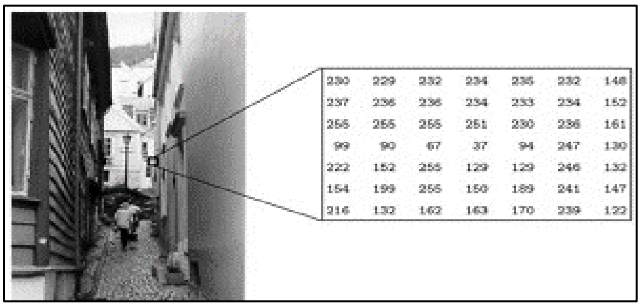
a. Binary images are digital images that have only two possible pixel values, namely black and white. Binary imagery also called B & W (black and white) or monochrome image. Only 1-bit needed to represent the value of each pixel of a binary image, as seen in Figure 1.



1	1	0	0	0	0
0	0	1	0	0	0
0	0	1	0	0	0
0	0	0	1	0	0
0	0	0	1	1	0
0	0	0	0	0	1

Figure 1. Binary Digital Image

b. Grayscale images are digital images that have only one channel value in each pixel; in other words, the value of Red = Green = Blue. Each pixel represents a degree of gray with values between 0 - 255, and this means that each pixel has an 8-bit size or 1 byte. This value used to indicate the level of intensity. Colors that owned are colors from black, gray and white. The gray level in question is the color of ash with various levels from black to nearly white, as seen in Figure 2.



230	229	232	234	235	232	148
237	236	236	234	233	234	152
255	255	255	251	230	236	161
99	90	67	37	94	247	130
222	152	255	129	129	246	132
154	199	255	150	189	241	147
216	132	162	163	170	239	122

Figure 2. Grayscale Digital Image

c. 8-bit image, there are two types of an 8-bit color image. First, an 8-bit color image using a 256-color palette with each palette having a specific RGB colormap mapping. This model used more often. Second, each pixel has an 8-bit format.

d. 16-bit color images (commonly referred to as high-color images) with each pixel represented by 2-byte memory (16 bits). In its bit formation, the red and blue values take place on the 5-bit right and left. The green component has 5-bits plus 1-bit extra. The selection of green components with a 6-bit sequence because human vision is more sensitive to green color.

e. Each pixel of a 24-bit color image represented by a 24-bit number making a total of 16,777,216 color variations. This variation is more than enough to visualize all the colors that human vision can see. Human vision believed only to be able to distinguish up to 10 million colors. Each RGB pixel information point stored in 1-byte data. The first 8-bits store the blue value, then followed by the green value in the second 8-bit, and the last 8-bits are red.

3. Research Methods

3.1. Research Approach

The research used is experimental. Experimental research is the most appropriate research and can genuinely test hypotheses about the cause and effect of a relationship when compared with other research methods. The process of making an image compression application using the Shannon-Fano algorithm will explained in this design section. From the determination of components to the stages of completion.

3.2. Designing Tools

MATLAB programming language will use in this study to display image data (images). It due to the image data basically in the form of a color matrix with a certain degree of gray (grayscale). MATLAB is a program for analysis and numerical computation and is a high-level programming language specifically for technical computing. MATLAB provides an interactive system that uses the concept of arrays/matrices as standard variable elements without the need to declare arrays as in other languages.

3.3. Product Trial

The product of this research will be tested using several images taken using a webcam. The product tested to find out whether the coding used is correct or has errors. Some images will take using a webcam, then the image from the webcam taker will compress using an application that made, then the size will be compared between the original image with the compressed image from the product made. The product will be successful if it can reduce the size of the compressed image from the original image.

3.4. Shannon-Fano Encoding Procedure

The Shannon-Fano method based on variable length-words, which means that some symbols in the data represented by symbol codes that are shorter than the symbols in the data. The higher the probability, the shorter the symbol code. In estimating the length of each symbol code, the Shannon-Fano method produces symbol codes that are not the same length, so the codes are unique and can decode.

The procedure in Shannon-Fano encoding is:

- Arrange the probability of the symbol from the highest source to the lowest.
- Divide into two equal parts and give a value of 0 for the top and 1 for the bottom.
- Repeat step two, each division with the same probability until it cannot divide again.
- Encoding each original symbol from the source into a binary sequence that generated by each of these sharing processes

The Shannon-Fano Algorithm Flow Chart in research can see in Figure 3 as follows:

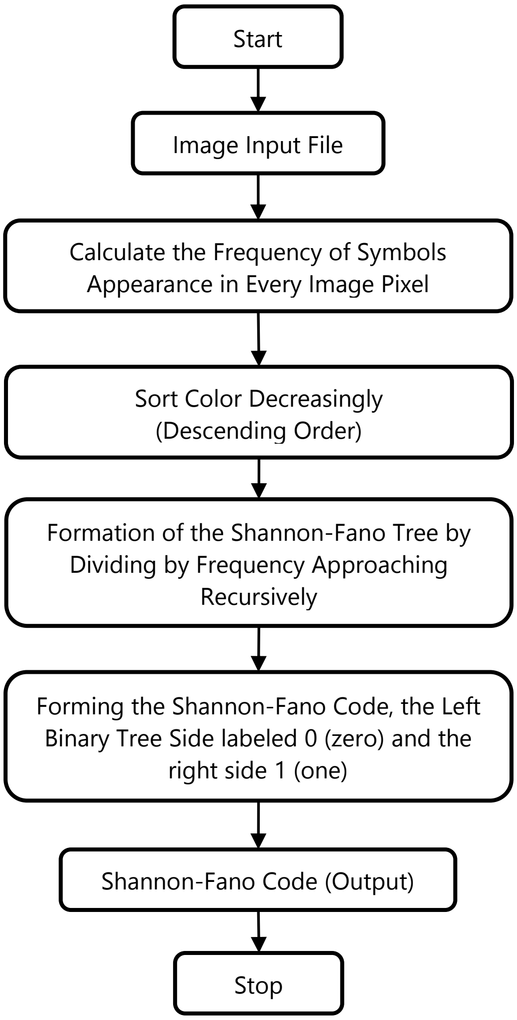

Start
Image Input File
Calculate the Frequency of Symbols Appearance in Every Image Pixel
Sort Color Decreasingly (Descending Order)
Formation of the Shannon-Fano Tree by Dividing by Frequency Approaching Recursively
Forming the Shannon-Fano Code, the Left Binary Tree Side labeled 0 (zero) and the right side 1 (one)
Shannon-Fano Code (Output)
Stop

Figure 3. Shannon-Fano Algorithm Flowchart.

The Compression System Flow Chart in research can see in Figure 4 as follows:

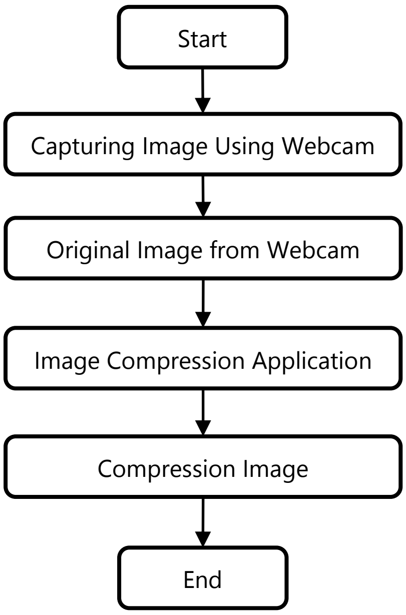

Start
Capturing Image Using Webcam
Original Image from Webcam
Image Compression Application
Compression Image
End

Figure 4. Compression System Flowchart.

The Decompression System Flow Chart in research can see in Figure 5 as follows:

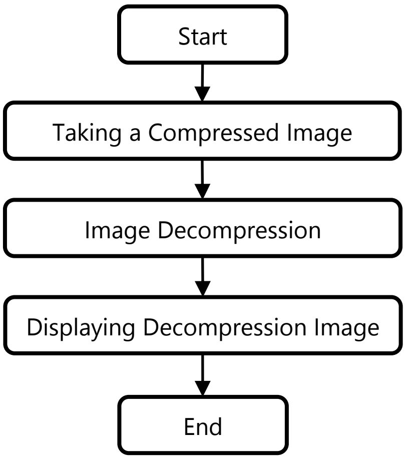
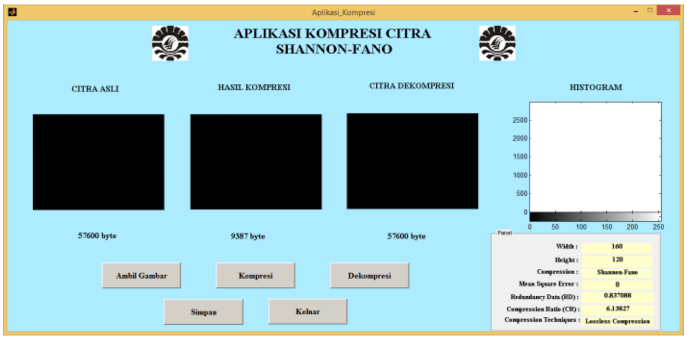


Figure 5. Decompression System Flowchart.

4. Results and Discussions

4.1. Image Compression Application Testing

The research results generated in the form of an application developed using MATLAB programming software with its pseudocode to form an image compression application using the Shannon-Fano algorithm. Figures 6 to 10 represent the GUI (Graphic User Interface) image compression application using the Shannon-Fano algorithm. Display the program after it has run by using several homogeneous images as follows:


Aplikasi_Kompresi

APLIKASI KOMPRESI CITRA
SHANNON-FANO

CITRA ASLI

HASIL KOMPRESI

CITRA DEKOMPRESI

57600 byte

9387 byte

57600 byte

HISTOGRAM

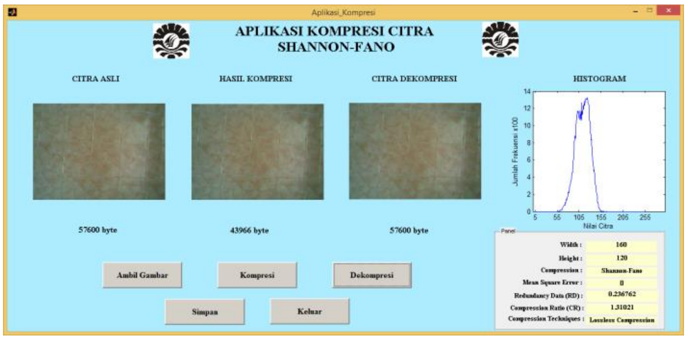
Panel

Width : 160
Height : 120
Compression : Shannon-Fano
Mean Square Error : 0
Redundancy Data (RD) : 0.837088
Compression Ratio (CR) : 6.13827
Compression Techniques : Lossless Compression

Figure 6. Results Display Image 1

The image contained in Figure 6 included in the category of images that are homogeneous because it has a low color intensity variation that is plain black. Can see in the histogram graph, image 1 is too dark because the intensity of the image gathered at a low gray level value that is 0. The histogram of an image that is too dark usually tends to gather at a low gray level (towards the value of 0).

The captured image is an RGB image with a size of 160×120. The image then compressed using the Shannon-Fano algorithm and produces a compression ratio of 6.13827 with redundancy data of about 0.837 (84%). The lower the color intensity of the compressed image, the higher the amount of redundant data that can generate. This will also affect the size of the compressed image which will be smaller than the original image. The difference in the length of the bytes between the original image and the resulting image in Figure 6 is huge at 48213 bytes. The image said to successfully compress if the measuring criteria are 0 (zero) because the Shannon-Fano algorithm uses a lossless compression technique. Figure 6 said to compress successfully because it gets the Mean Square Error value of 0 (zero).



57600 byte

43966 byte

57600 byte

Width : 160

Height : 120

Compression : Shannon-Fano

Mean Square Error : 0

Redundancy Data (RD) : 0.236762

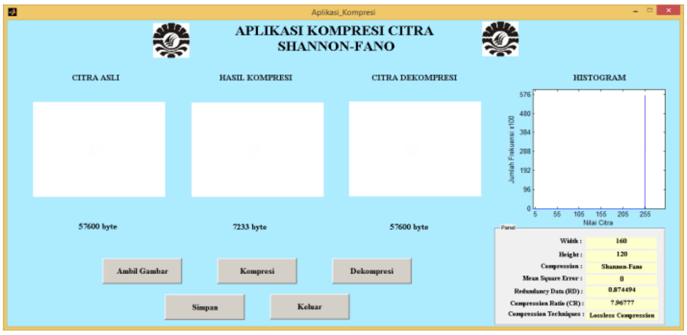
Compression Ratio (CR) : 1.31021

Compression Techniques : Lossless Compression

Figure 7. Results Display Image 2

The image contained in Figure 7 also included in the category of images that are homogeneous because it has a low color intensity variation. Can see in the histogram graph, image 2 is not too dark and too bright because the intensity of the image gathered in the range of values 55-155, although it can conclude that the image is more directed to a low gray level.

The captured image is an RGB image with a size of 160×120. The image then compressed using the Shannon-Fano algorithm and produces a compression ratio of 1.31027 with redundancy data of around 0.236 (24%). Excessive data generated in Figure 7 is lower than redundant data in Figure 6 because the captured image has more color intensity compared to the previous image in Figure 7. The difference can see in the histogram where the histogram in Figure 6 converges to only one value point which is 0 whereas in Figure 7 the histogram is gathered around 55-155. The difference in length of bytes between the original image and the image compression results in figure 7, which is 13634 bytes.



57600 byte

7233 byte

57600 byte

Width : 160

Height : 120

Compression : Shannon-Fano

Mean Square Error : 0

Redundancy Data (RD) : 0.874494

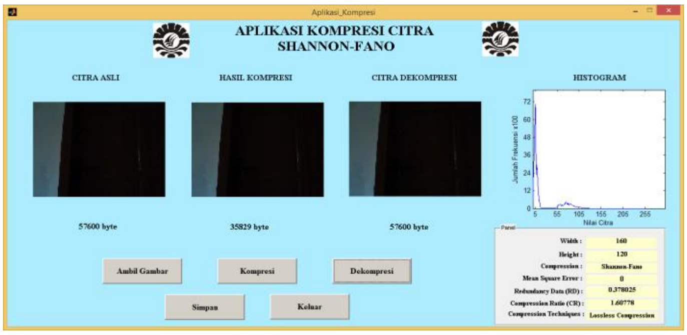
Compression Ratio (CR) : 7.96777

Compression Techniques : Lossless Compression

Figure 8. Results Display Image 3

The image contained in Figure 8 also included in the category of images that are homogeneous because it has a low color intensity variation that is plain white. Can see in the histogram graph, image 3 is too bright because the intensity of the image gathered at a high gray level value which is a value of 255. A histogram of an image that is too bright usually tends to collect at a high gray level value (towards the value of 255).

The captured image is an RGB image with a size of 160×120. The image then compressed using the Shannon-Fano algorithm and produces a compression ratio of 7.96777 with redundancy data of around 0.874 (87%). The lower the color intensity of the compressed image, the higher the amount of redundant data that can generate. This will also affect the size of the compressed image which will be smaller than the original image. The difference in the length of bytes between the original image and the image resulting from compression in figure 8 is huge namely 50367 bytes.



57600 byte

35829 byte

57600 byte

Width : 160

Height : 120

Compression : Shannon-Fano

Mean Square Error : 0

Redundancy Data (RD) : 0.378025

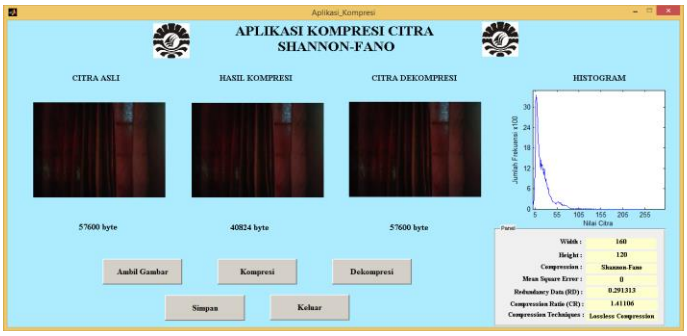
Compression Ratio (CR) : 1.60778

Compression Techniques : Lossless Compression

Figure 9. Results Display Image 4

The image contained in Figure 9 also included in the category of images that are homogeneous because they have a low color intensity variation. Can see in the histogram graph, image 4, the intensity of the image is more directed to the value of 0 and has a higher number of frequencies at the value of image 0.

The captured image is an RGB image with a size of 160×120. The image then compressed using the Shannon-Fano algorithm and produces a compression ratio of 1.60778 with redundancy data of around 0.378 (38%). Excessive data generated in Figure 4.5 is lower than redundant data in Figures 6 and 8 because the captured image has more color intensity brown, white, and looks dark. The difference in length of bytes between the original image and the image compression results in Figure 9, which is 21771 bytes.



57600 byte

40824 byte

57600 byte

Width : 160

Height : 120

Compression : Shannon-Fano

Mean Square Error : 0

Redundancy Data (RD) : 0.291313

Compression Ratio (CR) : 1.41106

Compression Techniques : Lossless Compression

Figure 10. Results Display Image 5

The image contained in Figure 10 still included in the category of similar images because it has a low color intensity variation, namely red. It can see in the histogram graph, image five that the image intensity is more directed to the value of 0 and has a higher number of frequencies at the value of image 0.

The captured image is an RGB image with a size of 160 × 120. The image then compressed using the Shannon-Fano algorithm and produces a compression ratio of 1.41106 with redundancy data of around 0.291 (29%). The lower the color intensity of the compressed image, the higher the amount of redundant data that can generate. This will also affect the size of the compressed image which will be smaller than the original image. The difference in length of bytes between the original image and the image compression results in Figure 10 which is 16776 bytes. Complete data for similar images 1 to 4 (Figure 7 until 9) taken using a webcam can see in more detail in Table 1.

Table 1. Summary Homogeneous Image Data with the Shannon-Fano Compression Technique

Homogeneous Imagery	Original Image Size	Compression Image Size	Decompression Image Size	Width × Height	C.R	R.D
Image 1	57600 bytes	9387 bytes	57600 bytes	160×120	6.13827	0.837 (84%)
Image 2	57600 bytes	43966 bytes	57600 bytes	160×120	1.31027	0.236 (24%)
Image 3	57600 bytes	7233 bytes	57600 bytes	160×120	7.96777	0.874 (87%)
Image 4	57600 bytes	35829 bytes	57600 bytes	160×120	1.60778	0.378 (38%)
Image 5	57600 bytes	40824 bytes	57600 bytes	160×120	1.41106	0.291 (29%)

Based on complete data on similar images that can see in table 1, it concluded that five similar images that have tested using the Shannon-Fano image compression application have an average compression ratio value of 52%. The compression results reach half of the original image. Any data is a series of bits 0 and 1. What distinguishes between data with other data is the size of the series of bits and how the 0 and 1 placed in the series of bits. For example, data in the form of text and data in the form of images, in-text data, a series of specific bits represents one character, whereas in image data a series of bits represents a color in one pixel. The more complex a data, the size of the required series of bits is more extended, thus the overall size of the data is also getting bigger [25], [26].

4.2. Product Testing

Testing is the process of running a program to find errors [27]. Testing of image compression applications using the Shannon-Fano algorithm will tested using the Black Box testing method and measuring the accuracy of the results of the compression process with objective fidelity criteria (MSE), square error, absolute difference, signal to noise ratio (SNR), peak signal to noise ratio (PSNR), and Average of absolute difference. However, in image compression applications using the Shannon-Fano algorithm, only MSE objective truth criteria displayed in the text box.

Black-Box testing focuses on the functional requirements of the software that allows engineers to obtain a set of input conditions that will fully implement the functional requirements for a program [28]. The black-box testing method carried out in research to ascertain whether the function runs according to the input provided. The results of this test will get the suitability of the compressed image.

Table 2. Homogeneous Image Suitability Test Results

Image	Compression	R.D	C.R	Result
Image 1	9387 bytes	0.837 (84%)	6.13827	exactly
Image 2	43966 bytes	0.236 (24%)	1.31027	exactly
Image 3	7233 bytes	0.874 (87%)	7.96777	exactly
Image 4	35829 bytes	0.378 (38%)	1.60778	exactly
Image 5	40824 bytes	0.291 (29%)	1.41106	exactly

In this test, several images will be captured using a webcam and then compressed to get a smaller size, and after that will be decompressed again to produce an output image where the decompressed image will produce the same size as the original image. Based on this stage, test data will be generated suitability to take pictures, compression, decompression, save, RD, and CR. Table 2 is a homogeneous image suitability testing table. Data on the original image and the results of decompression from images 1 to 5 have the same byte length that is 57600-bytes and stored in the file extension format. This is following what will generate in the test data.

The next test is to measure the truth criteria of the compression process with objective fidelity criteria. Criteria calculation is done with the following formula:

Formula 1. Mean Square Error (MSE)

$$error^2 = \frac{mserror + (result\ (x,y) - orginal(x,y))^2}{m \times n}$$

Formula 2. Square Error

$$error = mserror + (result\ (x,y) - original\ (x,y))^2$$

Formula 3. Absolute Difference

$$error = mserror + |\ (result\ (x,y) - original\ (x,y))\ |$$

Formula 4. Signal to Noise Ratio (SNR)

$$SNR = \frac{mserror + (result\ (x,y))^2}{mserror + (result\ (x,y) - original\ (x,y))^2}$$

Formula 5. Peak Singnal to Noise Ratio (PSNR)

$$PSNR = 20\,log\,10\ \frac{max\ |\,result\ (x,y)\ |}{mserror + (result\ (x,y) - original\ (x,y))^2}$$

Formula 6. Average of Absolute Difference

$$error^2 = \frac{mserror + |\ result\ (x,y) - original\ (x,y)\ |}{m \times n}$$

Shannon-Fano algorithm is image compression using lossless compression techniques so that to measure the truth, and it needs a method to measure the difference in distortion between the original image and the resulting image. This application can be successful if the results of the calculation of objective truth criteria obtained as in Table 3 as follows.

Table 3. Criteria for Correctness of The Results of The Compression Process

Image	Criteria for Objective Truth						Lossless Compression
	Square Error	Absolute Difference	MSE	SNR	PSNR	Average of Absolute Difference	
Image 1	0	0	0	Infinity	Infinity	0	exactly
Image 2	0	0	0	Infinity	Infinity	0	exactly
Image 3	0	0	0	Infinity	Infinity	0	exactly
Image 4	0	0	0	Infinity	Infinity	0	exactly
Image 5	0	0	0	Infinity	Infinity	0	exactly

Compressed images stored in files in specific formats, for example JPEG (Joint Photographic Experts Group). While the decompression process is the process to decompose the image that reused into an image that not compressed image stored in a file with a specific format, for example JPEG (Joint Photographic Experts Group). While the decompression process is a process to parse the image, which reused to an image that is not [29].

Usually, the more significant the percentage of compression, the smaller the memory requirements needed so that the image quality decreases. Furthermore, conversely, the smaller the percentage of the compressed image, the better the quality of the image compression results. An excellent compression method is a compression method that able to restore the compressed image to its original image without damaging the image quality or losing any information. Usually the more quality the compression results, the higher the memory needed. Conversely, the worse the quality of the image compression results, the smaller the memory requirements needed [30]. An excellent compressed image format is suitable for the needs of sending and storing data.

5. Conclusions

Image compression applications using the Shannon-Fano algorithm can apply to an image of a specific size. Image compression using the Shannon-Fano algorithm for similar images tested the average ratio is enormous at around 52%. Image compression using the Shannon-Fano algorithm, although still lacking in the compression ratio compared to other compression algorithms, but the advantage lies in quality because image compression Shannon-Fano algorithm uses lossless compression techniques to process the original image data into smaller sizes with no loss of information due to compression.

Acknowledgments

This research fully supported by the Department of Electrical, Faculty of Engineering, Universitas Negeri Makassar, and Mrs. Dyah Darma Andayani has provided input and solutions in the completion of this experiment.

References

[1] A. Rowstron and P. Druschel, “Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility,” in *ACM SIGOPS Operating Systems Review*, 2001, vol. 35, no. 5, pp. 188–201.

[2] S. Eickeler, S. Muller, and G. Rigoll, “High quality face recognition in JPEG compressed images,” in *Proceedings 1999 International Conference on Image Processing (Cat. 99CH36348)*, 1999, vol. 1, pp. 672–676.

[3] M. Mancuso and S. Battiato, “An introduction to the digital still camera technology,” *ST J. Syst. Res.*, vol. 2, no. 2, 2001.

[4] N. Goldberg, *Camera technology: the dark side of the lens*. Elsevier, 1992.

[5] K. Yabuta and H. Kitazawa, “Optimum camera placement considering camera specification for security monitoring,” in *2008 IEEE International Symposium on Circuits and Systems*, 2008, pp. 2114–2117.

[6] A. C. Bovik, *Handbook of image and video processing*. Academic press, 2010.

[7] R. Gryder and K. Hake, “Survey of data compression techniques,” Oak Ridge National Lab., TN (United States), 1991.

[8] G. Held and T. R. Marshall, *Data and image compression: tools and techniques*. John Wiley & Sons, Inc., 1996.

[9] K. Sayood, *Introduction to data compression*. Morgan Kaufmann, 2017.

[10] L. G. Rueda and B. J. Oommen, “A nearly-optimal Fano-based coding algorithm,” *Inf. Process. Manag.*, vol. 40, no. 2, pp. 257–268, 2004.

[11] L. Rueda and B. J. Oommen, “A fast and efficient nearly-optimal adaptive Fano coding scheme,” *Inf. Sci. (Ny).*, vol. 176, no. 12, pp. 1656–1683, 2006.

[12] J. S. Vitter, “Design and analysis of dynamic Huffman coding,” in *26th Annual Symposium on Foundations of Computer Science (sfcs 1985)*, 1985, pp. 293–302.

[13] J. S. Vitter, “Dynamic huffman coding,” 1989.

[14] D. E. Knuth, “Dynamic huffman coding,” *J. algorithms*, vol. 6, no. 2, pp. 163–180, 1985.

[15] D. A. Huffman, “A method for the construction of minimum-redundancy codes,” *Proc. IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[16] I. H. Witten, R. M. Neal, and J. G. Cleary, “Arithmetic coding for data compression,” *Commun. ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[17] P. G. Howard and J. S. Vitter, “Arithmetic coding for data compression,” *Proc. IEEE*, vol. 82, no. 6, pp. 857–865, 1994.

[18] J. Ziv and A. Lempel, “Compression of individual sequences via variable-rate coding IEEE Transactions Information Theory, 24: 530-53 (1978),” *Math. Sci. Dir. by Grad. Sch. Math. Sci.*

[19] J. Ziv and A. Lempel, “A universal algorithm for sequential data compression,” *IEEE Trans. Inf. theory*, vol. 23, no. 3, pp. 337–343, 1977.

[20] M. Nelson, “LZW data compression,” *Dr. Dobb’s J.*, 1989.

[21] H. Al-Bahadili and S. M. Hussain, “An adaptive character wordlength algorithm for data compression,” *Comput. Math. with Appl.*, vol. 55, no. 6, pp. 1250–1256, 2008.

[22] S. Shanmugasundaram and R. Lourdusamy, “A comparative study of text compression algorithms,” *Int. J. Wisdom Based Comput.*, vol. 1, no. 3, pp. 68–76, 2011.

[23] S. R. Kodituwakku and U. S. Amarasinghe, “Comparison of lossless data compression algorithms for text data,” *Indian J. Comput. Sci. Eng.*, vol. 1, no. 4, pp. 416–425, 2010.

[24] M. Vaidya, E. S. Walia, and A. Gupta, “Data compression using Shannon-fano algorithm implemented by VHDL,” in *2014 International Conference on Advances in Engineering & Technology Research (ICAETR-2014)*, 2014, pp. 1–5.

[25] T. J. Lynch, *Data compression: techniques and applications*, vol. 1. Lifetime Learning Publications Belmont, CA, 1985.

[26] M. Nelson and J.-L. Gailly, *The data compression book*. M & t Books New York, 1996.

[27] G. J. Myers, C. Sandler, and T. Badgett, “The art of software testing, JohnWiley & Sons,” *Inc, Canada*, 1979.

[28] R. S. Pressman, *Software engineering: a practitioner’s approach*. Palgrave Macmillan, 2005.

[29] R. L. De Queiroz, “Processing JPEG-compressed images and documents,” *IEEE Trans. Image Process.*, vol. 7, no. 12, pp. 1661–1672, 1998.

[30] B. C. Smith and L. A. Rowe, “Algorithms for manipulating compressed images,” *IEEE Comput. Graph. Appl.*, vol. 13, no. 5, pp. 34–42, 1993.