THE INFLUENCE OF SEGMENTATION USING VIOLA AND JONES ON THE MARKOV STATIONARY FEATURE - VECTOR QUANTIZATION IN THE CASE OF THE INTRODUCTION OF FACIAL EXPRESSIONS

Encep Suryana[1], Irfan Maliki[2]

[1.2] Informatics Engineering Program
Faculty of Engineering and Computer Science - University Computer Indonesia
Jl. Dipatiukur No. 112-114 Bandung
E-mail: encep.suryanajr@gmail.com1, irfanmaliki007@gmail.com2

ABSTRACT

Facial expressions play important in communicating with others people as expressing feelings or emotions. On the detection of facial expression can be segmentation process to separate the object contained in faces image including: eyes, nose and mouth. Segmentation method can be used is a Viola and Jones. In case of the introduction facial expressions using the method Markov Stationary Feature – Vector Quantization (MSF-VQ) without segmentation and use testing method Confusion Matrix get an accuracy 97.01%. In this research the data used in this research consisted of Training Data and Testing Data, a total data of 2205 images data obtained from 42 peoples. Data's include: 21 peoples expressions happy, sad, angry, disgust, scared and surprised, and 21 other peoples expressed neutral. Each data taken is 15 images per person. This research topic "the influence of segmentation Viola and Jones against features MSF-VQ to recognize facial expressions" helping with learning machines Multiclass Support Vector Machine and kernel linear influencing in faster training and testing datas and get better accuracy in the process recognizing facial expressions, get an accuracy 100% on recognition of facial expressions and an increase of 2.99%.

Keywords: Facial Expressions, Segmentation, Viola and Jones, Markov Stationary Feature, Vector Quantization, Support Vector Machine, Testing, Confusion Matrix, Feature Extraction.

1. INTRODUCTION

Facial expressions play an important role in communicating with others as an expression of feelings or emotions [1]. On detection of facial expressions to do the segmentation process that aims to separate objects contained in the face image include: eyes, nose and mouth, andsegmentation usually used in the field of human computer interaction and data-driven animation, especially on the segmentation of the facial image, are used to partition the digital image face into several segments or pixels that aims to simplify or change the representation of a facial image becomes easier to analyze [2] ,

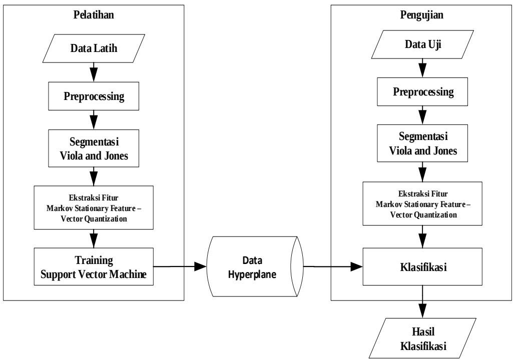
The studies have been done about the segmentation include: Conan perform face detection research based segmentation using the color model *template Matching* on moving objects get an accuracy of 65% [3], Douglas conducts research on the segmentation of the face using skin-color map on a videophone application to get an accuracy of 82% [4].

Based on research conducted by Chandra using methods *Viola and Jones*get an accuracy of 90.99% to detect a face objects [5]. Research by Dharani on Human Segmentation Using Haar-Classifier get an accuracy of about 93% [2] and a thesis by Fascal implementation Stationary Markov Feature - Vector Quantization For Facial Expressions Recognition get an accuracy of 97.14% [6]. Therefore research on the Effect of Segmentation Viola and Jones against Markov Stationary Features Feature - Vector Quantitazion in cases of facial expression recognition, in order to simplify the research area of the object face by using segmentation method viola and jones.

2. RESEARCH CONTENT

2.1. Analysis Methods

The analysis methods used have two processes including a process of training and testing process. The training process is used to train the training data and the testing process used for testing of the training data on the test data, the following is the flow of the system in this study:


Pelatihan
Data Latih
Preprocessing
Segmentasi Viola and Jones
Ekstraksi Fitur Markov Stationary Feature – Vector Quantization
Training Support Vector Machine
Data Hyperplane
Pengujian
Data Uji
Preprocessing
Segmentasi Viola and Jones
Ekstraksi Fitur Markov Stationary Feature – Vector Quantization
Klasifikasi
Hasil Klasifikasi

Picture 1, Flow system

From the analysis of the methods below is an explanation of the flow of the system:

2.1.1 Training process

The training process is done for the training process input data in the form of training and then through the image preprocessing process, viola and jones segmentation, feature extraction feature stationary Markov - vector quantization and Training Support Vector Machine. For the stages of the training process as follows:

2.1.1.1. *preprocessing*

*preprocessing*is the stage to change the image size to be smaller then change color image (RGB) into a grayscale image and image smoothing prior to segmentation and feature extraction. The stages as follows:

1. *Resize Image*

Resize Image used to change the image size becomes smaller. This process is used so that the calculation process is faster and does not spend much temporary storage memory, the following calculation process:

ResizeW = w × SkalaResize (1)
ResizeH = h × SkalaResize (2)

Information:
ResizeW = The size of the new image width (width)
ResizeH = Length of the new image (*height*)
w Width = size of the original image (*width*)
h = Length of the original image (*height*)

2. *grayscale*

The process of converting a grayscale or color image (RBG) of gray image to aim to change the RGB image into a single value. For example as follows:

Picture 2, grayscale

The process of changing the RGB image into grayscale image, using the following calculation:

***grayscale* = 0:29 × R + G + B × × 0:59 0:11** (3)

Information:
R: value of the pixels of red (Red)
G: value of green color pixels (Green)
B: the value of a pixel in blue (Blue)

3. *Image Smoothing*

Image smoothing process used to change the image to become softer with use as a convolution methods of calculation, where this method is a process for obtaining a pixel based on the pixel value of the pixel itself and its neighbors, with the involvement of a matrix called karnel representing weighting. The calculation process of smoothing the image using a median filter.

$$\boldsymbol{h(x,y)} = \frac{\mathbf{1}}{\boldsymbol{mn}} \sum_{\boldsymbol{x=0}}^{\boldsymbol{m-1}} \quad \sum_{\boldsymbol{y=0}}^{\boldsymbol{n-1}} \qquad (4)$$

Information:
h (x, y) = filter *h* (Median filter)
n = Number of lines in filter *h* (Median filter)
m = Number of columns on the filter *h* (Median filter)
x = Coordinates of the location of the image at point *x*
y = Coordinates of the location of the image at point *y*

Grayscale image is defined as f (x, y), then use karnel 33 for image smoothing process as h (x, y) and the value of the convolution of f (x, y) on a grayscale image, as follows:×

$$h(x,y) = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

The following steps to change the image becomes more refined to the value f (x, y) as follows:

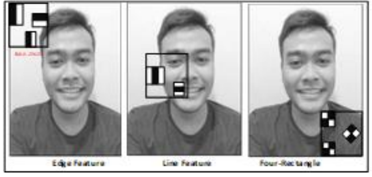
i. Select f (x, y) size 33, starting from the top left corner to the bottom right,×
ii. Then replaced the pixel values of the center position with the pixel values after convolution calculation process,
iii. Next slide pixels 33 of the pixel to the right and did until the final pixel value, and×
iv. For the pixel values which are on the side of right, left, up and down or pixels outside the pixel value of fixed or previous pixel value of the pixel grayscale value.

2.1.1.2. Segmentation Viola and Jones

The segmentation process is used to separate the object with the background needed to narrow the study area on the face of the object is focused on the mouth, eyes and nose object. Following stages:

1. *Haar-Like Feature*

*Haar-Like Feature*a technique for detecting an object in the image in a way to compartmentalize each pixel is called the base imagery ranging from the top-left to bottom right, the process is carried out to search if there are facial features in the image. In the Viola and Jones algorithm, there are several types of features that can be used as edge-feature, Line feature, and Four-rectangle feature.



Picture 3, Illustration haar-like feature

2. *integral image*

integral Image used for object detection process, for the calculation process using the integral image using a faster and more accurate results, calculations obtained from the pixel values in haar feature, then calculated the value of the integral image with the following formula:

$$s(x,y) = i(x,y) + s(x,y) + s(x,y-1) + s(x-1,y) - s(x-1,y-1) \quad (5)$$

Information:

s (x, y) = The value of the penjumlahan of each pixel.

i (x, y) = an intensity value obtained from the pixel values of the input image.

s (x-1, y) = The pixel values of the axis *x*,

s (x, y-1) = The pixel values of the axis *y*,

s (x-1, y-1)= merupakan diagonal pixel value.

The following integral calculation process image:

98	87	98	96
92	86	98	105
92	88	95	107
88	89	95	104
80	90	96	103
77	94	96	100
89	98	99	99
96	99	100	95

Picture 4, The pixel values of a feature.

i. Calculation of the pixel (36.57):

i(x,y) =98	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

Picture 5, The pixel values (36.57)

Pixel intensity values (36.57), dani (x, y) = 98

i (x, y) = 98

s (x-1, y) = 0 (outside the boundaries of the matrix)

s (x, y-1) = 0 (outside the boundaries of the matrix)

s (x-1, y-1) = 0 (outside the boundaries of the matrix)

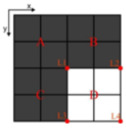
s (x, y) = 98 + 0 + 0-0 = 98.

The calculation is performed up to the last pixel, get the value of the integral image as follows:

98	185	185	194
190	167	195	201
184	174	195	209
180	173	191	211
168	171	192	208
157	181	196	203
166	204	199	202
185	204	200	195

Picture 6, Results integral image

Then do the calculation to find a specific region on the image as follows:



98	185	185	194
190	167	195	201
184	174	195	209
180	173	191	211
168	171	192	208
157	181	196	203
166	204	199	202
185	204	200	195

Picture 7, Calculate pixels in a certain area

The calculation process by summing the pixels in the region D with the following formula:

$$\mathbf{D = L1 + L4 - (L2 + L3)} \quad (6)$$

3. *Adaptive booster*

*Adaptive booster*is a technique used to combine many weak classifier and form a combined classifier better. By using algorithms adaboost an image is then detected again, perhitungannya as follows:

The initial weight: $\boldsymbol{w_{j_{y_i}} = \frac{1}{2m}, w_{t_{y_i}} = \frac{1}{2l}}$ (7)

information:

w = *classifier week*

m = The number of positive image

l = Number of negative image

t = Index of iterations of a positive image

j = Iteration index of the negative image

$h_t(x)$ = The value of a positive image features

$h_j(x)$ = Value of the negative image features

To get the value of the error rate in each weak classifier, then on any feature of the process is done as follows:

i. Calculate the value of the initial weight:

The initial weight = $w_{1,0} = \frac{1}{2\times4} = 0.125, w_{1,1} = \frac{1}{2\times1}$ = 0.5

Specify a positive image, to process the object peteksian or not:

a. Positive image to-1

173	150	150	149	146	146	146	146	153	151	150	145	144	142	142	142	145	144
151	150	150	149	146	146	146	146	151	150	149	145	144	143	142	142	145	143
151	150	150	149	146	146	146	146	150	150	149	145	144	143	143	143	145	143
151	150	150	149	146	146	146	146	150	149	149	146	145	144	144	143	145	143
151	150	150	149	146	146	146	146	146	146	146	146	145	145	144	144	145	143
151	150	150	149	146	146	146	146	145	145	146	146	146	146	145	145	145	143
151	150	150	149	146	146	146	146	144	144	145	146	146	146	146	145	145	143
151	150	150	149	146	146	146	146	143	144	145	146	149	149	146	146	145	143
151	151	150	150	150	149	133	146	132	146	146	149	149	146	146	146	145	143
151	151	150	150	149	149	146	146	145	146	146	146	146	146	146	145	145	143
151	150	150	150	149	146	146	146	145	146	146	146	146	146	146	145	145	143
155	150	150	150	149	146	146	145	147	145	146	146	146	146	145	145	145	146

Picture 8, Positive image to-1

Calculation:

a) $h_t(x)$= | ((147 + 173) - (155 + 153)) - ((146 + 153) - (147 + 144)) | = | 12 - (- 8) | = 4

b) Values obtained error rate:

$$\epsilon_t = (0.125)|4 - 1| = 0.375$$

Repeat until the entire positive image and generate error rate as follows: 0.375, 29.13, 31 875 and 18.7. then select the smallest error rate is 0.375, where:$\beta_t = \frac{0.375}{1-0.375} = 0.6$

ii. Weights iteration 1 are: w2,1 = 0.375× 0.6 = 0.225

ϵ_t = (+0.3750225) | 2-1 | = 0.6

$$\beta_t = \frac{\epsilon_t}{1-\epsilon_t} = \frac{0.6}{1-0.6} = 1.5$$

Repeat until iteration n <0 then the iteration stops, to get the results of the classification of the positive image using the following formula:

$$H(x) = \begin{cases} 1 \ \sum_{j=1}^{J} \alpha_j h_j \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ \qquad bukan\ objek. \\ 0 \end{cases} \qquad (8)$$

Where:

$$\alpha_j = \log \frac{1}{\beta_j}, \ \alpha_t = \log \frac{1}{\beta_t} \qquad (9)$$

Condition:

If position = the result is 1 then the image of the object$H(x)$

If position = the result is 0 then the image is not an object$H(x)$

Information:

$H(x)$ = Strong stating Classifier or classification of objects or not.

α_j = Level of learning was a positive image.

α_t = Level of learning was negative image.

β_j = The weight after *error rate* the negative image.

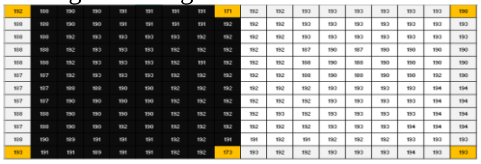
β_t = The weight after the error rate in a positive image.

h_j = *weak* or basic classifiers (beginning of classification) negative image.

h_t = Weak or basic classifiers (beginning of classification) positive image.

Determining the negative image is an object or not, as follows:

a. The negative image



192	188	190	190	191	191	191	191	171	192	192	193	193	193	193	193	193	198
189	189	190	190	191	191	191	191	192	192	192	193	193	193	193	193	193	193
188	188	192	193	193	193	193	192	192	192	192	193	193	193	193	193	193	193
188	188	192	193	193	193	192	192	192	192	192	187	190	187	190	190	190	190
188	188	192	193	193	193	192	191	192	192	192	188	190	188	190	190	190	190
187	187	192	193	193	193	192	192	192	192	192	188	190	188	190	190	192	190
187	187	188	188	190	190	192	192	192	192	192	192	193	193	193	193	194	194
187	187	190	190	190	190	192	192	192	192	192	192	193	193	193	193	194	194
188	188	190	190	190	190	192	192	192	192	193	192	193	193	193	193	194	194
187	188	190	190	192	190	192	192	192	192	192	192	193	193	193	194	194	194
188	190	189	191	191	191	192	192	191	191	192	191	192	192	192	193	193	193
193	191	191	189	191	191	192	192	173	193	192	192	193	193	193	194	193	193

Picture 9, The negative image

The calculations are as follows:

1. $h_j(x)$ = | ((173 + 192) - (193 + 171)) - ((193 + 171) - (173 + 198)) | = | 1 - (- 7) | = 8
2. Values obtained error rate:
   ϵ_j= (12:08) | 8-1 | = 0:58
3. Weight value after error rate
   $\beta j = \frac{\epsilon_j}{1-\epsilon_j}$ Then = 1.38$\beta j = \frac{0.58}{1-0.58}$
   ϵ_t= () | 1:38 | = 3.281 + (1.38)

After the first iteration, then updates the weights as follows:

$w_{3,1}$ = 0:08 × (-1.462) = -0,115

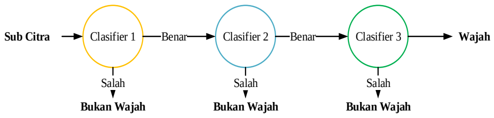
To find the value in use to determine if a negative image is the object sought or not, here are the results:$H(x) = \sum_{j}^{J} a_j h_j(x) \geq \frac{1}{2} \sum_{t}^{T} a_t$

$H(x)$ = 32.97984177 ≥ -0.069939543

Because32.97984177 ≥ -0.069939543 is is true then the negative image of the object.

4. *Casecade Classifier*

Cascade classifier a method for combining classifier complex in a multi-storey structure and can improve the speed of detection of an object in an image focused on the image area likely only, and here is the process of cascade classifier:



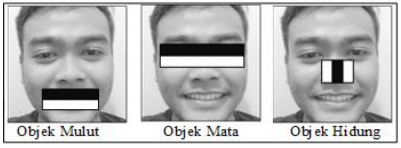
Picture 10, *Cascade Classifier*

From the haar clasified then generated an image that has been selected face, as shown below:

Picture 11, Citra Detected

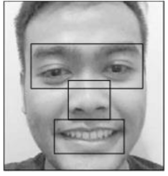
5. *Bounding Box haar-like feature*

This process is a process to find the object in the image as the object of both eyes, the object's nose and mouth object by marking the area to be used in the form of horizontal and vertical lines that make up the box on that object. Following the results of the search algorithm using Haar-like features [9]:



Picture 12, The results of the detection image

After that do the cutting process the image which already has a bounding box to separate objects that are not needed in the face image. Here's an example of a bounding box in the image:



Picture 13, *bounding box* imagery

Here's an example of the image that has been cut, by separating the objects used:

Picture 14, The image that has been cut.

6. *thresholding*

By using thresholding to convert the image into a binary image [7], in this study the degree of gray values using the value 195, the value is taken from the previous image testing using tools help ImageJ [12].

Basically, the process is changing quantization thresholding on the image, so as to do with the degree of gray thresholding using the following formula:

$$\boldsymbol{x = \frac{w}{b}} \qquad (10)$$

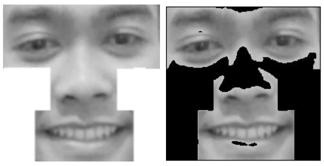
Information:

w = The value of the degree of gray before thresholding.

b = Number of degrees of gray desired.

x = The value of the degree of gray after thresholding.

Here are the results of the comparison image and the image is cut off after thrusholding stages, as follows:



Picture 15, image thresholding

2.1.1.3. Stationary Markov feature extraction Feature - Vector Quantization

Stationary Markov feature extraction feature - vector quantization (MSF-VQ) is to compress the image and evoke every feature in the image to recognize expressions. Here are the stages:

1. Method of Vector Quantization

vector quantization (VQ) is used to compress the image so that the image quality is getting better and one lossy compression methods are most strongly with low calculation complexity [8] [10]. The following stages of the method VQ:

a. Making the random codebook, with itisial as a codebook.CB_0
b. Add i = 0.
c. Perform the following steps to each vector trainer:
   i. Calculate the Euclidean distance between vectors in the codebook latihdan. And the Euclidean distance is defined as follows:CB_i

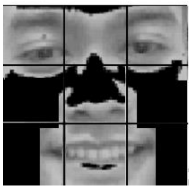
$$d(X,C) = \sqrt{\sum_{t=1}^{k}(x_t - c_t)^2} \qquad (11)$$

   ii. Find the closest codeword CB_i
d. Separate codebook into column N.
e. Calculate the average of each column to obtain the new codebook.CB_i
f. Calculate the average distortion in, if amended by the smallest amount in the last iteration, the calculation is repeated until prosdur codebook stalled. If not i = i + 1 and proceed to phase 3.CB_{i+1}

2. *Stationary Markov Feature*

Stationary Markov Feature (MSF) is used to generate feature on every expression of the image that has been processed to the stage Vector Quantization [11], while the stages as follows:

Before the process of MSF, added blocks on each image to provide a filter like the filter of 3 × 3, 5 × 5 and 7 × 7 is used for the weighting of feature extraction in the image, and the following is an example of using a filter of 3 × 3 in the image train :



Picture 16, Filter 3 × 3 in the image

a. Generating Co-Occurrance Matrix

Generation co-occurance matrix is used to normalize the image feature extraction with practice by changing the pixel intensity values. By changing the image of train into 5-bit image value range between 0-32 bits, using the following formula:

$$\boldsymbol{Citra\ Uji = Nilai\ Grayscale * \frac{Nilai\ Graylevel}{255}} \qquad (12)$$

Then do the calculation of all image values trainerbased on the angles used are,,, and $0^0 45^0 90^0 135^0$

b. Search Initial Distribution

Search intial distribution through proper co-occurance matrix diagonally, from top left to bottom right, so get initialization value matrix, then do the sum of values of diagonal and do averaging the value of the share to the total amount of the diagonal.

c. Generation Markov Transition Matrix

To generate Markov transition matrix using matrix computation of the co-occurance of the Markov transition matrix, determines value by using the following formula:p_{ij}

$$p_{ij} = c_{ij} / \pi \approx \frac{1}{K}\sum_{i=1}^{K} c_{ij} \qquad (13)$$

Information:

p_{ij} = as *Markov transition matrix*,

c_{ij} = Matrix of *Initial Distribution*

K = The value of *Initial Distribution.*

d. Finding Stationary Distribution

To find the value of stationary distribution using Markov Chain calculation, by doing a search of the limit values, which $A = \lim_{n\to\infty} A_n \, A_n = \frac{1}{n+1}(I + P + P^2 + \cdots P^n)$ (14)

Information:

I = Is the identity matrix.

n = The average value of each line.$\overrightarrow{a_i}A_n$

Make perhitunggan on feature values using the following formula:

$$\pi \approx \frac{1}{K}\sum_{i=1}^{K} \vec{\alpha}_{ij}, \text{ Where } A_n = [\overrightarrow{a_1}, \dots, \overrightarrow{a_n}]^T \quad (15)$$

e. MSF Feature Generation

MSF Feature Generation is a stage for normalizing the transition matrix is used to search for features MSF conducted to look for value$\overrightarrow{MSF} = [\pi(0), \pi]^T$ (16)

2.1.1.4. Training multiclass Support Vector Machine

Multiclass suppport Vector Machine(M-SVM) is used to perform a binary classification, this method is selected because it includes methods that are popular and can divide the space into 2 classes vector positive and negative class. It is possible to use for the purposes of dividing into more than two classes [14]. The approach used methods One-Against-All (OAA), and follows the M-SVM:

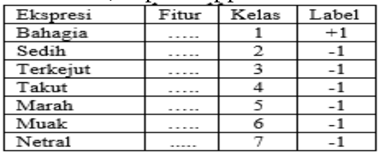
There are seven classes will be made on the training data, including: Unhappy with the label first, the label 2 Sad, Surprised by the label 3, Fear label 4, Angered by the label 5, Sick 6 and Neutral label by label 7. Here's the process M -SVM:

a. M-SVM with label 1 will be given a positive sign (+1) and label 2, 3 label, the label 4 and label 5 is given a negative label (-1). If the result is positive grade (+1), the result is the label 1. If the result is negative (-1) then go to the loop b.

b. M-SVM with label 2 will be labeled positive (+1) and label 1, label 3, 4 labels, and label 5 is given a negative label (-1). If the result is positive grade (+1), the result is the label 1. If the result is negative (-1) then looping up to the next class.

In the training process, use the values of the features that have been obtained from the extraction process features as support vector value.

Table1, Input Support Vector



Ekspresi	Fitur	Kelas	Label
Bahagia	.....	1	+1
Sedih	.....	2	-1
Terkejut	.....	3	-1
Takut	.....	4	-1
Marah	.....	5	-1
Muak	.....	6	-1
Netral	.....	7	-1

To find the value of the kernel using the kernel trick Linear, calculation kernel using the following formula:

$$\mathrm{K}(\mathrm{x_i}, \mathrm{x}) = \mathrm{x_i^T x} \quad (17)$$

$$K(\mathrm{y_i}, \mathrm{y}) = y_1^T\, y \quad (18)$$

To get the class of the value of the kernel, add up to the value of the support vector, using the following formula:$\mathrm{K}(\mathrm{x_i}, \mathrm{x})x$

$$x_i = \sum_{j=1}^{n} x_i^T x_j \quad (19)$$

$$y_i = \sum_{j=1}^{n} y_i^T y_j \quad (20)$$

For value and that is a support vector value, then define:$x_i y_i$

a. *kernel* Trick

In value x_i and have been found, then find the value of the kernel trick using the following formula:y_i

$$\varphi\binom{x}{y} = \begin{cases} \sqrt{x_n^2 + y_n^2} > 2, maka \begin{bmatrix} \sqrt{x_n^2 + y_n^2} - x & |x-y| \\ \sqrt{x_n^2 + y_n^2} - x & |x-y| \end{bmatrix} \\ \sqrt{x_n^2 + y_n^2} < 2, maka \binom{x}{y} \end{cases} \quad (21)$$

b. Value a

To determine the value by using the following formula:a

$$\sum_{i=1,j=1}^{n} a_i T_i^T T_j \quad (22)$$

c. value Bias

To determine the value of the bias by adding the number 1 in the Kernel Trick as follows:

$$s_i = \begin{matrix} x_i \\ y_i \\ 1 \end{matrix} \quad (23)$$

d. value hyperplane

then determine hyperplane value using the following formula:

$$W = \sum_i^n a_i s_i \quad (24)$$

2.1.2. Testing process

The testing process to test each image that has been trained, stages: preprocessing, segmentation viola and Jones, stationary Markov feature extraction feature - vector quantization and classification. In the testing process, the process of preprocessing, viola and jones segmentation, feature extraction feature stationary Markov - vector quantization, with the same process and following the training stage is the stage of classification in the testing process:

1. Classification

The classification process is a process of testing of the training image by using support vector value, then the value of that vector didistirbusikan using the following formula:

$$kelas\ x = \arg\max_{k=1..7}([w^1]^T.\varphi(x) + b^1, [w^2]^T.\varphi(x) + b^2, [w^3]^T.\varphi(x) + b^3, [w^4]^T.\varphi(x) + b^4, [w^5]^T.\varphi(x) + b^6, [w^6]^T.\varphi(x) + b^6), [w^7]^T.\varphi(x) + b^7 \quad (25)$$

2.2. Performance Testing

In performance testing using the Confusion Matrix [13] which is used to measure the performance at the time of classification. Tests conducted on the filter of 3 × 3, 5 × 5 and 7 × 7 in twice the testing process include:

a. 100% training data and test data of 100%, and

b. Training data as much as 80% and 20% of test data.

Description of the data being tested as follows: the first test by using training data 100% and the test data of 100%, all the data that were tested had been trained previously as much as in 2205 the data, then the second test using training data 80% as much as in 1764 the data and test data of 20 % who had previously trained as many as 441 data.

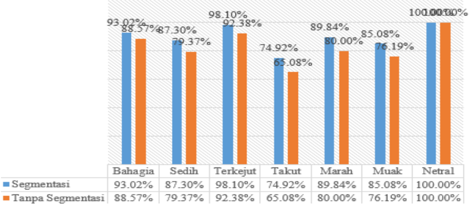
2.2.1. Test result

1. First Test

a. Filter 3 × 3

Table 2, Confusion Matrix filter 3 × 3

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	293	4	1	14	3	0	0	93.02%
Sedih	14	275	0	11	4	11	0	87.30%
Terkejut	0	3	309	2	0	1	0	98.10%
Takut	23	22	3	236	28	3	0	74.92%
Marah	7	6	2	7	283	10	0	89.84%
Muak	10	11	2	3	21	268	0	85.08%
Netral	0	0	0	0	0	0	315	100.00%
Rata-Rata Akurasi								89.75%

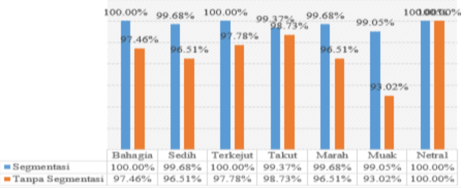


Picture 17, Graph accuracy of the filter 3 × 3

b. Filter 5 × 5

Table 3, Confusion Matrix filter of 5 × 5

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	293	4	1	14	3	0	0	93.02%
Sedih	14	275	0	11	4	11	0	87.30%
Terkejut	0	3	309	2	0	1	0	98.10%
Takut	23	22	3	236	28	3	0	74.92%
Marah	7	6	2	7	283	10	0	89.84%
Muak	10	11	2	3	21	268	0	85.08%
Netral	0	0	0	0	0	0	315	100.00%
Rata-Rata Akurasi								89.75%

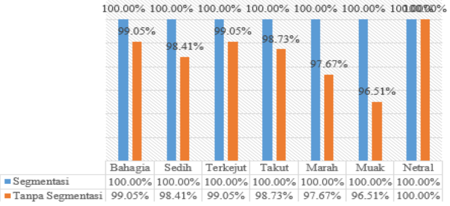


Picture 18, Graph accuracy of the filter of 5 × 5

c. Filter 7 × 7

Table4, Confusion Matrix filter 7 × 7

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	315	0	0	0	0	0	0	100.00%
Sedih	0	315	0	0	0	0	0	100.00%
Terkejut	0	0	315	0	0	0	0	100.00%
Takut	0	0	0	315	0	0	0	100.00%
Marah	0	0	0	0	315	0	0	100.00%
Muak	0	0	0	0	0	315	0	100.00%
Netral	0	0	0	0	0	0	315	100.00%
Rata-Rata Akurasi								100.00%



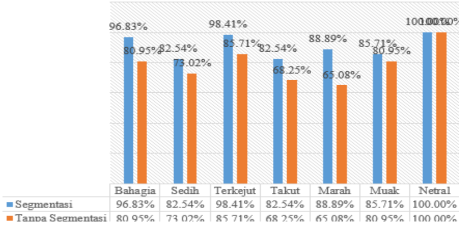
Picture 19, Graph accuracy in filter 7 × 7

2. Second Test

a. Filter 3 × 3

Table 5, Confusion Matrix filter 3 × 3

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	61	0	0	2	0	0	0	96.83%
Sedih	2	52	2	3	2	2	0	82.54%
Terkejut	0	1	62	0	0	0	0	98.41%
Takut	0	3	0	52	7	1	0	82.54%
Marah	0	2	1	0	56	4	0	88.89%
Muak	0	4	1	0	4	54	0	85.71%
Netral	0	0	0	0	0	0	63	100.00%
Rata-Rata Akurasi								90.70%

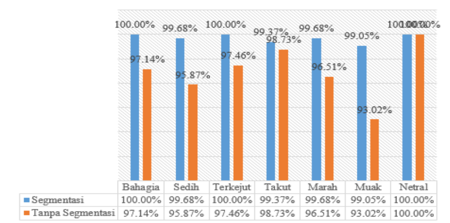


Picture 20, Graph accuracy of the filter 3 × 3

b. Filter 5 × 5

Table 6, Confusion Matrix filter of 5 × 5

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	63	0	0	0	0	0	0	100.00%
Sedih	0	61	0	1	0	1	0	96.83%
Terkejut	0	0	63	0	0	0	0	100.00%
Takut	0	1	0	62	0	0	0	98.41%
Marah	0	0	0	0	62	1	0	98.41%
Muak	0	0	0	0	0	63	0	100.00%
Netral	0	0	0	0	0	0	63	100.00%
Rata-Rata Akurasi								99.09%

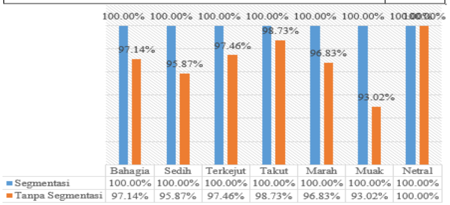


Picture 21, Graph accuracy of the filter of 5 × 5

c. Filter 7 × 7

Table 7, Confusion Matrix filter 7 × 7

Sebenarnya	Prediksi							Akurasi
	Bahagia	Sedih	Terkejut	Takut	Marah	Muak	Netral	
Bahagia	63	0	0	0	0	0	0	100.00%
Sedih	0	63	0	0	0	0	0	100.00%
Terkejut	0	0	63	0	0	0	0	100.00%
Takut	0	0	0	63	0	0	0	100.00%
Marah	0	0	0	0	63	0	0	100.00%
Muak	0	0	0	0	0	63	0	100.00%
Netral	0	0	0	0	0	0	63	100.00%
Rata-Rata Akurasi								100.00%



Picture 22, Graph accuracy in filter 7 × 7

The results of testing the accuracy by using the method of confusion matrix in 3 different filter accuracy of data obtained as follows:

Table 8, Data Accuracy

Metode	Pengujian	Data Training	Data Testing	Akurasi Filter 3×3	Akurasi Filter 5×5	Akurasi Filter 7×7
MSF-VQ+SVM	Pertama	2205	2205	83.08%	96.96%	97.01%
	Kedua	1764	441	79.14%	95.69%	96.60%
Segmentasi+MSF-VQ+SVM	Pertama	2205	2205	89.75%	99.68%	100.00%
	Kedua	1764	441	90.70%	99.09%	100.00%

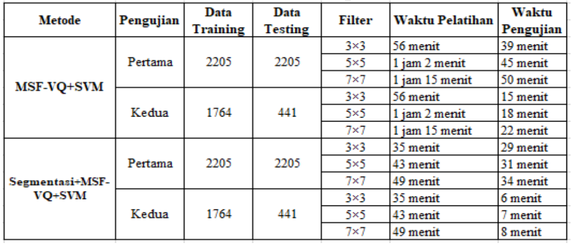
From the testing that has been done using the segmentation process, gain accuracy biggest filter 7 ×

7 with a value accuracy of 100% in the first test and the second while no segmentation process to get the accuracy biggest filter 7 × 7 with a value accuracy is 97.01% on testing the first and the increase in the detection of facial expressions by 2.99% with the same filter.

2.2.1.1. Speed Results

While the ratio of the speed in the training process the data, the following data obtained by using segmentation and training time without segmentation:

Table 9, Data speed



Metode	Pengujian	Data Training	Data Testing	Filter	Waktu Pelatihan	Waktu Pengujian
MSF-VQ+SVM	Pertama	2205	2205	3×3	56 menit	39 menit
				5×5	1 jam 2 menit	45 menit
				7×7	1 jam 15 menit	50 menit
	Kedua	1764	441	3×3	56 menit	15 menit
				5×5	1 jam 2 menit	18 menit
				7×7	1 jam 15 menit	22 menit
Segmentasi+MSF-VQ+SVM	Pertama	2205	2205	3×3	35 menit	29 menit
				5×5	43 menit	31 menit
				7×7	49 menit	34 menit
	Kedua	1764	441	3×3	35 menit	6 menit
				5×5	43 menit	7 menit
				7×7	49 menit	8 menit

3. FINAL

After conducting tests on the effect of segmentation using the Viola and Jones to feature MSF-VQ in the case of recognition of facial expressions that have an influence on the speed of training and testing data faster and produces better accuracy in the process of recognition of facial expressions to get an accuracy of 100%. And to develop this study, it was given advice by using Markov Stationary Feature - Vector quantization, take advantage of Stationary Markov method Feature - Vector quantization on the video to recognize facial expressions directly on the case to recognize one's feelings or lie detector.

BIBLIOGRAPHY

[1] Updadhyay, A., & Kumar, A. (2016). Facial Expression Recognition : A Review. *International Research journal of Engineering and Technology*, 1616-1620.

[2] Dharani S, G. S. (2014). Human Segmentation Using Haar-Classifier . *Journal of Engineering Research and Applications*, 89-93.

[3] Yusron Rijal, R. D. (2008). *Deteksi Wajah Berbasis Segmentasi Model Warna Menggunakan Template Matching Pada Objek Bergerak.* Surabaya: Skripsi S1 / Jurusan Sistem Komputer, Sekolah Tinggi Manajemen Informatika & Teknik Komputer Surabaya.

[4] Douglas Chai, K. N. (1999). Face Segmentation Using Skin-Color Map in Videophone Applications. *IEEE Transactions On Circuits And Systems For Video Technology*, 4-9.

[5] Chandra Kirana, C. K. (2016). Face Identification For Presence Applications Using Violajones and Eigenface Algorithm. *Jurnal SISFOKOM*, 1-5.

[6] Fascal Sapty Jarockohir, I. Maliki. (2018). *Implementasi Markov Stationary Feature – Vector Quantization Untuk Pengenalan Ekspresi Wajah.* Bandung: Skripsi, Fakultas Teknik dan Ilmu Komputer Universitas Komputer Indonesia.

[7] Emilio N.M. Cirillo, M. C. (2015). *Threshold effects in zero range processes.* Netherlands : Centre for Analysis, Scientific computing and Applications Department of Mathematics and Computer Science Eindhoven University of Technology.

[8] Yoseph Linde, Y. L. (1980). An Algorithm for Vector Quantizer Design. *IEEE Transactions On Communications*, 1-28.

[9] Pavani, S.-K., Delgado, D., & F.Frangi, A. (2010). Haar-like features with optimally weighted rectangles for rapid object detection. *Networking Research Center on Bioengineering, Biomaterials and Nanomedicine (CIBER-BBN), Barcelona, Spain*, 160-172.

[10] Athawale, A. K. (2007-2010). *Chapter 6 Information Hiding Using Vector Quantization.* Department of Information & Technology.

[11] Jianguo Li, W. W. (2008). *One Step Beyond Histograms: Image Representation using Markov Stationary Features.* Beijing: Intel China Research Center, Haidian, Beijing, 100190.

[12] Rasband, W. (2018, 12 1). *ImageJ*. Retrieved from ImageJ : Image Processing and Analysis in Java: https://imagej.nih.gov/ij/docs/intro.html

[13] Marina Sokolovaa, G. L. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing and Management*, 427–437.

[14] Nelly Indriani Widiastuti, E. R. (2017). Peringkasan dan Support Vector Machine pada Klasifikasi Dokumen. Jurnal Infotel, 416-421.