

## MODUL 5 – Operasi Aritmatika dan Logika - Gamma Correction, Image Depth, PSNR, Average Denoising, Image Masking

---

### A. Purpose

1. Students are able to understand the theory of Gamma Correction and make its application
2. Students are able to make image simulations with a specified image depth
3. Students are able to do image denoising using Averaging
4. Students are able to understand how to use PSNR
5. Students are able to do image masking using logical operators

### B. INSTRUCTIONS

1. Begin every practicum activity by praying
2. Read and understand the objectives, theoretical basis, and practical exercises properly
3. Do practicum assignments well, patiently and honestly
4. Ask the lecturer if there are things that are unclear

### C. Tools and Materials

1. PC/LAPTOP
2. Github
3. *Google Colaborator*

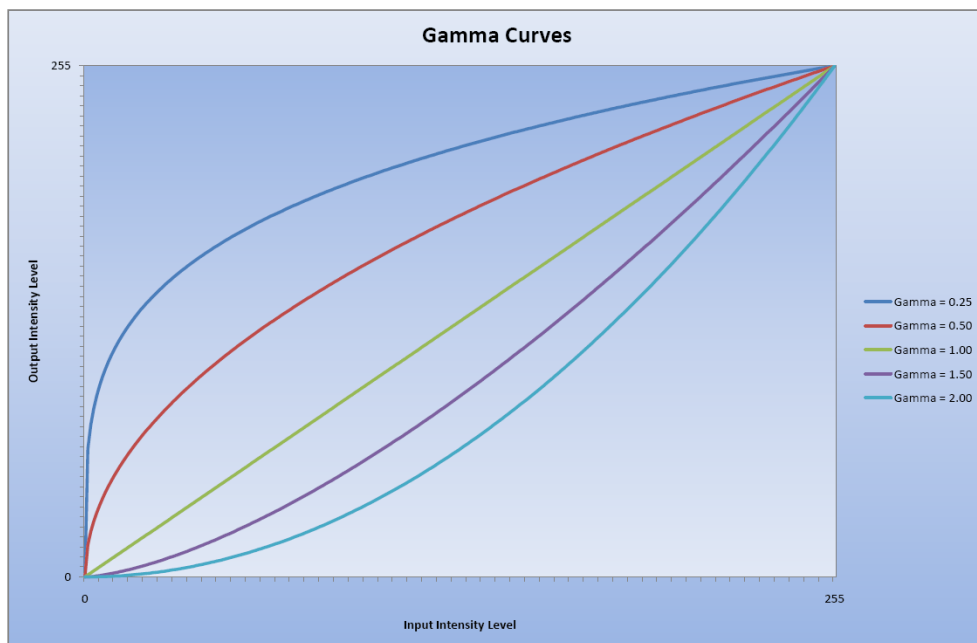
### D. Time Allocation: 6 hours

### E. BASIC THEORY

Gamma (Power-Law Transformation), symbolized by Greek letters  $\gamma$ , is described as the relationship between the resulting input and output. The input is the RGB intensity value of the image. The relationship between input and output in question is that the output is proportional to the input ranked by the Gamma value. The formula for Gamma Correction is as follows:

$$I' = 255 \times \left( \frac{I}{255} \right)^\gamma$$

The following is an illustration of the output of the Gamma curve at several Gamma values.

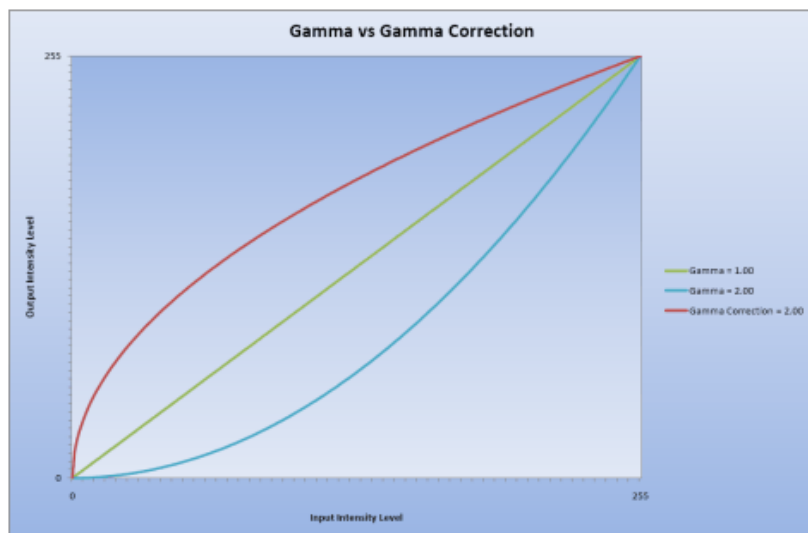


Gambar 5.1. Kurva Gamma

Notice in Figure 5.1 above that with Gamma being 1, the input value is the same as the output value and is shown in a straight line. To calculate Gamma Correction, the input value is raised to the inverse of the Gamma value. The formula for Gamma Correction is as follows:

$$I' = 255 \times \left( \frac{I}{255} \right)^{\frac{1}{\gamma}}$$

The following is a graph of the comparison between the gamma curve and the gamma correction curve:



The value used as Gamma depends on the application used. Here are some examples of RGB images before and after processing using Gamma Correction.



Gamma Correction with a Gamma value of 0.5



Gammar Correction with a Gamma value of 6

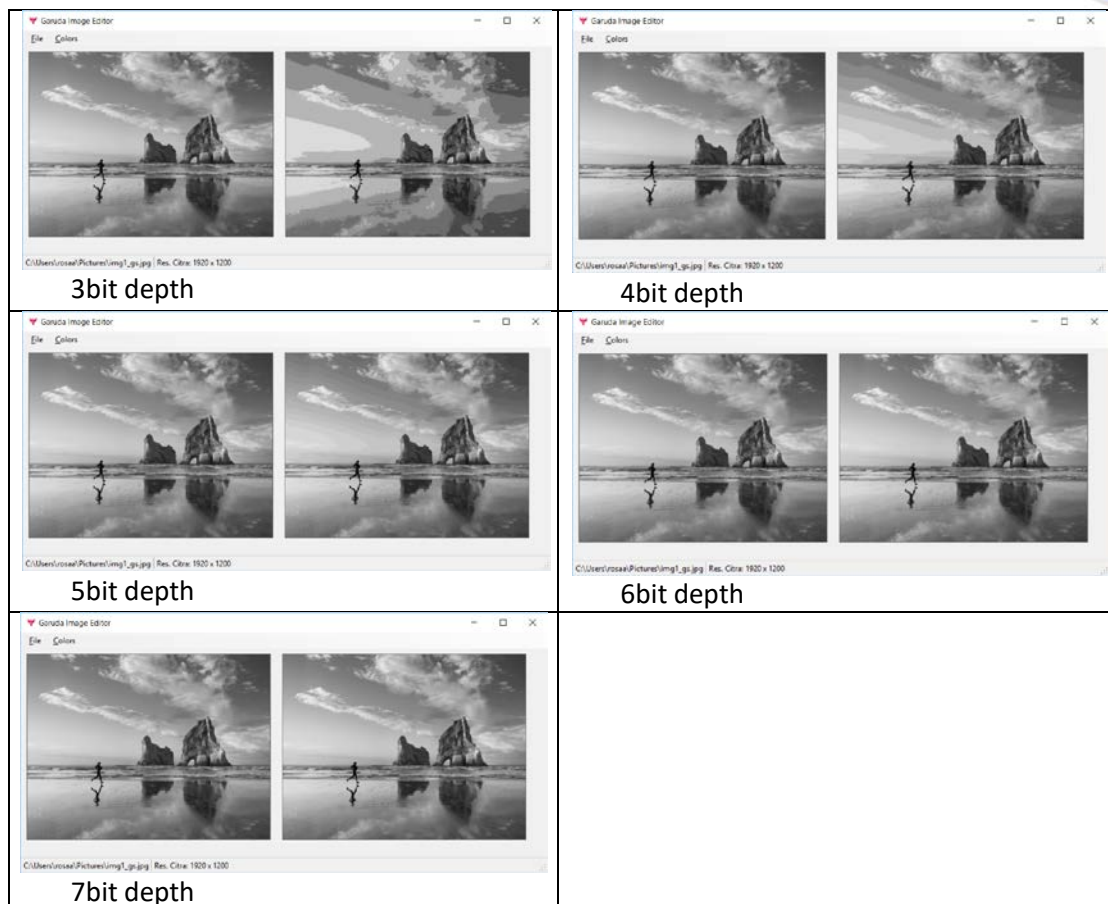
- Bit Depth

This operation is used to show you the quantization of the image. The quantization is done at a color depth value of 1 - 7bit. This transformation is a simulation only, the bits are not really 1-7 bits, only the color variations are 1-7 bits. Determine the depth level first, then the new color value is the result of the old color being operated with a predetermined level. Here is the formula used:

$$level = \frac{255}{2^{bit\_depth} - 1}$$

$$C' = round\left(\left(\frac{C}{level}\right) * level\right)$$





- PSNR

PSNR (Peak Signal-to-Noise Ratio) is the ratio between the maximum power value of the image and the power of the image that is exposed to noise which affects the quality of the noised image. To calculate the PSNR from an image, what needs to be done is to compare the denoised / compressed image with the ideal image that has the maximum power (in this case, the original image).

PSNR is formulated with the following formula:

$$PSNR = 10 \log_{10} \left( \frac{(L-1)^2}{MSE} \right) = 20 \log_{10} \left( \frac{L-1}{RMSE} \right)$$

Where L is the total possible color intensity of the image.

MSE is the mean squared error and is formulated as follows:

$$MSE = \frac{1}{MN} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (Original(i,j) - Denoised(i,j))^2$$

RMSE (Root Mean-Square-Error) is the root of MSE

The following are the functions of PSNR in Python

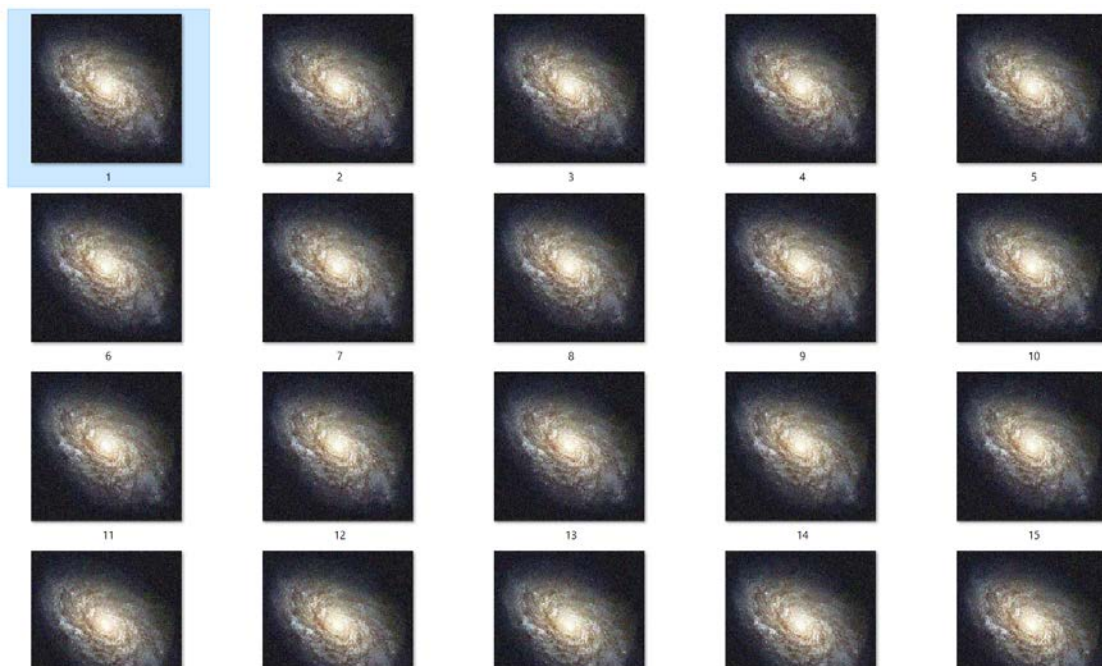
```
def PSNR(img1, img2):  
    mse = np.mean((img1 - img2) ** 2)  
    if(mse == 0): # MSE 0 maka tidak ada noise sama sekali, sehingga PSNR tidak memiliki arti  
        return 100  
    max_pixel = 255.0  
    psnr = 20 * log10(max_pixel / sqrt(mse))  
    return psnr
```

- Average Denoising

Average Denoising is an arithmetic operation applied to images for a variety of useful purposes. In this module, we will show the Averaging operation for denoising / reducing noise in the image. This operation calculates the average value of each pixel with the same coordinates in an image set. This operation is usually performed to remove noise from external factors that occur during image acquisition. Average denoising is commonly used in images that have external noise that occurs as a result of the distribution process from source to destination.

Average image works by calculating the average value of each pixel according to the number of images averaged. The more the number of images that are averaged, the better the denoised results, the higher the PSNR value. Note that the higher the PSNR value, the better the denoised image quality.

The following image is intentionally given Gaussian noise, randomly generated 100 times:



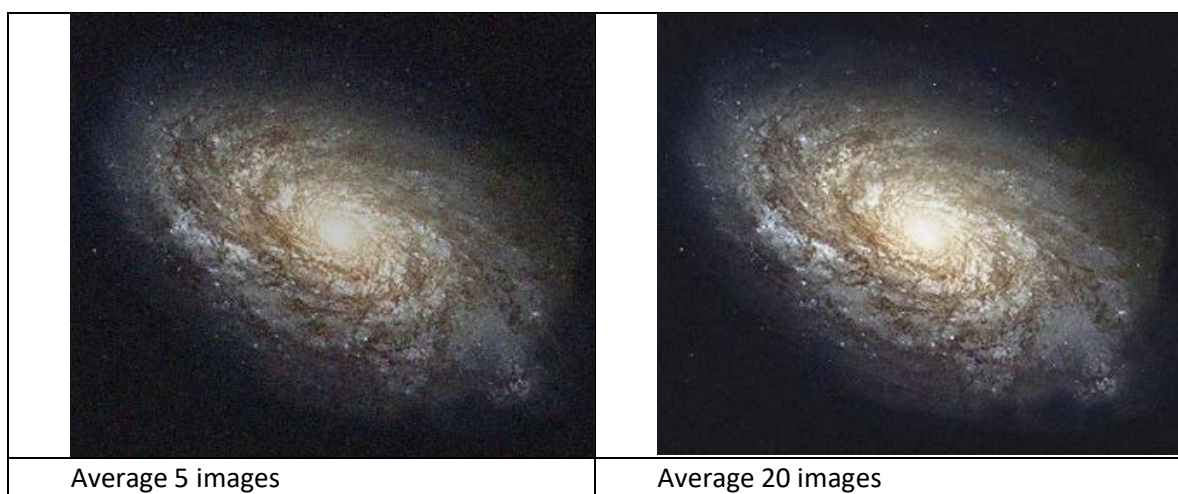
A set of Images that are given Gaussian Noise randomly

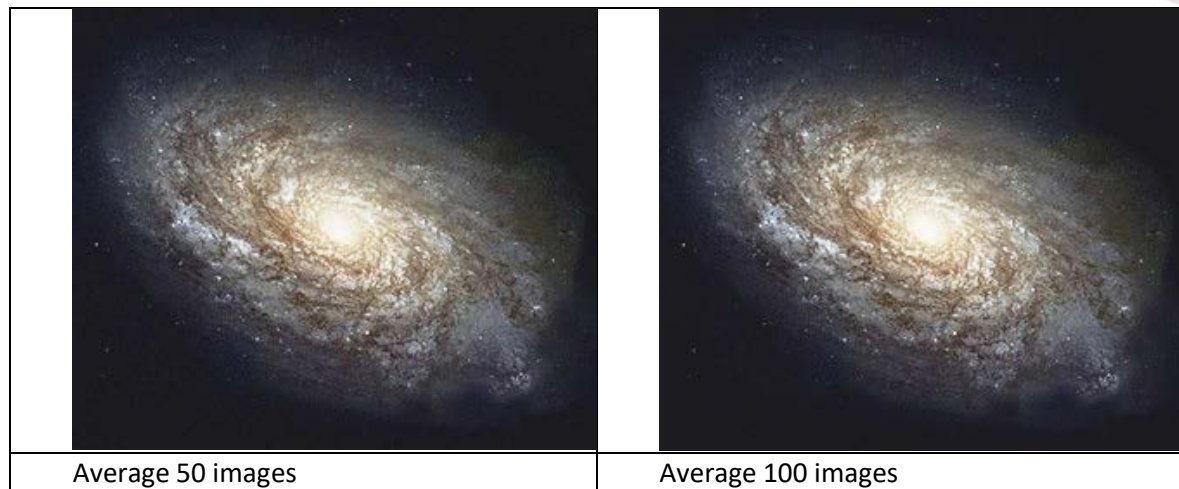




One of the images is enlarged and has Gaussian Noise applied

Following are the results of images that have been deenoised using averaging on 5 images, 20 images, 50 images, and 100 images.



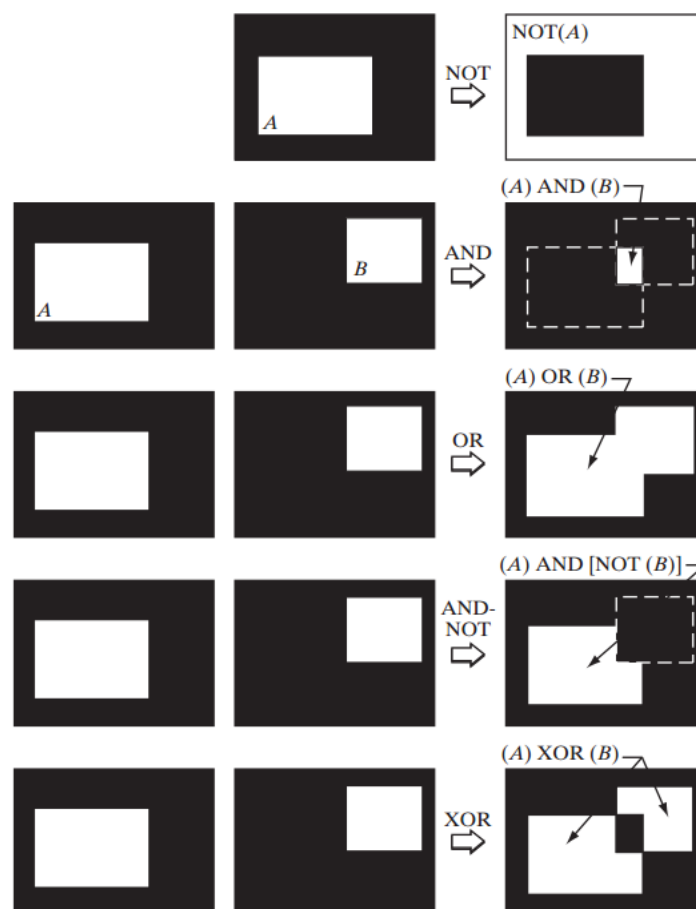


- Image Masking

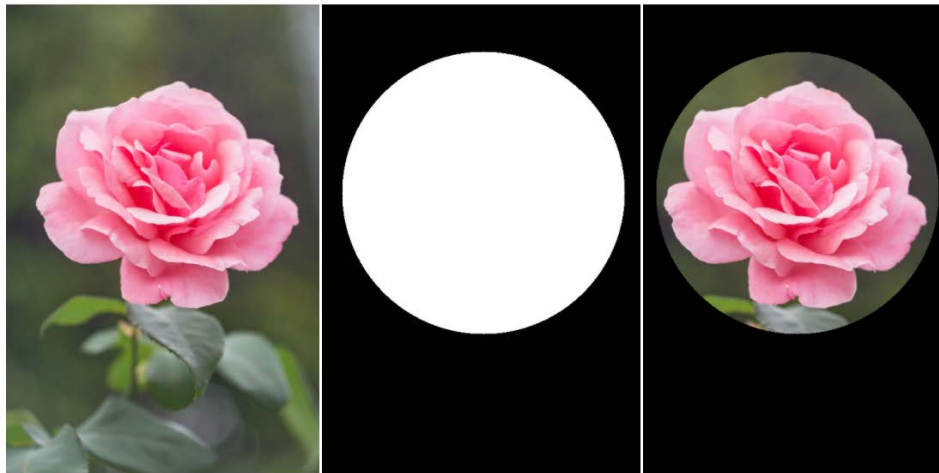
Image masking is an example of a Logic operation that is used to process images. Re-open Gonzales section 2.6.4 for a more detailed discussion of logical operations.

Some of the logical operators that are often used in image processing are the OR, AND, and NOT operators. In Image Masking, the operator usually used is the AND operator.

If it is assumed that two regions (sets) A and B consist of foreground pixels, then the OR operation of these two sets can contain the values of A or B. The following figure illustrates the Logical operations on the image.



The following image shows the image result using the image masking AND operator.



binary operation, n operator antara img a dan b

## F. TUGAS PRAKTIKUM

### 1. Make a Gamma Correction according to the following instructions

This experiment will ask you to make a Gamma Correction. In this experiment, the Gamma value will be set by asking for input from the user. Below is the code to request value input from the user. Continue the code by creating an image with gamma correction according to the formula given.

```
print(' Gamma Correction pada citra ')
print('-----')
try:
    gamma = int(input('Masukkan nilai Gamma: '))
except ValueError:
    print('Error, not a number')
```



Gamma Correction pada citra

Masukkan nilai Gamma: 3



The image above is an image that is processed using Gamma Correction, the gamma value of 3.

## 2. Create Image Depth Simulation

This experiment is used as a simulation of the image quantization process. In image quantization, pixels can be represented by n-bit depth (default is 8-bit). In an 8-bit pixel, the possible colors are 256 colors, from 0 (0000 0000) to 255 (1111 1111). On a 7-bit pixel, the possible colors are 128 colors, from 0 (000 0000) to 127 (111 1111). The possible color is obtained from the power of 2 the number of bits. If it is 7 bits, then the number of colors will be  $2^7 = 128$ , etc. Since Visual Studio 2017 only works on 8 bits, this experiment only manipulates colors so that the number of colors matches their depth. For the 7-bit case, two 8-bit colors are represented by one 7-bit color. Examples of color pixels 0 and 1 at 8-bit, represented by color 0 at 7-bit. color pixels 2 and 3 at 8-bit, represented by color 1 at 7-bit, etc.

## 3. Create the Average Denoising module

Create an average denoising module according to the formula given in the previous section. Original images are provided at `/images/galaxy.jpg`.


100 Images with Gaussian Noise are provided at `/images/noises/*.jpg`

You can use the following code to read all the images in one folder, use the glob module (`import glob`).

```
cv_img = []
for img in glob.glob('/content/drive/MyDrive/Polinema/Kuliah/PCVK/
Images/noises/*.jpg'):
    n= cv.imread(img)
    cv_img.append(n)
```

By using the code above, all you have to do is call the image in the folder using `cv_img [0]`, `cv_img [1]`, and so on.

Record the PSNR results in the following table. From the results you have recorded, write down your conclusions:

No	Number of Images in Average	Image Result	PSNR value (dB)
1.	5		
2.	30		
3.	60		
4	80		
5	100		

From the PSNR results that you recorded in the table above, the conclusions that can be drawn are..

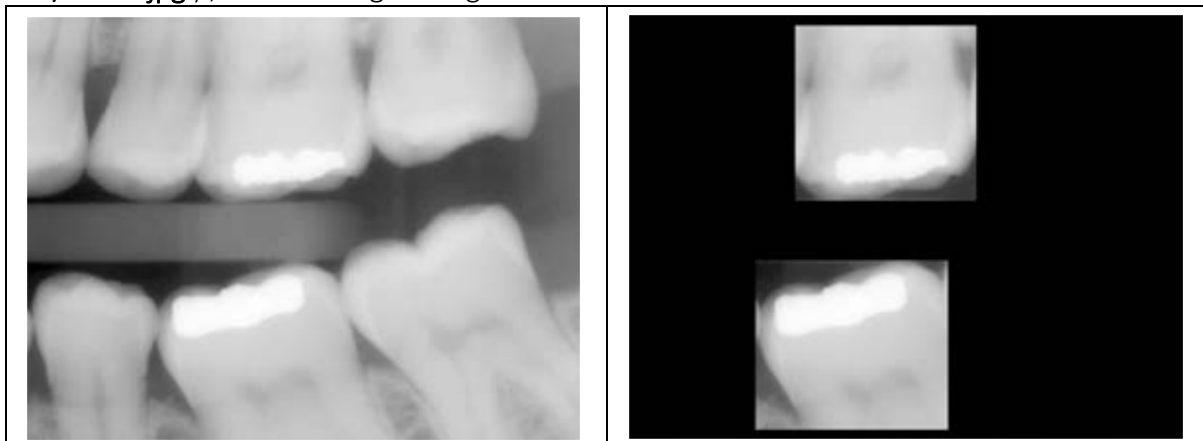
.....

.....

.....

.....

4. Create image masking for the following image. The left image is the original image ( `images / teeth.jpg` ), while the right image is the result:



masking, pakai n

1. Do the experiment using other operators and show the results in this module. Write down the results of your analysis why the output image shown like that.

No.	Operator	Image Input	Image Output
1.	<b>OR</b>		
2.	<b>NOT</b>		
3.	<b>NAND</b>		



4.	<b>XOR</b>		
----	------------	--	--

Write down the results of your analysis:

.....

.....

.....

.....

.....

.....

.....

.....

--- SELAMAT BELAJAR ---