CHAPTER 4

RESULT AND DISCUSSION

4.1 Read Image

After finishing and simulated the program with the attain image. This is the sample of the image that had been read from the work folder. There are 4 model consist of 33 pictures of each with 3-6 images in a same distance as stated in literature review. The images are taken from the inside the house.

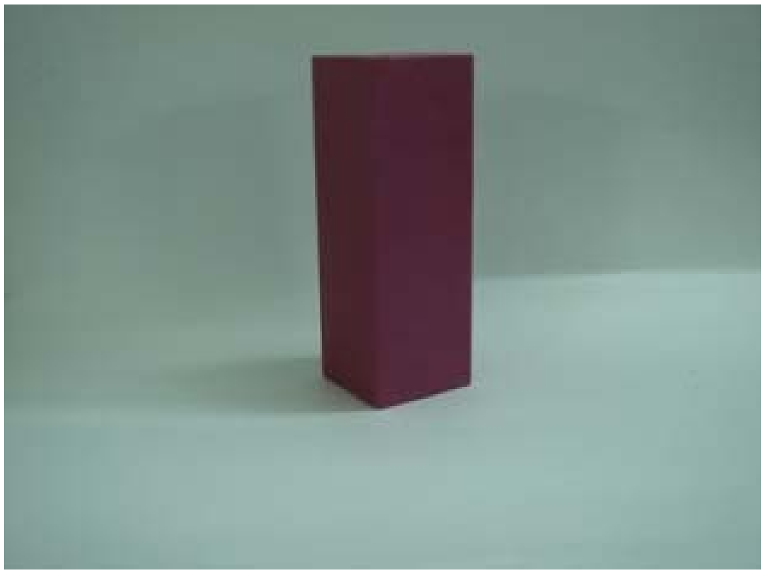


Figure 4.1 Image with resolution 1200x1600

The distance for this image is the initial activation of the system, which is 132.5cm. This image is taken using the digital camera instead of the web camera to get a clearer view of the picture.

The command ‘imwrite’ there is use to overwrite the previous command that is to call the images. This is to prevent image stacking as this will mess up the data acquisition. The digital cameras used are 3megapixel with no zooming addition. Therefore, the image is the default resolution of 1200-by-1600. This is because the image will be resized to 80-by-60.

4.1.1 Camera Image Captured

The web camera captured the image with set to the real time when the program is running through the MATLAB software, hence the image captured will be stored and process.

4.2 Image Resizing

The systems need to resize the entire image in 80-by-60 resolution because this will limit the information of the Singular Value Decomposition (SVD) from hundreds to less than 90. The image however, will not have any data loss after this resolution is resized as it can be seen through the grayscale and binary image. The binary image must show a good shape if the SIGMA and THRESHOLD value is accurate.

The reason why it uses this resolution is that when the image is processed, it will be smaller. But when the SVD value is acquired, it can take the whole values and use it for an input database without losing any values that might indicate the important part in the images. This will make the system more accurate and will read the whole picture instead of only ¾ or upper ½ of the image. The result is shown Figure 4.2:

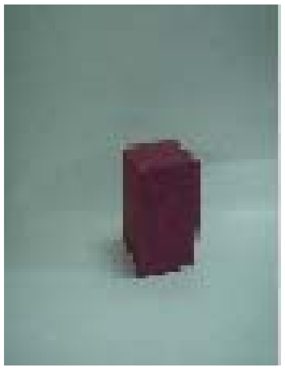


Figure 4.2 Resized image to 80-by-60

The image size is small, with a lot of salt and pepper noise in it. There is probability that unknown noise also available. This image is where the limitation to all the information begins as a simple image gives small information.

4.3 Image Conversion

The image is converted to grayscale (also known as intensity image) then to binary images. The function to convert a RGB image to grayscale is to change the image to 0 and 1 rather than having to process 256 color image. For grayscale, the image will be in black and white as can see from the Figure 4.3. This image is good for data processing in edge detection and SVD acquisition for the system.

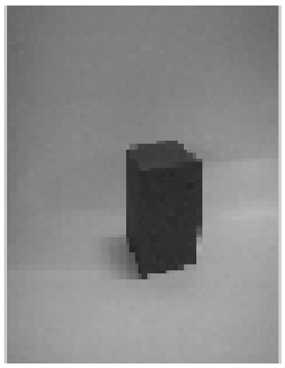


Figure 4.3 Grayscale image with resolution of 80-by-60

The image has the same value now as it is in black and white form. This is easier to process as the image can be read as 0 and 1 integer by the MATLAB program. The comparison of the images form is shown in Table 4.1:

Table 4.1 Comparison of RGB and Grayscale Image

RGB image	Grayscale Image
Image in which each pixel is specified by three values -- one each for the red, green, and blue components of the pixel's color. In MATLAB, an RGB image is represented by an m-by-n-by-3 array of class uint8, uint16, or double. This documentation uses the variable name RGB to represent an RGB image in the workspace. This type of image is also known as a true-color image.	Image consisting of intensity (grayscale) values. In MATLAB, intensity images are represented by an array of class uint8, uint16, or double. While intensity images are not stored with colormaps, MATLAB uses a system colormap to display them. This documentation uses the variable name to represent an intensity image in the workspace.

Next is the binary image. This is a binary coding (consist of 0 and 1 only) in the image. Usually, the foreground is white as it represents the value of 1 in the binary integer and the background is black where it represents the 0 value in binary. Refer to the image result in the Figure 4.4:

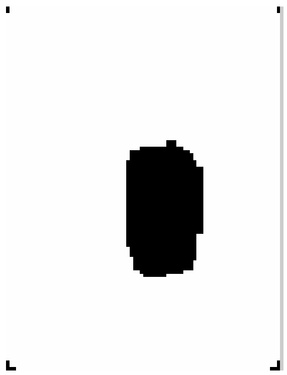


Figure 4.4 Binary Image with resolution of 80-by-60

The image of the model is quite dark but still can be seen. This is due to the threshold value that is not accurate. This can be corrected by examining different value for the threshold value. However, in Chapter 3 Methodology, the program is stated that will be using median filtering before showing the images. This is because in grayscale image, the systems need to remove all the noise before it is process. This is where median filtering came in. It is important that the area it covered for filtering is accurate. In the program, the method use medfilt2 function to remove the noise. ‘medfilt2’ is used to perform two-dimensional median filtering.

Description Median filtering is a nonlinear operation often used in image processing to reduce "salt and pepper" noise. Median filtering is more effective than convolution when the goal is to simultaneously reduce noise and preserve edges.

B = medfilt2(A,[m n]) performs median filtering of the matrix A in two dimensions. Each output pixel contains the median value in the m by n neighborhood around the corresponding pixel in the input image.

Medfilt2 pads the image with 0's on the edges, so the median values for the points within [m n]/2 of the edges might appear distorted. B = medfilt2(A) performs median filtering of the matrix A using the default 3-by-3 neighborhood. B = medfilt2 (A,'indexed',...) processes A as an indexed image, padding with 0's if the class of A is uint8, or 1's if the class of A is double. The input image A can be of class logical, uint8, uint16, or double (unless the 'indexed' syntax is used, in which case A cannot be of class uint16). The output image B is of the same class as A.

If the input image A is of an integer class, all the output values are returned as integers. If the number of pixels in the neighborhood (i.e., m*n) is even, some of the median values might not be integers. In these cases, the fractional parts are discarded. Logical input is treated similarly. For example, suppose it calls medfilt2 using 2-by-2 neighborhoods, and the input image is a uint8 array that includes this neighborhood.

$$\begin{matrix} 1 & 5 \\ 4 & 8 \end{matrix}$$

medfilt2 returns an output value of 4 for this neighborhood, although the true median is 4.5.

After the salt and pepper noise is removed, this is where the image can be process. This is because do not need any extra small edge as it will change the value of the SVD. The threshold value use is 0.3 with the pixel of 3-by-3. After this, the systems can get the image processes in any form desired and use the filtering to clear everything from noise so that it can later be use to display a good edge image.

4.4 Filtering Binary Image

The acquired images in this system have a lot of small unwanted particles. For each images, it seems that for the left images and the right images is not in the same light condition as the other is a bit brighter than the left images.

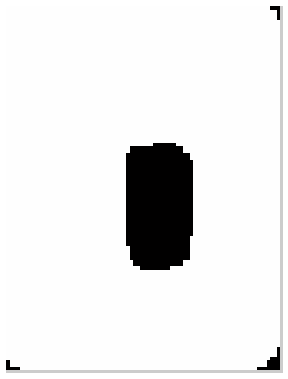


Figure 4.5 Filtered with resolution 80-by-60

As seen in Figure 4.5, the image is dark where the light is not reflected and the white is where all the light intensity is high. However, the image of the model is still shown as it is still having a high threshold value which is 0.4 (default value). The image is filtered with median of a group of pixel consist of 2 by 2 and it is filtered twice to make the system more clear without the worries about small particle of unwanted edge.

Filtering an image can take out all the unwanted noises in a picture. It is compulsory to get a good type of filtering as there are lot types of filter that can be use to remove noise. As explained in Chapter 3, it is important to compare first before applying into the program for the proper result.

4.5 Edge Image

Edge image is a black and white image that consist of the number 0 for black also known as background and 1 for white also known as foreground. The edges command used is Canny edge detector. Canny is a second derivative feature that says that a new edge detector that is optimal for step edges corrupted by white noise. Therefore, Canny method is one of the most powerful edge detection method that is related to 3 important criteria.

4.5.1 The detection criterion

It is required that important edge (in this context is the model shape) should never be missed.

4.5.2 The localization criterion

Distance between the actual and located position of edge should be minimal. This means that sometime a program cannot differentiate a model and its shadow. Canny edge can detect this but it must not be too far from the actual edge of the model.

4.5.2 The one response

There will be so many responses when it gets from filtering process. It will try to create boundaries between pixels. This means that the edge will be mess up due to uneven lighting in each pixel value. This should be minimized when responding to a single edge.

Overall, Canny edge is good when detecting the uneven edge to separate the weak edge and the strong edges. The result is shown on Figure 4.7:

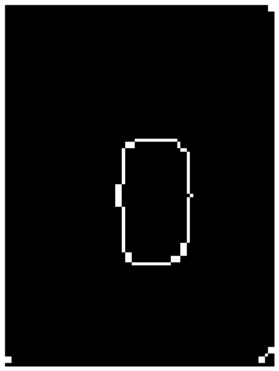
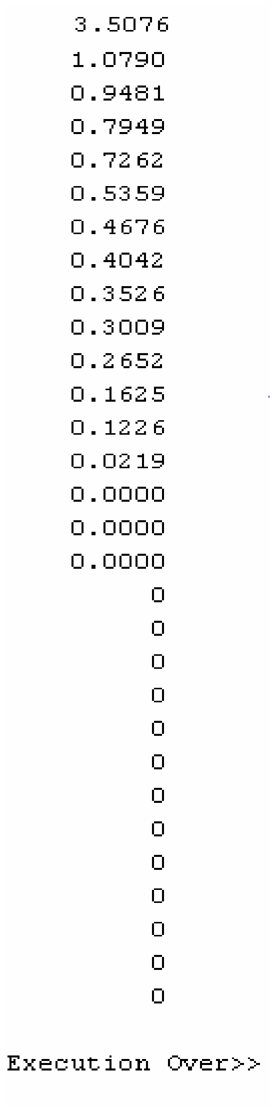


Figure 4.6 Canny Edge Image

4.6 Database

There are two type of database; input database and output database. These databases are obtained from the Singular Value Decomposition (SVD) value when it extracts the images pixel’s identities. From the program, notice that there is a fopen folder which is the input data and the fout to display the SVD value in command windows. The execution will be over after displaying a sum of number in the MATLAB command windows.



```
   3.5076
   1.0790
   0.9481
   0.7949
   0.7262
   0.5359
   0.4676
   0.4042
   0.3526
   0.3009
   0.2652
   0.1625
   0.1226
   0.0219
   0.0000
   0.0000
   0.0000
        0
        0
        0
        0
        0
        0
        0
        0
        0
        0
        0
        0
        0

Execution Over>>
```

Figure 4.7 Executions for SVD

In Figure 4.7, SVD display, the database only get 40 values consists of ¾ of the images identities. Therefore, the images already get its own identification that later on can be trained using the neural network. Before that, it is easier if the database is form using the Microsoft Excel file so that the neural network can call the command back from the database easily.