OPEN ACCESS

International
Journal
Of Modern Engineering Research (IJMER)

Real Time 2-D Convolution Layer for Feature Extraction

Panca Mudjirahardjo
[1]PhD –Dept. of Electrical Engineering, Faculty of Engineering, Universitas Brawijaya, Indonesia.
Corresponding Author

ABSTRACT:*Feature extraction is an important process in computer vision. A good feature can distinguish the shape of one object from another. It will increase the classification and detection accuracy. There are many method to build the object's feature. One of them using convolutio process. Nowadays, convolution neural network (CNN) is widely uased for object classification and detection. The convolution stage is used for feature extraction. In this paper, researcher will build and observe the real time 2-D convolution layer for feature extraction. The effect of types and size of filter kernel will be observed in this experiment.*

KEYWORDS: *Feature extraction, real-time, 2-D convolution, filter kernel.*

Date of Submission: 26-05-2024 Date of acceptance: 07-06-2024

I. INTRODUCTION

Nowadays, computer vision is widely used in many field areas for automatic system. The most common uses are for object classification, detection and recognition.In this system, the machine is expected to be able to distinguish and recognize objects captured by a camera.After the object is captured by camera, the next stage is pre-processing. The pre-processing consists of image enhancement and feature extraction.

Feature extraction stage is required to show the different points of object classes and to reduce the computation time. One of the interest platform for object detection and classification is CNN based. In CNN consists of several layer of convolution process to extract feature.

The distinctive object will increase the classification accuracy. This property is often difficult to obtain because the light intensity on an object is not uniform. Sometimes the objects get enough light, sometimes not enough light. One way to overcome this lack of light is to process histogram images, namely histogram equalization.Histogram equalization stretches the histogram distribution so that it does not accumulate at certain light intensities.One of these histogram equalization methods is CLAHE.

Object'sfeature can be obtained from the convolution process of the input image with a filter.Usually the output of this convolution process is the detection of edges of an object in certain directions according to the filter used.So that a filter corresponds to certain directions.These particular directions can be obtained by increasing the amplitude of the desired direction and suppressing the amplitude of the undesired direction.

II. METHODS

In this section, we briefly explain the proposed method to study the effect of image format and filter images in convolution process. The method is shown in Figure 1.

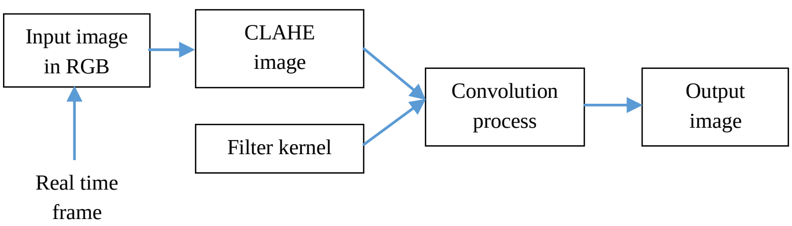


Figure 1. The proposed method of this study

In Figure 1, the input image from static image from a file and real time frame captured by a camera. Then the RGB image converts into grayscale image. This grayscale image, then convert into CLAHE image for contrast enhancement with histogram equalization. Convolution process requires two images, this is done by an input image in CLAHE format and a filter kernel. The result of convolution process can be shown in output image.

2.1. Contrast Limited Adaptive Histogram Equalization (CLAHE)

Adaptive histogram equalization (AHE) is an image pre-processing technique used to improve contrast in images. It computes several histograms, each corresponding to a distinct section of the image, and uses them to redistribute the luminance values of the image. It is therefore suitable for improving the local contrast and enhancing the definitions of edges in each region of an image. However, AHE has a tendency to over amplify noise in relatively homogeneous regions of an image. A variant of adaptive histogram equalization called contrast-limited adaptive histogram equalization (CLAHE) prevents this effect by limiting the amplification.

Ordinary AHE tends to overamplify the contrast in near-constant regions of the image, since the histogram in such regions is highly concentrated. As a result, AHE may cause noise to be amplified in near-constant regions. Contrast Limited AHE (CLAHE) is a variant of adaptive histogram equalization in which the contrast amplification is limited, so as to reduce this problem of noise amplification[1-5].

In CLAHE, the contrast amplification in the vicinity of a given pixel value is given by the slope of the transformation function. This is proportional to the slope of the neighborhood cumulative distribution function (CDF) and therefore to the value of the histogram at that pixel value. CLAHE limits the amplification by clipping the histogram at a predefined value before computing the CDF. This limits the slope of the CDF and therefore of the transformation function. The value at which the histogram is clipped, the so-called clip limit, depends on the normalization of the histogram and thereby on the size of the neighborhood region. Common values limit the resulting amplification to between 3 and 4.

It is advantageous not to discard the part of the histogram that exceeds the clip limit but to redistribute it equally among all histogram bins[1].

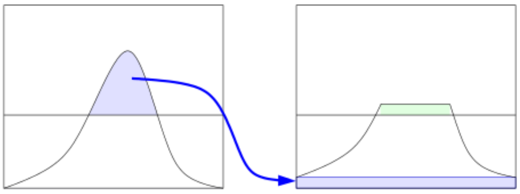


Figure 2. **The histogram distribution in CLAHE [1]**

The redistribution will push some bins over the clip limit again (region shaded green in the Figure 2), resulting in an effective clip limit that is larger than the prescribed limit and the exact value of which depends on the image. If this is undesirable, the redistribution procedure can be repeated recursively until the excess is negligible.

The CLAHE algorithm has three major parts: tile generation, histogram equalization, and bilinear interpolation. The input image is first divided into sections. Each section is called a tile. Histogram equalization is then performed on each tile using a pre-defined clip limit. Histogram equalization consists of five steps: histogram computation, excess calculation, excess distribution, excess redistribution, and scaling and mapping using a cumulative distribution function (CDF). The histogram is computed as a set of bins for each tile. Histogram bin values higher than the clip limit are accumulated and distributed into other bins. CDF is then calculated for the histogram values. CDF values of each tile are scaled and mapped using the input image pixel values. The resulting tiles are stitched together using bilinear interpolation, to generate an output image with improved contrast.

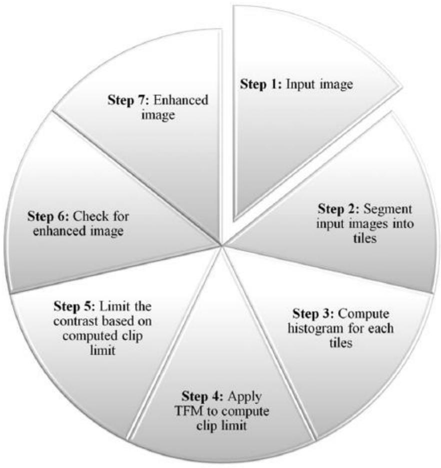


Figure 3. **Steps followed in CLAHE algorithm [5].**

To increase image contrast, use the CLAHE algorithm as below. Grayscale and color photos can both be processed using this approach.CLAHE algorithm steps are as follows [5]:

Step 1: Input image
Step 2: Segment input images into tiles
Step 3: Compute histogram for each tiles
Step 4: Apply TFM to compute clip limit
Step 5: Limit the contrast based on computed clip limit
Step 6: Check for enhanced image
Step 7: Enhanced image

Figure 3 illustrates the steps to be followed in the CLAHE algorithm. Prior to creating a histogram for each context region, a given input image is first separated into context regions. So that various portions of the image may be easily linked, a mapping function is used to produce an image mapping. The image noise is subsequently reduced using an interpolation approach. This enables us to lessen the noise in particular regions of the image. Although the method denoises the image, it does not do so fully.

2.2. Filter Kernel.

A filter, or kernel, in a CNN is a small matrix of weights that slides over the input data (such as an image), performs element-wise multiplication with the part of the input it is currently on, and then sums up all the results into a single output pixel. This process is known as convolution.

Filters are at the heart of what makes CNNs work. They are the primary component that helps the model extract useful features from the input data.

Prewitt Filters

$$\underset{\text{horizontal}}{\begin{bmatrix}-1 & 0 & 1\\-1 & 0 & 1\\-1 & 0 & 1\end{bmatrix}} \quad \underset{\text{vertical}}{\begin{bmatrix}1 & 1 & 1\\0 & 0 & 0\\-1 & -1 & -1\end{bmatrix}}$$

Sobel Filters

$$\underset{\text{horizontal}}{\begin{bmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{bmatrix}} \quad \underset{\text{vertical}}{\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}}$$

Laplacian Filters

$$\begin{bmatrix}-1 & -1 & -1\\-1 & 8 & -1\\-1 & -1 & -1\end{bmatrix}$$

Robinson Compass Masks

$$\underset{\text{north-west}}{\begin{bmatrix}-2 & -1 & 0\\-1 & 0 & 1\\0 & 1 & 2\end{bmatrix}} \quad \underset{\text{north}}{\begin{bmatrix}-1 & -2 & -1\\0 & 0 & 0\\1 & 2 & 1\end{bmatrix}} \quad \underset{\text{north-east}}{\begin{bmatrix}0 & -1 & -2\\1 & 0 & -1\\2 & 1 & 0\end{bmatrix}} \quad \underset{\text{east}}{\begin{bmatrix}1 & 0 & -1\\2 & 0 & -2\\1 & 0 & -1\end{bmatrix}}$$

$$\underset{\text{south-east}}{\begin{bmatrix}2 & 1 & 0\\1 & 0 & -1\\0 & -1 & -2\end{bmatrix}} \quad \underset{\text{south}}{\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}} \quad \underset{\text{south-west}}{\begin{bmatrix}0 & 1 & 2\\-1 & 0 & 1\\-2 & -1 & 0\end{bmatrix}} \quad \underset{\text{west}}{\begin{bmatrix}-1 & 0 & 1\\-2 & 0 & 2\\-1 & 0 & 1\end{bmatrix}}$$

Krisch Compass Masks

$$\underset{\text{north-west}}{\begin{bmatrix}-3 & 5 & 5\\ -3 & 0 & 5\\ -3 & -3 & -3\end{bmatrix}} \underset{\text{north}}{\begin{bmatrix}-3 & -3 & 5\\ -3 & 0 & 5\\ -3 & -3 & 5\end{bmatrix}} \underset{\text{north-east}}{\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & 5\\ -3 & 5 & 5\end{bmatrix}} \underset{\text{east}}{\begin{bmatrix}-3 & -3 & -3\\ -3 & 0 & -3\\ 5 & 5 & 5\end{bmatrix}}$$

$$\underset{\text{south-east}}{\begin{bmatrix}-3 & -3 & -3\\ 5 & 0 & -3\\ 5 & 5 & -3\end{bmatrix}} \underset{\text{south}}{\begin{bmatrix}5 & -3 & -3\\ 5 & 0 & -3\\ 5 & -3 & -3\end{bmatrix}} \underset{\text{south-west}}{\begin{bmatrix}5 & 5 & -3\\ 5 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}} \underset{\text{west}}{\begin{bmatrix}5 & 5 & 5\\ -3 & 0 & -3\\ -3 & -3 & -3\end{bmatrix}}$$

Figure 4. Types of filter 3×3 used in this experiment[7].

$$\begin{bmatrix}-1 & 0 & 1\\ -2 & 0 & 2\\ -1 & 0 & 1\end{bmatrix}\begin{bmatrix}-5 & -4 & 0 & 4 & 5\\ -8 & -10 & 0 & 10 & 8\\ -10 & -20 & 0 & 20 & 10\\ -8 & -10 & 0 & 10 & 8\\ -5 & -4 & 0 & 4 & 5\end{bmatrix}$$

(a)

$$\begin{bmatrix}-1 & -1 & -1\\ 0 & 0 & 0\\ +1 & +1 & +1\end{bmatrix}\begin{bmatrix}-2 & -2 & -2 & -2 & -2\\ -1 & -1 & -1 & -1 & -1\\ 0 & 0 & 0 & 0 & 0\\ +1 & +1 & +1 & +1 & +1\\ +2 & +2 & +2 & +2 & +2\end{bmatrix}$$

(b)

Figure 5. Filter kernel 3×3 and 5×5 (a) Sobel filter horizontal (b)Prewitt filter vertical

2.3. 2D Convolution Operation

Convolution is a technique widely used in image processing. Convolution is a mathematical operation on two functions that produces a third function expressing how the shape of one is modified by the other.

Many image processing results come from a modification of one pixel with respect to its neighbors. When this modification is similar in the entire image g, it can be mathematically defined using a second image h which defines the neighbor relationships. This results in a third image f. This is the so-called convolution and it is denoted with $*$:

$$f(x,y) = (g * h)(x,y) = \sum_m \sum_n g(x-m, y-n)\, h(m,n) \tag{1}$$

Intuitively, the convolution "spreads" each pixel $(m,)$ in g following h and proportionally to the intensity $g(m,n)$. Figure6 gives an example of the computing of a particular pixel.

$$\begin{array}{|c|c|c|c|c|}\hline f_{1,1} & f_{1,2} & f_{1,3} & f_{1,4} & f_{1,5}\\ \hline f_{2,1} & f_{2,2} & f_{2,3} & f_{2,4} & f_{2,5}\\ \hline f_{3,1} & f_{3,2} & f_{3,3} & f_{3,4} & f_{3,5}\\ \hline f_{4,1} & f_{4,2} & f_{4,3} & f_{4,4} & f_{4,5}\\ \hline f_{5,1} & f_{5,2} & f_{5,3} & f_{5,4} & f_{5,5}\\ \hline\end{array} = \begin{array}{|c|c|c|c|c|}\hline g_{1,1} & g_{1,2} & g_{1,3} & g_{1,4} & g_{1,5}\\ \hline g_{2,1} & g_{2,2} & g_{2,3} & g_{2,4} & g_{2,5}\\ \hline g_{3,1} & g_{3,2} & g_{3,3} & g_{3,4} & g_{3,5}\\ \hline g_{4,1} & g_{4,2} & g_{4,3} & g_{4,4} & g_{4,5}\\ \hline g_{5,1} & g_{5,2} & g_{5,3} & g_{5,4} & g_{5,5}\\ \hline\end{array} * \begin{array}{c|c|c|c|c} & -1 & 0 & +1 & \\ \hline & h_{-,-} & h_{-,0} & h_{-,+} & -1\\ \hline & h_{0,-} & h_{0,0} & h_{0,+} & 0\\ \hline & h_{+,-} & h_{+,0} & h_{+,+} & +1\\ \hline\end{array}$$

$$f_{2,2} = g_{3,3}h_{-,-} + g_{3,2}h_{-,0} + g_{3,1}h_{-,+} + g_{2,3}h_{0,-} + g_{2,2}h_{0,0} + g_{2,1}h_{0,+} + g_{1,3}h_{+,-} + g_{1,2}h_{+,0} + g_{1,1}h_{+,+}$$

Figure 6. Example for computing the pixel (2,2) of f [6].

III. RESULTS

In this section, the experimental procedure and result are briefly explain. This experiment is performed using programming language C++ and openCV library.
The programming code to convert the RGB input image into grayscale is:

```
cvtColor(imgOriginal,imgGrey,COLOR_BGR2GRAY);
```

The programming code to convert the grayscale image into CLAHE image with clip limit 4, are:

```
Ptr<CLAHE> clahe = createCLAHE();
clahe->setClipLimit(4);
clahe->apply(imgGrey,imgClahe);
```

To create the filter kernel 3×3 is as follows:

```
kernelPFH = (Mat_<int>(3,3) << -1, 0, 1, -1, 0, 1, -1, 0, 1);        //Prewitt filter horizontal
kernelPFV = (Mat_<int>(3,3) << 1, 1, 1, 0, 0, 0, -1, -1, -1);        //Prewitt filter vertical
```

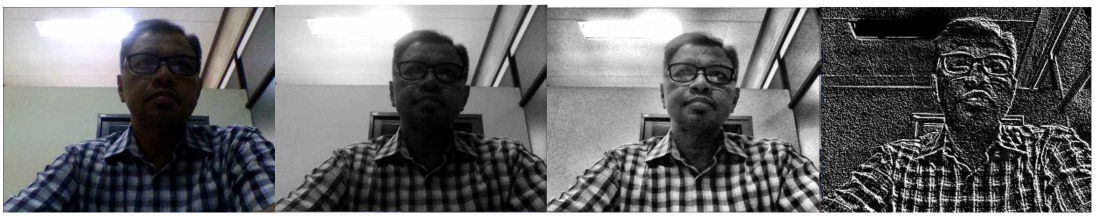
Then the convolution process is performed in `filter2D(src,dst,ddepth,kernel,anchor,delta,BORDER_DEFAULT)` as:

```
filter2D(imgClahe,output, -1 , kernel, Point(-1, -1), 0, 4);
```

where the arguments denote:

- *src* : source image.
- *dst* : destination image.
- *ddepth* : the depth of *dst*. A negative value (such as -1) indicates that the depth is same as the source.
- *kernel* : the kernel to be scanned through the image.
- *anchor* : the position of the anchor relative to its kernel. The location *Point(-1,-1)* indicates the center by default.
- *delta* : a value to be added to each pixel during the correlation. By default it is 0.
- *BORDER_DEFAULT* : we let this value by default.

The result images are depicted in Figure 7 and 8. Figure 7, the experiment is performed at light and no light condition. It is shown that the CLAHE image can improve the contrast image and affecting to the convolution result.



(a) (b)(c) (d)

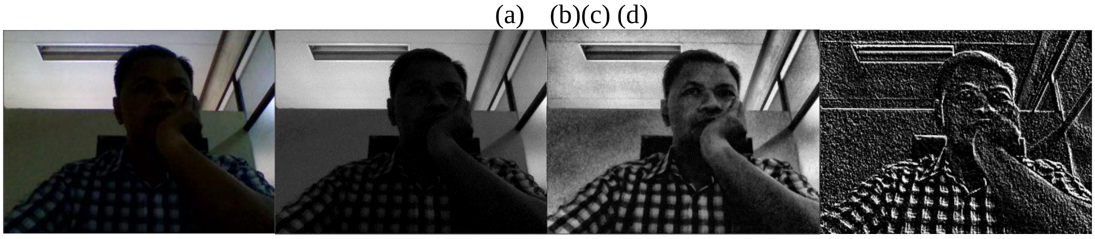


Figure 7.Processing real time video, (top) at light condition, (bottom) no light condition, (a) original frame (b) grayscale frame (c) CLAHE frame (d) convolved frame.

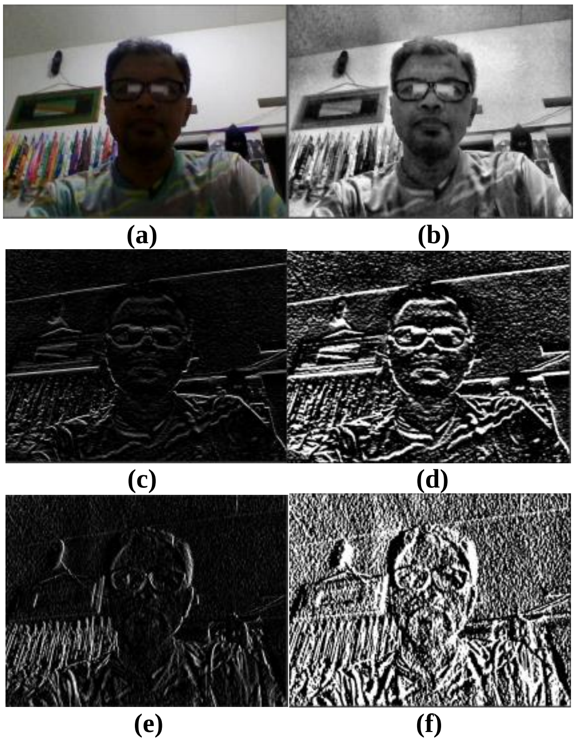


Figure 8.The original and convolved image, (a) original image, (b) CLAHE image, (c) result of Prewitt filter 3×3 (d) 5×5 (e) result of Sobelfilter 3×3(f) 5×5.

IV. CONCLUSIONS

From the above study, the implementation of CLAHE image improves the image contrast then increase the texture pattern of convolution output. This texture pattern is almost same at light and no light condition. This output becomes a feature for the next classifier stage. The distinctive pattern will become the feature of the object. The effect of filter size is the difference texture pattern.

REFERENCES

[1] S. M. Pizer, E. P. Amburn, J. D. Austin, et al.: *Adaptive Histogram Equalization and Its Variations*. Computer Vision, Graphics, and Image Processing 39. 1987.Pp. 355-368.

[2] K. Zuiderveld: *Contrast Limited Adaptive Histogram Equalization*. In: P. Heckbert: *Graphics Gems IV*, Academic Press 1994, ISBN 0-12-336155-9.

[3] T. Sund & A. Moystad: *Sliding window adaptive histogram equalization of intra-oral radiographs: effect on diagnostic quality*. Dentomaxillofac Radiol. 2006 May;35(3):133-8.

[4] G. R. Vidhya and H. Ramesh, *Effectiveness of contrast limited adaptive histogram equalization technique on multispectral satellite imagery*, Proc. Int. Conf. Video Image Process., Dec. 2017.pp. 234-239.

[5] Venkatesh S., John De Britto C., Subhashini P., Somasundaram K. *Image Enhancement and Implementation of CLAHE Algorithm and Bilinear Interpolation*. Cybernetics and systems: an International Journal.2022.

[6] Vincent Mazet. *Convolution*. Basics of Image Processing — (Université de Strasbourg), 2020-2024 https://vincmazet.github.io/bip/filtering/convolution.html

[7] *Filters in convolutional neural networks*, 2022. https://blog.paperspace.com/filters-in-convolutional-neural-networks/.