EasyChair Preprint

№ 4576

Orthogonality in Image Processing

Pankaj Kumar Saini

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

November 16, 2020

ORTHOGONALITY IN IMAGE PROCESSING

Pankaj kumar saini

MIT2020117@iiita.ac.in

Indian Institute of Information Technology, Prayagraj

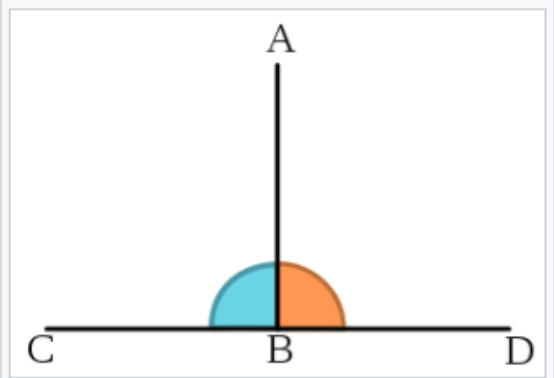
Abstract

Orthogonality means two vectors are perpendicular to each other and due to this we can get to know that those two vectors are independent to each other. This paper is devoted towards the orthogonality applications in image processing.

Keyword: - orthogonality, image-processing, vectors.

Introduction

In mathematics, orthogonality is the generalization of the notion of perpendicularity to the linear algebra of bilinear forms. Two elements u and v of a vector space with bilinear form B are orthogonal when B(u, v) = 0. Depending on the bilinear form, the vector space may contain non-zero self-orthogonal vectors. In the case of function spaces, families of orthogonal functions are used to form a basis.



The line segments AB and CD are orthogonal to each other.

By extension, orthogonality is also used to refer to the separation of specific features of a system. The term also has specialized meanings in other fields including art and chemistry.

Definitions

- In geometry, two Euclidean vectors are orthogonal if they are perpendicular, i.e., they form a right angle.
- Two vectors, x and y, in an inner product space, V, are orthogonal if their inner product <x, y> is zero.
- Two vector subspaces, A and B, of an inner product space V, are called orthogonal subspaces if each other in A is orthogonal to a given subspace is its orthogonal complement.

A set of vectors in an inner product space is called pairwise orthogonal if each pairing of them is orthogonal. Such a set is called an orthogonal set.

In certain cases, the word normal is used to mean orthogonal, particularly in the geometric sense as in the normal to a surface. for example, the y-axis is normal to the curve y = x^2 at the origin. However, normal may also refer to the magnitude of a vector. In particular, a set is called orthonormal (orthogonal plus normal) if it is an orthogonal set of unit vectors. As a result, use of the term normal to mean "orthogonal" is often avoided. The word

"normal" also has a different meaning in probability and statistics.

Orthogonality in programming language design is the ability to use various language features in arbitrary combinations with consistent results. This usage was introduced by Van Wijngaarden in the design of Algol 68.

The number of independent primitives' concepts has been minimized in order that the language be easy to describe, to learn, and to implement. On the other hand, these concepts have been applied "orthogonally" in order to maximize the expressive power of the language while trying to avoid deleterious superfluities.

Orthogonality in statistics

When performing statistical analysis, independent variables that affect a particular dependent variable are said to be orthogonal if they are uncorrelated, since the covariance forms an inner product. In the case the same results are obtained for the effect of any if the independent variables upon the dependent variables, regardless of whether one models the effects of the variables individually with simple regression or simultaneously with multiple regression. If correlation is present, the factors are not orthogonal and different results are obtained by the two methods. This usage arises from the fact that if centered by subtracting the expected value (the mean), uncorrelated variables are orthogonal in the geometric sense discussed above, both as observed data (i.e., vectors) and as random variables (i.e., density functions). One econometric formalism that is alternative to the maximum likelihood framework, the generalized method of moments, relies on orthogonality conditions. In particular, the ordinary least squares estimator may be easily derived from an orthogonality condition between the explanatory variables and model residuals.

Orthogonal means that the inner product is zero. For example, in the case of using dot product as your inner product, two perpendicular vectors are orthogonal. Orthogonal means these vectors have been normalized so that their length is 1.

Orthogonal vectors are useful for creating a basis for a space. This is because every point in the space can be represented as a (linear) combination of the vectors. So for example in 3D space, x = [0, 0, 1], y = [1, 0, 0] form an orthonormal basis. No component of x can be represented with a component of the other vectors. This is because they are linearly independent.

This is different type of independence to statistical dependence however.

Correlation is a different notion. There are different ways of representing the correlation of two vectors (random variables X and Y).

$$corr(X,Y) = \frac{cov(X,Y)}{\sigma_x \sigma_y} = \frac{E[X-\mu_x]E[Y-\mu_y]}{\sigma_x \sigma_y}$$

Correlation

- A correlation is a statistic intended to quantify the strength of the relationship between two variables.
- A challenge in measuring correlation is that the variables we want to compare are often not expressed in the same units. And even if they are in the same units, they come from different distributions.
- There are two common solutions to these problems:

  1. Transform each value to a standard score, which is the number of standard deviations from the mean. This transform leads to the

"Pearson product-moment correlation coefficient."

2. Transform each value to its rank, which is its index in the sorted list of values. This transform leads to the "Spearman rank correlation coefficient."

Covariance

- Covariance is a measure of the tendency of two variables to vary together.
- If you have studied linear algebra, you might recognize that Cov is the dot product of the deviations, divided by their length. So, the covariance is maximized if the two vectors are identical, 0 if they are orthogonal, and negative if they point in opposite directions.

Another measure of correlation is cross-correlation. This is measure of the similarity of two functions (or vectors).

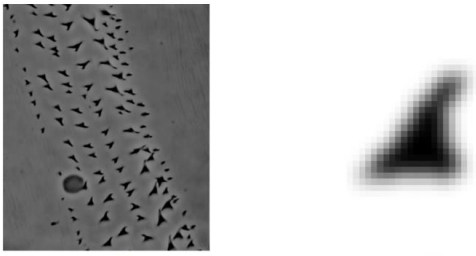
These concepts are used in many different ways in image processing.

For example, cross-correlation is used in template matching when looking for a small image inside a much larger image, the small template image is 'slided' over the large image and the cross-correlation is computed for each position. Locations with a high cross-correlation Are likely to contain the image we are looking for.

The concept of linearly independent component vectors is used in principal component analysis (PCA). PCA takes a cloud of points in space and calculates a set of orthogonal basis vectors to represent them.

Independent component analysis (ICA) is used for separating mixed signals. i.e., separating a speaker at a noisy cocktail party. ICA used properties of the signal correlation to separate the signals.

1 Image Correlation The image in figure 1(a) shows a detail of the ventral epidermis of a fruit fly embryo viewed through a microscope. Biologists are interested in studying the shapes and arrangement of the dark, sail-like shapes that are called denticles. A simple idea for writing an algorithm to find the denticles automatically is to create a template T, that is, an image of a typical denticle. Figure 1(b) shows a possible template, which was obtained by blurring (more on blurring later) a detail out of another denticle image. One can then place the template at all possible positions (r, c) of the input image I and somehow measure the similarity between the template T and a window W(r, c) out of I, of the same shape and size as T. Places where the similarity is high are declared to be denticles, or at least image regions worthy of further analysis.



(a) (b)

Figure 1: (a) Denticles on the ventral epidermis of a *Drosophila* embrio. Image courtesy of Daniel Kiehart, Duke University. (b) A denticle template.

Template matching is a method for searching and finding the location of a template image in a large image. OpenCV comes with a function cv2.matchTemplate() for this purpose simply slides the template image over the input image (as in 2D convolution) and compares the template and patch of input image under the template image. Several comparison methods are implemented in OpenCV. (you can check docs for more details). It returns a grayscale image, where each pixel denotes how much does the neighbourhood of that pixel match with template.

If input image is of size (WxH) and template is of size (wxh), output image will have a size of (W-w+1, H-h+1). Once you got the result, you can use cv2.minMaxLoc() function to find where is the maximum/minimum value. Take it as the top-left corner of rectangle and take (w, h) as width and height of the rectangle is your region of template.

Time Complexity:

Template matching is a technique in digital image processing for finding small parts of an image which match a template image. It can be used in manufacturing as a part of quality control, a way to navigate a mobile robot, or as a way to detect edges in images.

To match 2 2D arrays, to find all sub arrays, it will take m*n*o*p. where m*n is first array and o*p is second array.

Application in Real World:

In most computer vision and image analysis problems, it is necessary to define a similarity measure between two or more different objects or images. Template matching is a classic and fundamental method used to score similarities between objects using certain mathematical algorithms. These models have broad applications in image registration, and they are a fundamental aspect of novel machine vision or deep learning algorithms, such as convolutional neural networks (CNN), which perform shift and scale invariant functions followed by classification. In general, although template matching methods have restrictions which limit their application, they are recommended for use with other object recognition methods as pre- or post-processing steps. Combining a template matching technique such as normalized cross-correlation or dice coefficient with a robust decision-making algorithm yields a significant improvement in the accuracy rate for object detection and recognition.

In simple implementation, it is assumed that the escribed method is applied on grey images. This is why grey is used as pixel intensity. The final position in this implementation gives the top left location for where the template image best matches the search image.

Conclusion:

Orthogonality is simple and effective method to et more from image processing. A basic method of template matching uses an image patch (template), tailored to a specific feature of the search image, which we want to detect. This technique can be easily performed on grey images or edge images. The cross-correlation output will be highest at places where the image structure matches the mask structure, where large image values get multiplied by large mask values.

References:

[1].
https://stackoverflow.com/questions/22546325/worst-time-complexity-for-2d-patten-matching
[2].https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_template_matching/py_template_matching.html
[3].https://en.wikipedia.org/wiki/Orthogonality
[4].https://dsp.stackexchange.com/questions/11507/concept-of-orthogonality-and-orthonormality
[5].
https://www2.cs.duke.edu/courses/spring19/compsci527/notes/convolution-filtering.pdf

APPENDIX

IMPLEMENTATION :

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('messi5.jpg',0)
img2 = img.copy()
template = cv2.imread('template.jpg',0)
w, h = template.shape[::-1]

# All the 6 methods for comparison in a list
methods = ['cv2.TM_CCOEFF', 'cv2.TM_CCOEFF_NORMED', 'cv2.TM_CCORR',
            'cv2.TM_CCORR_NORMED', 'cv2.TM_SQDIFF', 'cv2.TM_SQDIFF_NORMED']

for meth in methods:
    img = img2.copy()
    method = eval(meth)

    # Apply template Matching
    res = cv2.matchTemplate(img,template,method)
    min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res)

    # If the method is TM_SQDIFF or TM_SQDIFF_NORMED, take minimum
    if method in [cv2.TM_SQDIFF, cv2.TM_SQDIFF_NORMED]:
        top_left = min_loc
    else:
        top_left = max_loc
    bottom_right = (top_left[0] + w, top_left[1] + h)

    cv2.rectangle(img,top_left, bottom_right, 255, 2)

    plt.subplot(121),plt.imshow(res,cmap = 'gray')
    plt.title('Matching Result'), plt.xticks([]), plt.yticks([])
    plt.subplot(122),plt.imshow(img,cmap = 'gray')
    plt.title('Detected Point'), plt.xticks([]), plt.yticks([])
    plt.suptitle(meth)

    plt.show()
```