Local self similarity histogram

(LSS)

Hariom Verma (CS171026)
[hariom18599@gmail.com]

Sahil Babani (CS171066)
[sahilbabani1997@gmail.com]

Abhiti Darbar (CS171005)
[darbarabhiti@gmail.com]

December 13, 2020

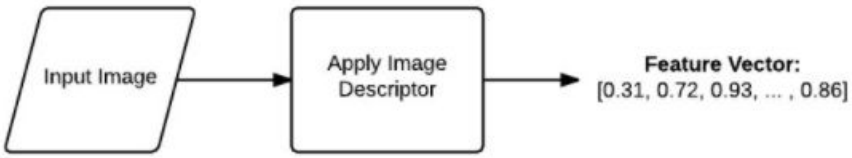
Contents

1. **Introduction**
    - 1.1. What are Descriptors
    - 1.2. Local Self-Similarities
2. **Mathematical Formulation**
    - 2.1 Sum of square differences
    - 2.2 Formula
3. **Algorithm**
4. **Documentation of API**
5. **Examples**
6. **Learning Outcome**
7. **References**

Introduction

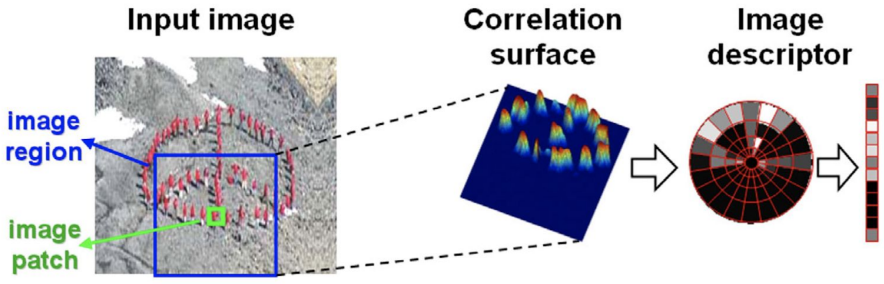
1.1 What are Descriptors

Image descriptors are descriptions of the visual features of the contents in images, videos, or algorithms or applications that produce such descriptions. They describe elementary characteristics such as the shape, the color, the texture or the motion, among others.



1.2 Local Self-Similarities

It is an approach for measuring similarity between images based on matching internal self-similarities. These in- ternal self-similarities are efficiently captured by a com- pact local "self-similarity descriptor". Local self-similarity descriptors which can now be matched across images. We associate a "local self-similarity" descriptor with every pixel. *This is done by correlating the image patch with a larger surrounding image region (e.g., of radius 40 pixels)*, resulting in a local internal "correlation surface" (the statistical independence of two points on a **surface**). *The correlation surface is then transformed into log-polar coordinates, and partitioned into 80 bins* (20 angles, 4 radial



intervals).

Img source: [1]

Mathematical Formulation

2.1 Sum of Square Differences

Sum of squared differences (SSD) is one of the measures of matching which is based on pixel by pixel intensity differences between the two images. It calculates the summation of squared for the product of pixel subtraction between two images. With this similarity measure, matching point can be determined by considering the location of minimum value in the image matrices.

The formula to calculate the SSD for images :

$$SSDq(x,y) = sum(sum(image1-image2)^2)$$

2.2 Formula

The resulting *distance surface* SSDq(x,y) is normalized and transformed into a "correlation surface" Sq (x, y).

The formula for calculating Sq (x, y) :

$$S_q(x,y) = \exp\left(-\frac{SSD_q(x,y)}{\max\left(var_{noise}, var_{auto}(q)\right)}\right)$$

where var_{noise} is a constant that corresponds to acceptable photometric variations. $var_{auto}(q)$ is the maximal variance of the difference of all patches within a very small neighborhood of q (of radius 1) relative to the patch centered at q.

The correlation surface Sq (x, y) is then transformed into log-polar coordinates centered at q, and partitioned into 80 bins (20 angles, 4 radial intervals). We select the maximal correlation value in each bin. The maximal values in those bins form the 80 entries of our local "self-similarity descriptor" vector dq associated with the pixel q. Finally, this descriptor vector is normalized by linearly stretching its values to the range [0..1], in

order to be invariant to the differences in pattern and color distribution of different patches and their surrounding image regions.

Algorithm

Local Self similarity Histogram Algorithm

Input: `Image`

Output: `2D-Array having r rows and 80 bins per row`

1. Generate all center points having coordinate:(xp,yp) of radius 40 pixel
2. Calculate single self-similarity vector for each center point:

   2.1 Initialize circle_discriptor matrix of 4*20 containing values 0 and correlation_img with (2*radius,2*radius) matrix with value 0 (radius=40p)

   2.2 Find all the internal points inside inside 80*80 image with center (xp,yp) and radius 40p

   2.3 Calculate overall difference in pixels between center patch and its offset-patch using sum of square difference i.e is simply pixel wise subtracting 2 patches from each other

      2.3.1 Find the similarity using formula:

         similarity=exp(-(difference/var_noise))

      2.3.2 Append this value to its corresponding theta section and rho section in circle_discriptor:

         theta_section=theta*20/(360)

         rho_section=rho*4/(cor_radius)

      2.3.3 Find mean in all the circle_discriptor of 4*20 matrix and return its 80 feature

3. Obtain all the feature with rows*80 (i.e bins in center with radius 40p) and return this 2d matrix feature

Documentation of API

Local Self Similarity calculates and returns the descriptors for all patch positions in the image.

Syntax:

local_self_similarity(image)
or
local_self_similarity(image, cor_radius=40, patch_size=5, step=10)

Parameters:

`image` : Image for which descriptors to be calculated. (use opencv or any other library to read image)

Optional Parameters:

`patch_size` : Size of the image patch (default and preferred: 5x5).
`cor_radius` : Radius for image region centered at patch. (default and preferred: 40)
`step` : Incremental steps (default: 10)
Note:

i) Patch size needs to be odd.

ii) Don't change default values unless you know what you are doing.

Return type:

2D Array

Example

```
import cv2 as cv
from package.local_self_similarity import local_self_similarity as lss
img = cv.imread('image.jpg')
img = cv.resize(img, (150, 100))
features = lss.local_self_similarity(img)
```

Learning Outcome

- Extracting the local “self-similarity” descriptor.
- Understanding the concept of descriptor.
- Comparing similarity between two images.

References

[1]:
https://www.researchgate.net/figure/Extracting-the-local-self-similarity-descriptor-a-at-an-image-pixel-b-at-a-video_fig2_221362526

[2]:
https://www.researchgate.net/publication/301443589_Template_Matching_using_Sum_of_Squared_Difference_and_Normalized_Cross_Correlation