Δ05 Computer Vision 2018-2019
Assignment 4 – Keypoint Desccriptors

Student: Christodoulos Asiminidis

Professor: Christophoros Nikou

Assignment sent to: agiotis@cse.uoi.gr

Department of Computer Science and Engineering

University of Ioannina

Ioannina

January 2019

The purpose of this assignment is to implement a function called descriptor() which takes at its takes at its input an image, the locations of the keypoints in the respective scale and their orientation and outputs the descriptor and the locations of the keypoints that correspond to the descriptor. Our goal in this assignment is to implement the descriptor for each keypoint.

The implementation has been conducted in Python IDLE 3.7.0. The system used has 4GB RAM and an AMD E1 processor. The operation system that runs is windows 10.

To begin with, the libraries I have used in the Python script are the following ones: numpy, PIL, sklearn, scipy and math.

The python script runs very well if the file is in the same folder where the images are. I set a variable imagpath which is the path of the lena.txt file in the beginning of the file.

Hence, I created three functions which the first loads the file as .txt format, the second one loads the initial keypoints and the last one loads the orientation.

Then, I implemented a function which returns an image as output and saves the image in the current directory. The image that is being returned is 12 times the centered keypoint.

The next function called patches takes as a parameter the keypoints that have been read, calculates the patches of the keypoints, decomposes them into 4 times 4 tiles. The function that has been used is built-in in sklearn.feature_extraction.image. After that, I save the returned array into a variable called patches. Then, the Gaussian of hsize, the size of the sub-image and scale 3 times larger than the original has been calculated in the gaussiansubimageandscale(sub_image) function.

The histogram of its pixels' gradient orientations with 8 bins has been calculated, either using as weights the magnitude times the Gaussian of step 4 using the computehistogramofitspixelsorientationweigthedwithgaussian or been normalized adding the density extra parameter as True at the numpy.histogram function built in numpy library. The functions return tuples which are a data structure in Python.

The next function that I have implemented is the euclidenadistancebetweenimages(image1,image2) which takes as parameters two images and returns the Euclidean distance between them.

The computetheminimumvalue function takes as parameters the euclideanmatrix and a threshold. For every row in the Euclidean matrix the minimum value has been calculated. The function returns the minimum value of each row as an array.