# SIFT Features:

we use the key points results from the Harris Feature detection part. Then apply the following steps like in section record:

* Get the widow images centered around each key point.
* Detect the patch main orientation using the histogram peak around the feature point.
* Rotate the image patch, and extract is as the standard deviation.
* Computing the gradient at each pixel in a 16x16 window around the detected feature point.
* Multiply with gaussian Kernel which size is 16x16 with 1.5 sigma.
* in each 4x4 quadrant, create gradient weighted orientation histogram of 8 bins(8 angles, pi/4 step)
* concatenate the histograms from the 4x4 guardant to form the 128 element descriptor
* Normalize the descriptor to 1.

Then use these descriptors in Feature matching.

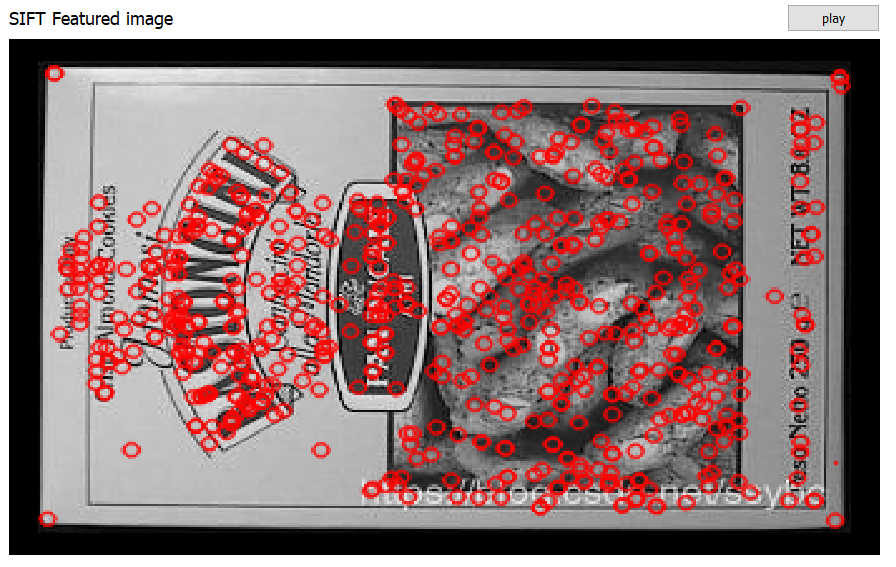
Note: I make to two Implementations to the SIFT.

First, with the concept of pyramids image that I found in the reference and paper of SIFT 'I Know the part of this implementation is Harris but with some layers of images to extract SIFT features and descriptors Like OpenCV ' in "pysift.py" file and do that in multiple image and apply 'Knnmatching function' to Know if this good features and useful for matching or not.

Second, with the concept of using key points of Harris features, then do multiple steps as described previously the get descriptor similar to the result of pyramiding implementation.

Note: There are someone help me in solving some problems in second implementation, who is 'Adel Mustafa'. the problems in the the multiple of gaussian kernels because the problem of the shape. Then this problem is solved.

The result of the SIFT is descriptor and I draw the new key points in the original image like that:



The computation time for the two implementations:

* Pyramidal implementation:

computation time: 64 seconds

* Section implementation:

computation time: 20 seconds