Easy steps for Haar cascade training on windows platform

This is a compilation of what I learned and read about while I was trying to train haar classifiers on my windows system. One major problem was that there were almost no sources that explain the exact procedure of training haar classifiers on windows. Most of the articles and blogs dealt with linux. From reference sources added below and few tips I followed, here is an article about easy way to train a haar cascade classifier on a windows system

Pre requisites:  
I hope everyone has the open cv zip file, extracted in some folder of choice. The library contains several tools that make our job of training classifiers much easier than what it takes the hard way. Apart from that, brief experience with haar cascade programs and a little knowledge of how haar cascade training works. We will be discussing about the same in the next section.   
Hardware requirement would be a powerful spare PC which you are ready to leave for a few days( yes! DAYS!) for the training to finish. For reference, minimum power would be 4gb ram,2.8khz,i3 pc, which is what I have used for this demonstration.  
You can download a zip file of all the requirements for the tutorial from this

Introduction :   
  
Haar cascade procedure is developed from the concept proposed by voila johns paper on rapid object detection using haar like features. In this procedure, we train a classifier using a few hundread sample views of a particular object ( in this case, my palm), called positive samples. For efficiency and to increase the speed of training, we scale down these images to small dimensions. Although I go with 50x70 as my size, it is advisable to go with smaller size ( approx 20x20). This is one factor that decides the training time your machine would take. And we also need a few hundred negative images

To test and verify computation procedure, you could start with around 40-50 positive images and a few hundred negative images

The cascading(multiple stages of classifying an ROI) classifier is trained such that it detects a specified region that bounds the object of interest.The classifier is designed such that it can be easily resized to detect features similar to those marked in the samples provided for training. So, to find the object of interest, the image has to be scanned for different scales of classifier.

For detailed description about the whole procedure, visit here:  
[link1](http://note.sonots.com/SciSoftware/haartraining.html) (Recomended)

[Link2](coding-robin.de/2013/07/22/train-your-own-opencv-haar-classifier.html) (Compiled and brief version with a fun example)

**Step1: Collecting Positive samples.**  
  
I had to take about 100 snaps of my open palm at slightly changed angles and different background. So I used a python scripts that does it for me :  
  
*import cv2*

*import time*

*import numpy as np*

*cap = cv2.cv.CaptureFromCAM(0)*

*img=cv2.cv.QueryFrame(cap)*

*i=0*

*while i in range(0,100):*

*cv2.imshow('go',img)*

*img=cv2.cv.QueryFrame(cap)*

*cv2.cv.SaveImage(str(i)+".jpg",img)*

*print "F:\Python\_programs\python ocv\haartraining\opencv-haar-classifier-training-master\positive\_images\\"+str(i)+".jpg"*

*time.sleep(.10)*

*i+=1*

*print " "*

*text\_file.close()*

The program captured about hundred images for me,numbered them, and saved them as jpegs in the positive\_images folder.

**Step 2: cropping and scaling images**

Positive images: This may be an arduous task! But you will have to crop each image carefully to have a smaller image containing your object!   
Make sure you keep the dimensions of the cropped images constant, or maintain a constant ratio for every image, because every image will be scaled down to much smaller dimensions of size 20x 20 (approx.)

Negative Images: We need to simply scale these images down to same size as positive images. And store them in a folder called negative\_images