



Main Page

Related Pages

Namespaces ▾

Classes ▾

Files ▾

Examples

Java documentation

Q ▾ Search

OpenCV Tutorials

Other tutorials (ml, objdetect, photo, stitching, video)

Cascade Classifier Training

[Prev Tutorial: Cascade Classifier](#)

[Next Tutorial: Barcode Recognition](#)

Introduction

Working with a boosted cascade of weak classifiers includes two major stages: the training and the detection stage. The detection stage using either HAAR or LBP based models, is described in the [object detection tutorial](#). This documentation gives an overview of the functionality needed to train your own boosted cascade of weak classifiers. The current guide will walk through all the different stages: collecting training data, preparation of the training data and executing the actual model training.

To support this tutorial, several official OpenCV applications will be used: `opencv_createsamples`, `opencv_annotation`, `opencv_traincascade` and `opencv_visualisation`.

Note

Createsamples and traincascade are disabled since OpenCV 4.0. Consider using these apps for training from 3.4 branch for Cascade Classifier. Model format is the same between 3.4 and 4.x.

Important notes

- If you come across any tutorial mentioning the old `opencv_haartraining` tool (*which is deprecated and still using the OpenCV1.x interface*), then please ignore that tutorial and stick to the `opencv_traincascade` tool. This tool is a newer version, written in C++ in accordance to the OpenCV 2.x and OpenCV 3.x API. The `opencv_traincascade` supports both HAAR like wavelet features [290] and LBP (Local Binary Patterns) [169] features. LBP features yield integer precision in contrast to HAAR features, yielding floating point precision, so both training and detection with LBP are several times faster than with HAAR features. Regarding the LBP and HAAR detection quality, it mainly depends on the training data used and the training parameters selected. It's possible to train a LBP-based classifier that will provide almost the same quality as HAAR-based one, within a percentage of the training time.
- The newer cascade classifier detection interface from OpenCV 2.x and OpenCV 3.x (`cv::CascadeClassifier`) supports working with both old and new model formats. `opencv_traincascade` can even save (export) a trained cascade in the older format if for some reason you are stuck using the old interface. At least training the model could then be done in the most stable interface.
- The `opencv_traincascade` application can use TBB for multi-threading. To use it in multicore mode OpenCV must be built with TBB support enabled.

Table of Contents

- ↓ Introduction
- ↓ Important notes
- ↓ Preparation of the training data
- ↓ Negative Samples
- ↓ Positive Samples
- ↓ Extra remarks
- ↓ Using OpenCV's integrated annotation tool
- ↓ Cascade Training
- ↓ Visualising Cascade Classifiers