COMS30121: Assignment 02 (‘Object Recognition’)

Guidelines

- *This task is the first assessed and summative piece of coursework in the unit.*
- *This assignment is to be completed in groups of 2 or 3 (please report any change to the current team structure to the course director BEFORE starting your assignment)*
- *This assignment is worth 25% of the unit mark*
- *Every student is required to upload their full piece of work (that is a 1page PDF file and all code and executables in a project folder including all your source code files and project files as a single ZIP file to SAFE before 23:59:59, Wed 20th November 2012. Make sure you upload it early enough (not last minute!) to avoid upload problems. (Each member of a team has to upload an identical copy of the team’s work.)*
- *In addition to the online submission, you will present your submitted program running in the labs on 21st November 2012. You will need to attend this lab session to get a mark. At these labs, we will ask you questions about your work and you will be able to showcase the merits of it.*
- *Your work will be marked on an individual basis considering both the submission and the lab presentation.*
- *Do not attempt to plagiarise or copy code between subgroups etc. It is not worth it, be proud of your own work! We will individually ask you questions about your work in the labs - so you must understand the code your team developed in any case.*

Introduction

This assignment consists of TWO subtasks. The first subtask is on experimenting with a face detector, and the second subtask is on boosting and training of an object detector. You will be provided with working code “face.cpp” that reads an XML file containing (a strong, boosted classifier) together with an image (path provided as parameter). It then performs object detection on this image and outputs the result by drawing bounding boxes around the objects found.

Task One: Experimenting with a Face Detector

As discussed in recent lecture, being able to detect and locate instances of an object class in images is important for many computer vision applications and is an ongoing area of research. This first task requires you to experiment with the powerful Viola-Jones object face detection framework provided by OpenCV. It uses boosted Haar-like features for the detection of near-frontal faces as discussed in the lectures.

Task One Breakdown

1. Build the provided source code in order to generate an executable

2. You are given 12 images of the US president (depicted below and available on the webpage). Apply your face detector to these images by running the executable on these face images, and view the results.

3. Per group, create a brief, maximum 1/2 page summary in form of a PDF file, which provides per image your interpretation of reasons for detection success or failure, where failure could be a missed face (false negative) or the detection of something that is not a face (false positive). This interpretation should show your understanding of the underlying features and boosting algorithm used for detection.

4. Per group, take six images of group members yourself (with your phone or webcam or photo camera, any device will do, please let us know if you have no camera available in your group). Make sure that all face detections are correct in half of these images and incorrect in the others. Per group, create another brief, maximum 1/2 page summary in form of a PDF file, which provides per image your interpretation of reasons for detection success or failure, where failure could be a missed face (false negative) or the detection of something that is not a face (false negative). This interpretation should again show your understanding of the underlying features and boosting algorithm used for detection.

Task Two: Constructing Object Detectors

This second task requires you to build an object detector that recognises dart boards. The initial steps of this subtask introduce you to OpenCV's boosting tool, which you can use to construct an object detector that uses Haar-like features (here for recognizing a dart board, but potentially for any rigid object you want to detect). In the second part you can then expand on the resulting object detector to improve performance. In order to give you a head start you are provided with readily compiled versions of `opencv_createsamples.exe` and `opencv_haartraining.exe`, plus a large set of negative images (containing no dart boards).

Task Breakdown

1. Create your positive training data set of 500 dart board samples from a single image of a dart board (`dart.bmp`) shown below. To do this, you can use opencv_createsamples.exe via the following command within the folder `training` provided:
   `opencv_createsamples.exe -img dart.bmp -vec dart.vec -neg negatives.dat -show -w 20 -h 20 -num 600 -maxidev 80 -maxxangle 0.8 -maxyangle 0.8 -maxzangle 0.2`

   

   This will show the created tiny 20×20 images of dart boards (press SPACE to see subsequent images) and create the file `dart.vec` which contains all 600 small sample images of this dart board. Each of these has been created by randomly changing in viewing angle and contrast (up to the maximum values specified) to reflect the possible variability in real images better.

2. Use the created sample set to train a dart board detector via boosting. To do this, you can use opencv_haartraining.exe via the following command within the folder `training` provided:
   `opencv_haartraining.exe -data dartcascade -vec dart.vec -bg negatives.dat -nstages 3 -nsplits 2 -minhitrate 0.999 -maxfalsealarm 0.04 -npos 600 -nneg 600 -w 20 -h 20 -sym -mem 2048 -mode ALL`

   This will start the boosting procedure and construct a strong classifier stored in the file `dartcascade.xml`. In short: during boosting the program will consider the 600 positive images and employ 600 negative images to sample from. The detector window will be 20×20 pixels and features are learned in a symmetrical fashion, that is object symmetry about the vertical axis is assumed. (Just for a complete view: to further speed up the detection process, the strong classifier is built in 3 parts (nstages) and combines them in binary trees (nsplits) instead of a linear fashion.). Boosting stops if the constructed detector can identify 99.9 percent (i.e. all) of the 600 positive samples provided and only fails on 0.0064 percent ($= 0.04^3$) of negative samples. The training procedure may take approx. 15min.

3. Test your detector on the images below (`dart0.jpg` to `dart11.jpg`) by using the skeleton code from subtask one and loading your `dartcascade.xml` instead of the face classifier.

4. Combine your detector with other methods of your choice (e.g. differently built Haar-like detectors, edge descriptors, histograms, shape descriptors, segmentation algorithms, ...) to improve on the detection performance (both to detect more dart boards or reduce the detection of non dart boards, or to detect dart boards more accurately). You are allowed to use any functionality/command OpenCV offers.


GAME OF DARTS


unicorn
unicorn


BODDINGTONS
MAXIMUM ROOM CAPACITY
Carlsberg


MODUS darts.tv


THE DIAMOND