Running performance tests and analyzing the results

To simplify the task of performance measurement OpenCV includes several scripts to run the test cases and analyze results.

The following conditions are applicable to the all scripts:

- Scripts are written in python and require python 2.6 or newer (but not the 3.x).
- They are located at /modules/ts/misc/ folder in the OpenCV source tree.
- All reporting scripts are able to output results in plain text and html formats.
- Each script accept -h option to print help on supported parameters.

The most useful scripts are:

- *run.py* - tests runner - able to run tests on Windows, Linux, Android and Mac (is not tested yet).
- *report.py* - prints all measurements from single test run in user-friendly way.
- *summary.py* - creates comparison table for several test runs (able to compare different platforms or revisions).
- *chart.py* - shows performance dependencies from test parameters (works on single log).

Running the tests with run.py

Script run.py works as test launcher and is able to run tests on different platforms. It automatically detects target OS and CPU architecture, SVN revision, GPU support and several other things; it automatically generates descriptive file names for log files and is able to find all OpenCV performance tests. To run the tests on Android platform you should have the only device connected to your host with adb tool. Script is able to run tests from several build directories at one command.

Please note the following:

run.py never fires build command. You should build all tests by yourself before running them. run.py is able to run tests from any project which uses cmake tool. But first of all it is designed for OpenCV. If you use IDE supporting multiple configurations (i.e. Visual Studio or XCode) then run.py is unable to determine your current IDE configuration and tries to run the tests from defaults cmake configuration (it is "Release" usually). So you need to specify configuration manually if you want to run tests from non-default configuration. run.py does not set OPENCV_TEST_DATA_PATH environment variable on the desktop OS'es (but sets it for Android) so you should set it yourself before the test run. run.py accepts the following options:

- `-h` print help and exit
- `-t <comma-separated list of tests>` list of tests to run. By default script tries to run all the tests starting with opencv_perf_ prefix. You can use OpenCV module name, executable name or full path as a test name;
- `-w <working directory>` sets the working directory for the performance tests. All the tests will be executed in the directory and all the log files will be placed there too. In case of Android this directory will be used for logs only.
- `--android_test_data_path <path on device>` this parameter allows to change location of OpenCV test data on the Android device. By default run.py assumes that the test data is placed in '`/mnt/sdcard/opencv_testdata`;
- `--configuration <Release|Debug>` forces run.py to use specific configuration in a case when IDE supports multiple configurations;
- `--perf_XXXX=<value>` and `--gtest_XXXX=<value>` these options are passed to each test without the modifications (exception is `--gtest_output` parameter).

Usage examples

Run all performance tests:

```
python <opencv_dir>/modules/ts/misc/run.py <build_dir>
```

Run all performance tests for desktop, then for Android:

```
python <opencv_dir>/modules/ts/misc/run.py <desktop_build_dir> <android_build_dir>
```

Run only tests for core module:

```
python <opencv_dir>/modules/ts/misc/run.py <build_dir> -t core
```

Run only tests for core and imgproc modules on single channel data:

```
python <opencv_dir>/modules/ts/misc/run.py <build_dir> -t core,imgproc --gtest_filter=*C1*
```

Run test in logs working directory:

```
mkdir -p logs
python <opencv_dir>/modules/ts/misc/run.py <build_dir> -w ./logs
```

Run test on Android and store tests in the logs working directory:

```
mkdir -p logs
python <opencv_dir>/modules/ts/misc/run.py <build_dir> -w ./logs
```

Run tests on Android using tests directory as OPENCV_TEST_DATA_PATH:

```
python <opencv_dir>/modules/ts/misc/run.py <build_dir> --android_test_data_path .
```

How to update perf data

Imagine that you've modified algorithm, and now it gives different results. If there is obvious regression (results are worse), you have to fix it. But what if the results are better or just very slightly different (as you discovered by visual inspection of the results)? Since OpenCV performance tests include some regression check, your code will fail the tests, and so you need to regenerate the test data. You can do it in 3 simple steps:

1. Remove old sanity data

```
cd opencv_extra/testdata/perf
python ./clean_regex.py <module>.xml "<regex_matching_your_tests>"
# see that regex works as expected
python ./clean_regex.py <module>.xml "<regex_matching_your_tests>" do
```

2. Generate new data

```
# export OPENCV_TEST_DATA_PATH=<opencv_extra_dir>/testdata
python <opencv_dir>/modules/ts/misc/run.py <build_dir> -t <module> --check --perf_write_sani
```

3. Verify new data

```
cd build
python <opencv_dir>/modules/ts/misc/run.py -t <module> --check
```

Viewing results with report.py

Script report.py is used to display all the results from single test run in table form. It can read multiple logs at time but it will simple concatenate results from all passed logs. By default report.py skips test that were not run. If some test repeats several times in passed logs, then resulting table will hve several lines with the same name for this case; order of this lines is undefined.

report.py accepts the following options:

- `-h` print options help and exit;
- `-o <txt|html|auto>` sets the output format; default format is auto - script tries to determine desired format automatically;
- `-u <s|ms|mks|ns|ticks>` units for output values; default is ms;
- `-f <regex filter>` filter for test names; if this parameter is passed then only tests having non-empty match will be included;
- `-c <comma-separated list of columns>` list of columns to be displayed; by default all columns are printed. Supported column names are:
  - name - name of test; if this column is not specified then script automatically prin name as first column;
  - samples - number of collected samples;
  - outliers - number of outliers excluded from final results calculation;
  - min - minimal observed time;
  - median - median over all collected time measurements;

  - gmean - geometric mean;
  - mean - mean;
  - stddev - standard deviation.
- `--show-all` also include test which were not run.

Usage examples

Process all xml files from the current directory:

```
cd <build_dir>
python <opencv_dir>/modules/ts/misc/report.py *.xml
```

Show medians for single log file:

```
python <opencv_dir>/modules/ts/misc/report.py objdetect_posix_x64_6706M_2011-09-12--12-54-17
```

And here is an output:

```
objdetect_posix_x64_6706M_2011-09-12--12-54-17.xml

Name of Test                                               Median
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 24) 87.19 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 30) 60.63 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 40) 46.59 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 50) 40.45 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 60) 26.20 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 70) 16.91 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 80) 13.36 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 90) 10.61 ms
```

Show medians for single log file and output results into html:

```
python <opencv_dir>/modules/ts/misc/report.py objdetect_posix_x64_6706M_2011-09-12--12-54-17
```

And here is an output:

```
objdetect_posix_x64_6706M_2011-09-12--12-54-17.xml
Name of Test     Median
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 24)   87.19 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 30)   60.63 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 40)   46.59 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 50)   40.45 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 60)   26.20 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 70)   16.91 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 80)   13.36 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 90)   10.61 ms
```

Display only subset of tests:

```
python <opencv_dir>/modules/ts/misc/report.py objdetect_posix_x64_6706M_2011-09-12--12-54-17
```

output:

```
objdetect_posix_x64_6706M_2011-09-12--12-54-17.xml
Name of Test     Median
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 40)   46.59 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 50)   40.45 ms
CascadeClassifierLBPFrontalFace::ImageName_MinSize::("cv/shared/lena.jpg", 60)   26.20 ms
```

Comparing results with summary.py

Script summary.py is used to compare results of the same tests from different revision or executed on different hardware. It reads multiple log files and creates a comparison table.

summary.py accepts the following options:

- `-h` print options help and exit;
- `-o <txt|html|auto>` sets the output format; default format is auto - script tries to determine desired format automatically;
- `-u <s|ms|mks|ns|ticks>` units for output values; default is ms;
- `-f <regex filter>` filter for test names; if this parameter is passed then only tests having non-empty match will be included;
- `-m <output metric>` the target metric to be compared. By default script outputs geomenric means. This parameter can have one of the following values:
  - min - minimal observed time;
  - median - median over all collected time measurements;
  - gmean - geometric mean;
  - mean - mean;
  - stddev - standard deviation.
- `--show-all` also include test which were not run;
- `--no-relatives` do not print columns with relative values.

Usage examples

Show comparison for “add” tests from all logs of core module:

```
python <opencv_dir>/modules/ts/misc/summary.py core*.xml -f add:
```

output:

```
Geometric mean
Name of Test                                            core                core
posix              posix                          posix
x64                x64                            x64
6693M              6695                           6695
2011-09-08--13-13-41  2011-09-08--13-30-06     2011-09-08--13-30-06
```

```
vs
core
posix
x64
6693M
2011-09-08--13-13-41
core_arithm__add::Size_MatType::(127x61, 32FC1)    0.008 ms           0.008 ms
core_arithm__add::Size_MatType::(127x61, 32SC1)    0.009 ms           0.009 ms
core_arithm__add::Size_MatType::(127x61, 8SC1)     0.024 ms           0.024 ms
core_arithm__add::Size_MatType::(127x61, 8UC1)     0.008 ms           0.008 ms
core_arithm__add::Size_MatType::(127x61, 8UC4)     0.031 ms           0.031 ms
core_arithm__add::Size_MatType::(1280x720, 32FC1)  1.495 ms           1.213 ms
core_arithm__add::Size_MatType::(1280x720, 32SC1)  1.492 ms           1.332 ms
core_arithm__add::Size_MatType::(1280x720, 8SC1)   3.056 ms           3.112 ms
core_arithm__add::Size_MatType::(1280x720, 8UC1)   0.937 ms           0.929 ms
core_arithm__add::Size_MatType::(1280x720, 8UC4)   3.909 ms           3.855 ms
core_arithm__add::Size_MatType::(1920x1080, 32FC1) 3.016 ms           3.055 ms
core_arithm__add::Size_MatType::(1920x1080, 32SC1) 3.125 ms           3.199 ms
core_arithm__add::Size_MatType::(1920x1080, 8SC1)  6.952 ms           6.841 ms
core_arithm__add::Size_MatType::(1920x1080, 8UC1)  2.189 ms           2.133 ms
core_arithm__add::Size_MatType::(1920x1080, 8UC4)  8.632 ms           8.831 ms
core_arithm__add::Size_MatType::(640x480, 32FC1)   0.329 ms           0.331 ms
core_arithm__add::Size_MatType::(640x480, 32SC1)   0.395 ms           0.391 ms
core_arithm__add::Size_MatType::(640x480, 8SC1)    1.020 ms           1.001 ms
core_arithm__add::Size_MatType::(640x480, 8UC1)    0.317 ms           0.310 ms
core_arithm__add::Size_MatType::(640x480, 8UC4)    1.269 ms           1.270 ms
```

Compare results only for “add” function operating on 4 channel matrixes and show results in seconds:

```
python <opencv_dir>/modules/ts/misc/summary.py core*.xml -f "add:.*C4" -u s
```

output:

```
Geometric mean
Name of Test                                         core                      core
posix                     posix                   posix
x64                       x64                     x64
6693M                     6695                    6695
2011-09-08--13-13-41       2011-09-08--13-30-06    2011-09-08--13-30-06
vs
core
posix
x64
6693M
2011-09-08--13-13-41
core_arithm__add::Size_MatType::(127x61, 8UC4)     0.000 s                   0.000 s
core_arithm__add::Size_MatType::(1280x720, 8UC4)   0.004 s                   0.004 s
```

```
core_arithm__add::Size_MatType::(1920x1080, 8UC4)  0.009 s                   0.009 s
core_arithm__add::Size_MatType::(640x480, 8UC4)    0.001 s                   0.001 s
```

Compare minimal values instead of geometric mean only for "add" function operating on 4 channel matrixes:

```
python <opencv_dir>/modules/ts/misc/summary.py core* -f "add:.*C4" -m min
```

output:

```
Min
Name of Test                                       core                    core
posix                   posix                    posix
x64                     x64                      x64
6693M                   6695                     6695
2011-09-08--13-13-41    2011-09-08--13-30-06     2011-09-08--13-30-06
vs
core
posix
x64
6693M
2011-09-08--13-13-41
core_arithm__add::Size_MatType::(127x61, 8UC4)     0.031 ms                0.031 ms
core_arithm__add::Size_MatType::(1280x720, 8UC4)   3.772 ms                3.770 ms
core_arithm__add::Size_MatType::(1920x1080, 8UC4)  8.544 ms                8.545 ms
core_arithm__add::Size_MatType::(640x480, 8UC4)    1.227 ms                1.227 ms
```

Display only relative values of geometric mean for "add" function:

```
python <opencv_dir>/modules/ts/misc/summary.py core* -f add: -m gmean%
```

output:

```
Geometric mean (relative)
Name of Test                                       core                    core
posix                  posix
x64                    x64
6693M                  6695
2011-09-08--13-13-41   2011-09-08--13-30-06
core_arithm__add::Size_MatType::(127x61, 32FC1)     1.00                   1.00
core_arithm__add::Size_MatType::(127x61, 32SC1)     1.00                   1.00
core_arithm__add::Size_MatType::(127x61, 8SC1)      1.00                   1.00
core_arithm__add::Size_MatType::(127x61, 8UC1)      1.00                   1.00
core_arithm__add::Size_MatType::(127x61, 8UC4)      1.00                   1.00
core_arithm__add::Size_MatType::(1280x720, 32FC1)   1.00                   0.81
core_arithm__add::Size_MatType::(1280x720, 32SC1)   1.00                   0.89
core_arithm__add::Size_MatType::(1280x720, 8SC1)    1.00                   1.02
core_arithm__add::Size_MatType::(1280x720, 8UC1)    1.00                   0.99
core_arithm__add::Size_MatType::(1280x720, 8UC4)    1.00                   0.99
core_arithm__add::Size_MatType::(1920x1080, 32FC1)  1.00                   1.01
```

```
core_arithm__add::Size_MatType::(1920x1080, 32SC1)   1.00                  1.02
core_arithm__add::Size_MatType::(1920x1080, 8SC1)    1.00                  0.98
core_arithm__add::Size_MatType::(1920x1080, 8UC1)    1.00                  0.97
core_arithm__add::Size_MatType::(1920x1080, 8UC4)    1.00                  1.02
core_arithm__add::Size_MatType::(640x480, 32FC1)     1.00                  1.00
core_arithm__add::Size_MatType::(640x480, 32SC1)     1.00                  0.99
core_arithm__add::Size_MatType::(640x480, 8SC1)      1.00                  0.98
core_arithm__add::Size_MatType::(640x480, 8UC1)      1.00                  0.98
core_arithm__add::Size_MatType::(640x480, 8UC4)      1.00                  1.00
```

Viewing dependency from parameters with chart.py

Script chart.py is used to visualise dependency from test parameters within a single test suite. This scripts requires a single log file and unique test suite filter. Script executed without filter expression will only print names of test suites available in parsed file.

chart.py accepts the following options:

- `-h` print options help and exit;
- `-o <txt|html|auto>` sets the output format; default format is auto - script tries to determine desired format automatically;
- `-u <s|ms|mks|ns|ticks>` units for output values; default is ms;
- `-f <regex filter>` filter for test names;
- ‘`-m <output metric>` the target metric to be compared. By default script outputs geomenric means. This parameter can have one of the following values:
  - min - minimal observed time;
  - median - median over all collected time measurements;
  - gmean - geometric mean;
  - mean - mean;
  - stddev - standard deviation.
- `x <row index>` number of test argument to be listened in rows;
- `-y <col index>` number of test argument to be listened in columns.

Usage examples

List all test suites from the log file

```
python <opencv_dir>/modules/ts/misc/chart.py imgproc_posix_x64_6695_2011-09-08--13-34-18.xml
```

output:

```
Error - unable to create tables for functions from different test suits:
1:   cvtColorGray2::Size_CvtMode
2:   cvtColorGray::Size_CvtMode
3:   cvtColorH::Size_CvtMode
4:   cvtColorYUV420::Size_CvtMode_OutChNum
5:   cvtColorYUV::Size_CvtMode_OutChNum
```

```
6:   integral1::Size_MatType_OutMatDepth
7:   integral2::Size_MatType_OutMatDepth
8:   integral3::Size_MatType_OutMatDepth
9:   resizeDownLinear::MatInfo_Size_Size
10:   resizeUpLinear::MatInfo_Size_Size
```

View tables for the cvtColorYUV420::Size_CvtMode_OutChNum suite:

```
python <opencv_dir>/modules/ts/misc/chart.py imgproc_posix_x64_6695_2011-09-08--13-34-18.xml
```

output:

```
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, X, 3)
X\Y  130x60  640x480     1280x720    1920x1080
CV_YUV420i2BGR   0.10 ms     4.27 ms     12.98 ms    29.14 ms
CV_YUV420i2RGB   0.11 ms     4.36 ms     13.09 ms    29.39 ms
CV_YUV420sp2BGR  0.12 ms     4.37 ms     13.05 ms    29.43 ms
CV_YUV420sp2RGB  0.12 ms     4.35 ms     13.10 ms    29.54 ms
View tables for the `cvtColorYUV420::Size_CvtMode_OutChNum` suite:
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, X, 4)
X\Y  130x60  640x480     1280x720    1920x1080
CV_YUV420i2BGR   0.10 ms     4.10 ms     12.21 ms    27.50 ms
CV_YUV420i2RGB   0.10 ms     4.03 ms     12.18 ms    27.41 ms
CV_YUV420sp2BGR  0.11 ms     4.13 ms     12.28 ms    27.65 ms
CV_YUV420sp2RGB  0.11 ms     3.99 ms     12.12 ms    27.30 ms
```

Compare times for image size to number of channels in the cvtColorYUV420::Size_CvtMode_OutChNum suite:

```
python <opencv_dir>/modules/ts/misc/chart.py imgproc_posix_x64_6695_2011-09-08--13-34-18.xml
```

output:

```
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, CV_YUV420i2BGR, X)
X\Y  130x60  640x480     1280x720    1920x1080
3    0.10 ms     4.27 ms     12.98 ms    29.14 ms
4    0.10 ms     4.10 ms     12.21 ms    27.50 ms
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, CV_YUV420i2RGB, X)
X\Y  130x60  640x480     1280x720    1920x1080
3    0.11 ms     4.36 ms     13.09 ms    29.39 ms
4    0.10 ms     4.03 ms     12.18 ms    27.41 ms
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, CV_YUV420sp2BGR, X)
X\Y  130x60  640x480     1280x720    1920x1080
3    0.12 ms     4.37 ms     13.05 ms    29.43 ms
4    0.11 ms     4.13 ms     12.28 ms    27.65 ms
```

```
Geometric mean for
cvtColorYUV420::Size_CvtMode_OutChNum::(Y, CV_YUV420sp2RGB, X)
X\Y  130x60  640x480     1280x720    1920x1080
3    0.12 ms     4.35 ms     13.10 ms    29.54 ms
4    0.11 ms     3.99 ms     12.12 ms    27.30 ms
```