The page contains Frequently Asked Questions and Answers for them and list of well known problems with links to related discussions.

Build & Install

Q: *OpenCV has very powerful testing framework, but it's not a part of distribution. Why?*
A: "ts" is internal module, it is not designed for using it externally - so it was removed from regular public modules list. You may consider opencv_contrib approach of developing OpenCV external modules. (`-DOPENCV_EXTRA_MODULES_PATH=path1\;path2` - note on backslash before `;`, required by "bash").
Discussion: #8408

Q: *OpenCV build fails with internal compiler error (a.k.a ICE) or segmentation fault on my ARM board. What can I do?*
A: OpenCV team does not work on such kind of bug reports and the best solution for it is to file bugs to compiler team. However there are several popular solutions to fix the problem: - Reduce amount of build jobs in make and even remove `-j` option from `make` command line. Each build job is dedicated compiler process that consumes RAM. Serial build without `-j` consumes minimal amount if RAM. - Add swap space in your OS. Relates to previous one. - Use cross compilation approach:link - Update compiler.

Discussion: TBD

General API Concepts

Q: *What is InputArray and how can I understand the actual input types of parameters?*
A: This is the proxy class for passing read-only input arrays into OpenCV functions. The class should not be used in user code, pass `cv::Mat`, `cv::UMat`, `cv::GpuMat` or `std::vector<>` as function parameter and OpenCV extracts data from it without memory copying. In case if you work on new OpenCV module in Contrib repository or new function for core library you should use `cv::_InputArray::getMat()` method to construct a matrix header for the array inside function. `cv::_InputArray::kind()` can be used to distinguish actual input container format, but usually it is not needed.

Documentation: https://docs.opencv.org/master/d4/d32/classcv_1_1_1___InputArray.html

Hardware & OS

TBD

Video I/O, Image I/O, Data serialization

Q: *Why OpenCV does not provide functions with `wchar_t`, `wstring`? What about non-Latin symbols in file names?*
A: OpenCV team decided to stay conservative and do not introduce new API calls with `wchar_t`, `wstring` and other string types By the following reasons: - Most of image decoding and encoding libraries use standard `fopen` call to open files and extra `wchar_t` support requires domain libraries modification. - Modern Linux, Mac OS and latest Windows releases support UTF-8 encoding that allows to use `std::string` as container to pass it to OpenCV functions. - Popular FS, including solutions on Linux does not use `wchar_t` natively and the overloads are not cross platform.

There are 2 alternatives to use `wchar_t` strings with OpenCV (see discussion section for code snippets and solutions):

1. Convert `wchar_t` strings to UTF-8 and pass UTF-8 string as `cv::imread`, `cv::imwrite`, etc parameter. UTF-8 string is handled by system `fopen` call and it's behavior depends on OS support and locale. See `mbstowcs` in C++ standard for more details.

2. OpenCV provides `cv::imdecode` and `cv::imencode` functions that allow to decode and encode image using memory buffer as input and output. The solution decouples file IO and image decoding and allows to manage path strings, locales, etc in user code.

Discussion: 4292, 5631, 13368

Q: *VideoCapture cannot open my file/camera or does it in wrong way. How can I resolve the issue?*
A: `cv::VideoCapture` uses different backends for different cameras, files and platforms. There are several things to check and try with OpenCV: 1. OpenCV build options and available backends with `cv::getBuildInformation()` (`cv2.getBuildInformation()` in Python). See documentation for details. 2. Run your application with `OPENCV_VIDEOIO_DEBUG=1` option to enable extra logging inside the library. It helps to identify the backend used in run time and configuration details. 3. Force OpenCV to use particular backend with the second parameter in constructor. See documentation for details.

Discussion: TBD

Classic Computer Vision

Q: *Which is more efficient, use `contourArea()` or count number of ROI non-zero pixels?*
A: `cv::contourArea()` uses Green formula to compute the area, therefore its complexity is `O(contour_number_of_vertices)`. Counting non-zero pixels in the ROI is `O(roi_width*roi_height)` algorithm, i.e. much slower. Note,

however, that because of finite, and quite low, resolution of the raster grid, the two algorithms will give noticeably different results. For large and square-like contours the error will be minimal. For small and/or oblong contours the error can be quite large.

Links: Green's Theorem

DNN

Q: *Cannot read frozen object detection `.pb` with OpenCV DNN API after it was trained with TF Object Detection API. What is the reason?*

A: TF Object Detection API provides opportunities for detection model training. There are scripts for TF2, TF1.5 training and further model export, namely `model_main_tf2.py` and `exporter_main_v2.py`, `model_main.py` and `export_inference_graph.py`. After `exporter_main_v2.py` execution the model, checkpoints and variables will be saved in the specified directory. Observing this model after freezing, you will find `StatefulPartitionedCall/...` nodes. It indicates TF Eager Mode, which is not supported in OpenCV. One of the options is using TF1.5 scripts: `model_main.py` and `export_inference_graph.py`. **Discussion:** #19257

Q: *Is there control flow support in OpenCV?*

A: OpenCV doesn't support control flow and currently, there is no plan for the near future to implement it. Error example: `onnx_graph_simplifier.cpp:592: error: (-210:Unsupported format or combination of formats) Unsupported data type: BOOL in function 'getMatFromTensor'` The problem appeared during `.onnx` feeding into `cv2.dnn.readNetFromONNX(...)`. **Discussion:** #19366, #19977

Python Bindings

Q: *I call OpenCV function as it's done in C++ or stated in documentation, but get some strange data type or buffer size exception. How to debug it?*

A: OpenCV Bindings for Python use Numpy Array as base container instead or wrapping `cv::Mat` to Python. Data conversion is done on-the-go in C++ during function call. Some of conversions could be buggy or counterintuitive. OpenCV provides function `cv.utils.dumpInputArray()` that returns details of C++ representation of Python arrays that can help.

Documentation: #16807 #19091, #17456

Q: *May I use inheritance with OpenCV classes in Python?*

A: No, OpenCV classes are C++ classes wrapped to Python. Inheritance is not supported right now and may lead to segmentation fails and other undefined

behavior.

Discussion: #15804, #21697

Q: *I tried to build the OpenCV-Python from source on Windows with Python 3.9, but after* `import cv2` *got an error:* `ImportError: DLL load failed while importing cv2: The specified module could not be found.`

A: Python 3.9 has changed the way libraries are loaded. You can downgrade your Python's version to 3.8 or setup additional environment variables as described at #20206.