# Mobile Application Development

## (Native C++ & OpenCV)

Instructor: Thanh Binh Nguyen

February 1st, 2020

S³Lab
*Smart Software System Laboratory*


“The future of mobile is the future of online. It is how people access online content now.”
– David Murphy, Founder and Editor of Mobile Marketing Daily

# OpenCV

*Introduction*

- OpenCV stands for the Open Source Computer Vision Library.
  - It was founded at Intel in 1999, went through some lean years after the .bust but is now under active development, now receiving ongoing support from Willow Garage.
- OpenCV is free for commercial and research use.
  - It has a BSD license.  The library runs across many platforms and actively supports Linux, Windows and Mac OS.
- OpenCV was founded to advance the field of computer vision.
  - It gives everyone a reliable, real time infrastructure to build on.  It collects and makes available the most useful algorithms.

# OpenCV

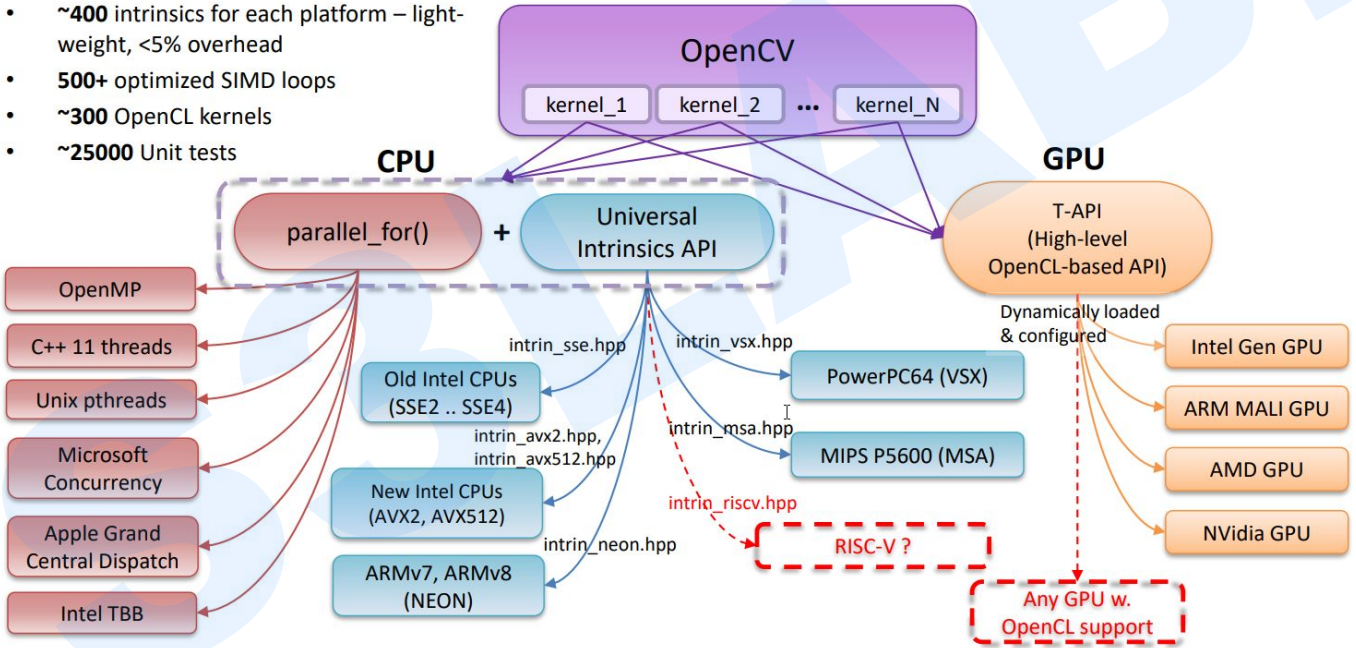
*Features*

- Cross Platform
  - Windows, Linux, Mac OS
- Portable
  - iPhone
  - Android.
- Language Support
  - C/C++
  - Python

# OpenCV

*write once, run fast everywhere*

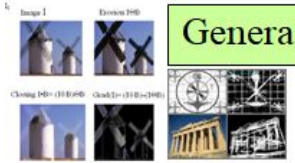
- **~400** intrinsics for each platform – light-weight, <5% overhead
- **500+** optimized SIMD loops
- **~300** OpenCL kernels
- **~25000** Unit tests

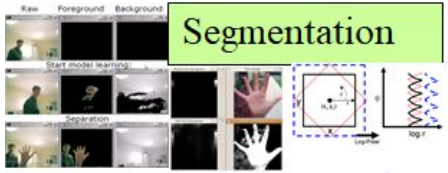


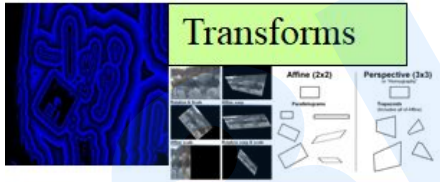
https://www.youtube.com/watch?v=aDpnaxPAmtU

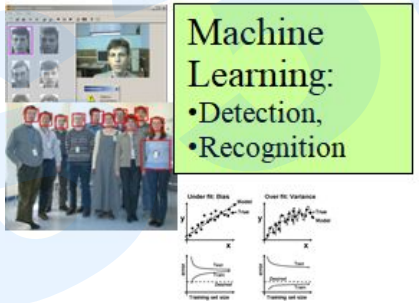
# OpenCV

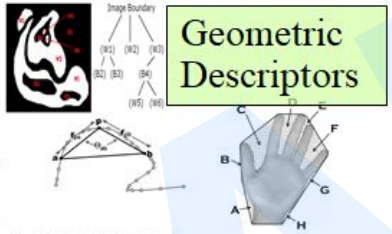
*> 500 funcs*

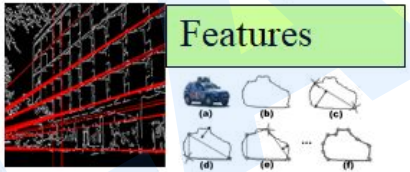


General Image Processing Functions

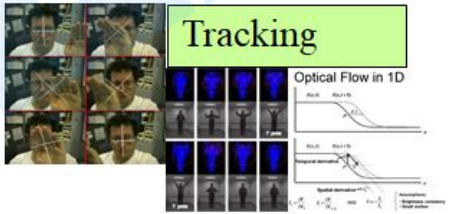

Segmentation


Transforms


Machine Learning:
•Detection,
•Recognition


Geometric Descriptors


Features


Tracking


Matrix Math

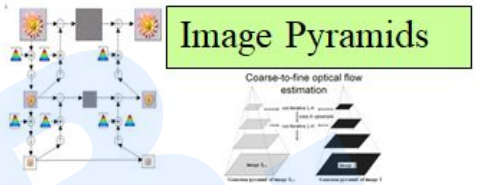
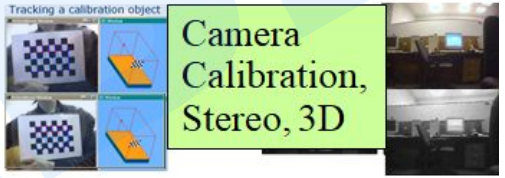
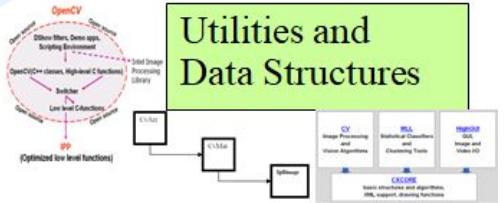
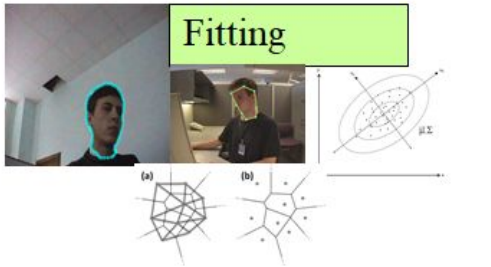


Image Pyramids


Camera Calibration, Stereo, 3D

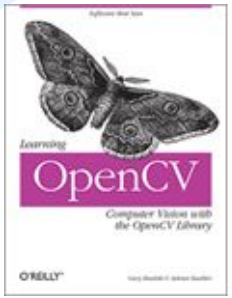

Utilities and Data Structures


Fitting

# OpenCV

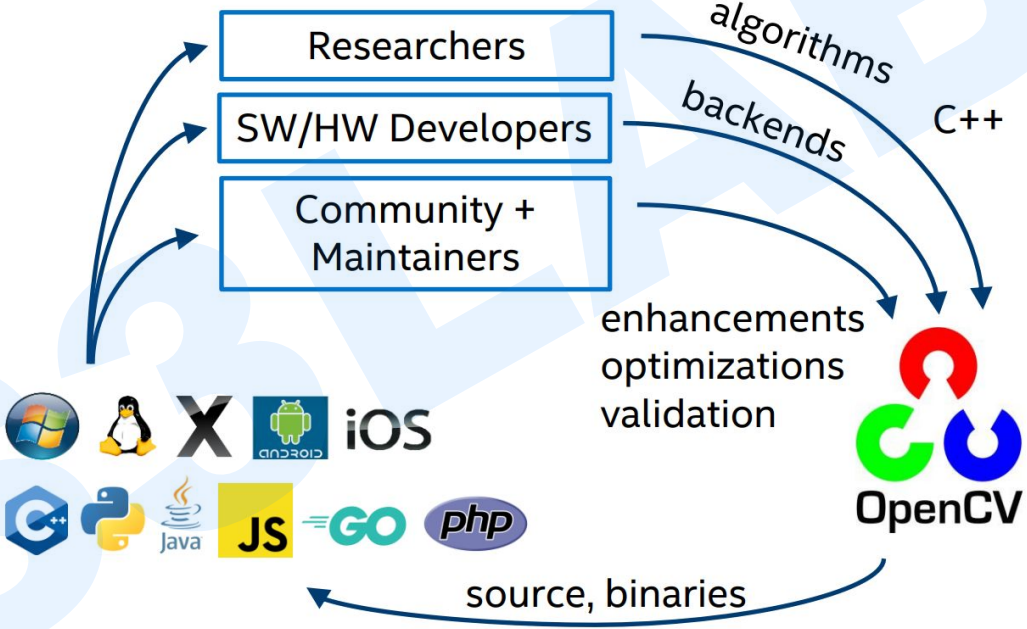
*Getting Start*

- Download OpenCV
  - http://opencv.org
- Install from macports/aptitude
- Setting up
  - Comprehensive guide on setting up OpenCV in various environments at the official wiki.
- Online Reference:
  - http://docs.opencv.org
- Two books



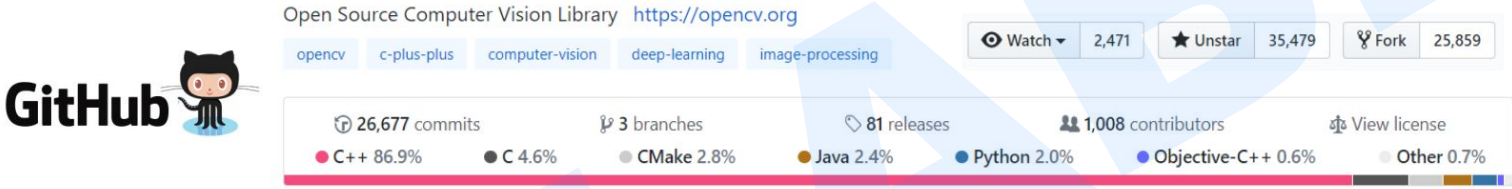

# OpenCV

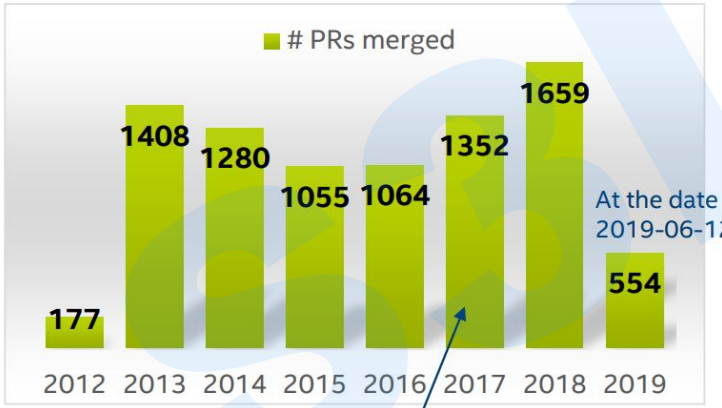
*Contribution to OpenCV as an investment*



# OpenCV

*Statistics*



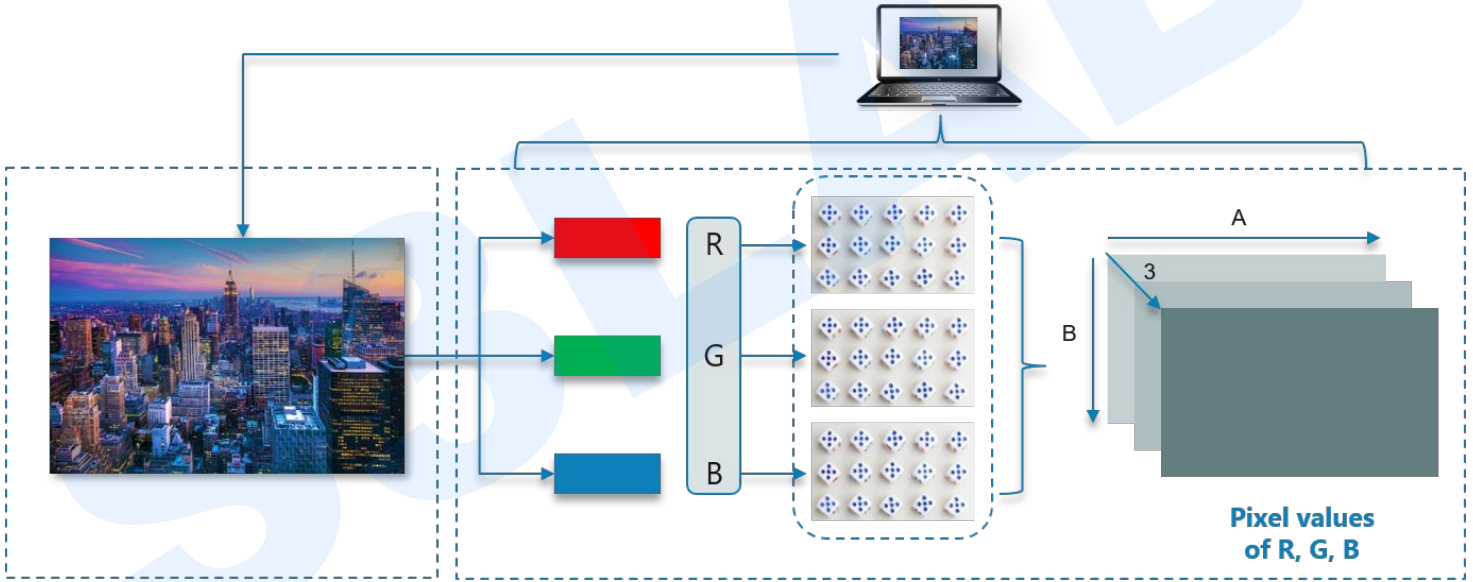




June 2016: Itseez is now Intel's IOTG Computer Vision (ICV) group

- >19M downloads from https://sourceforge.net
- ~5-6 pull requests are merged per working day
- 7th of TOP-10 trending C++ repos on GitHub (June, 2019)

# OpenCV

*Some Examples - Load Image*



# OpenCV

*Some Examples - Video Capture*

```
import cv2,time

video = cv2.VideoCapture(0)

a = 1

while True:
    a = a + 1
    check, frame = video.read()
    print (frame)

    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

    cv2.imshow('Capturing',gray)

    key = cv2.waitKey(1)

    if key == ord('q'):
        break

print(a) # This will print the number of frames
video.release()
cv2.destroyAllWindows
```

Convert each frame into a gray scale image

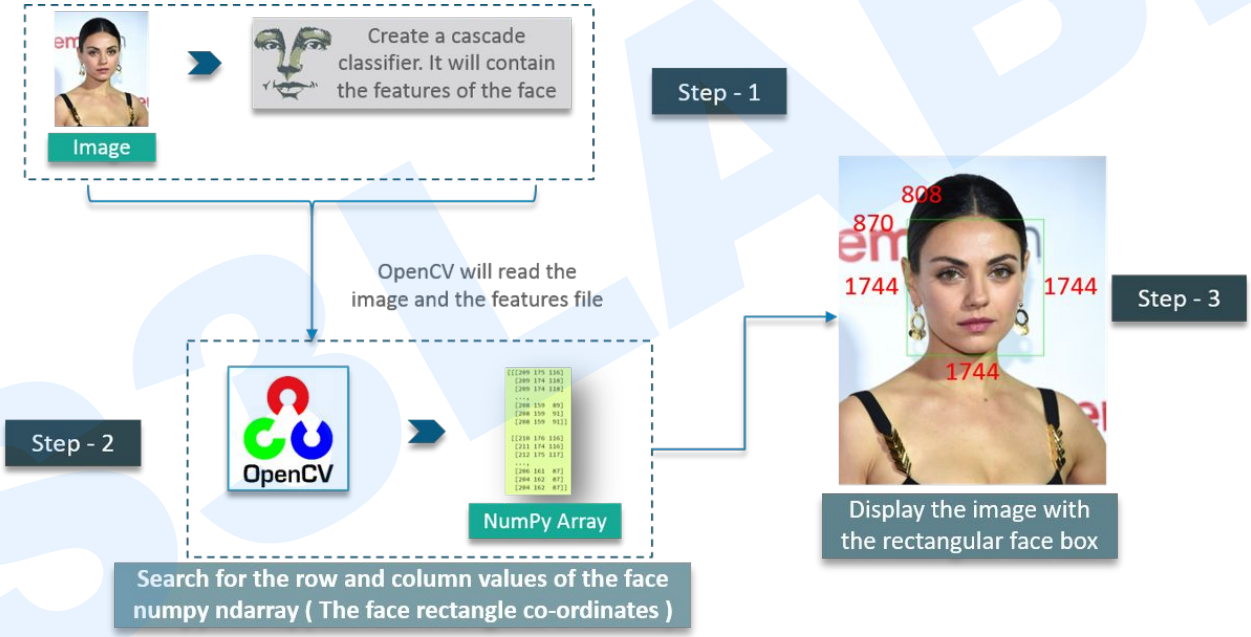
This will iterate through the frames and display the window

This will generate a new frame after every 1 miliseconds

Once you enter 'q' the window will be destroyed

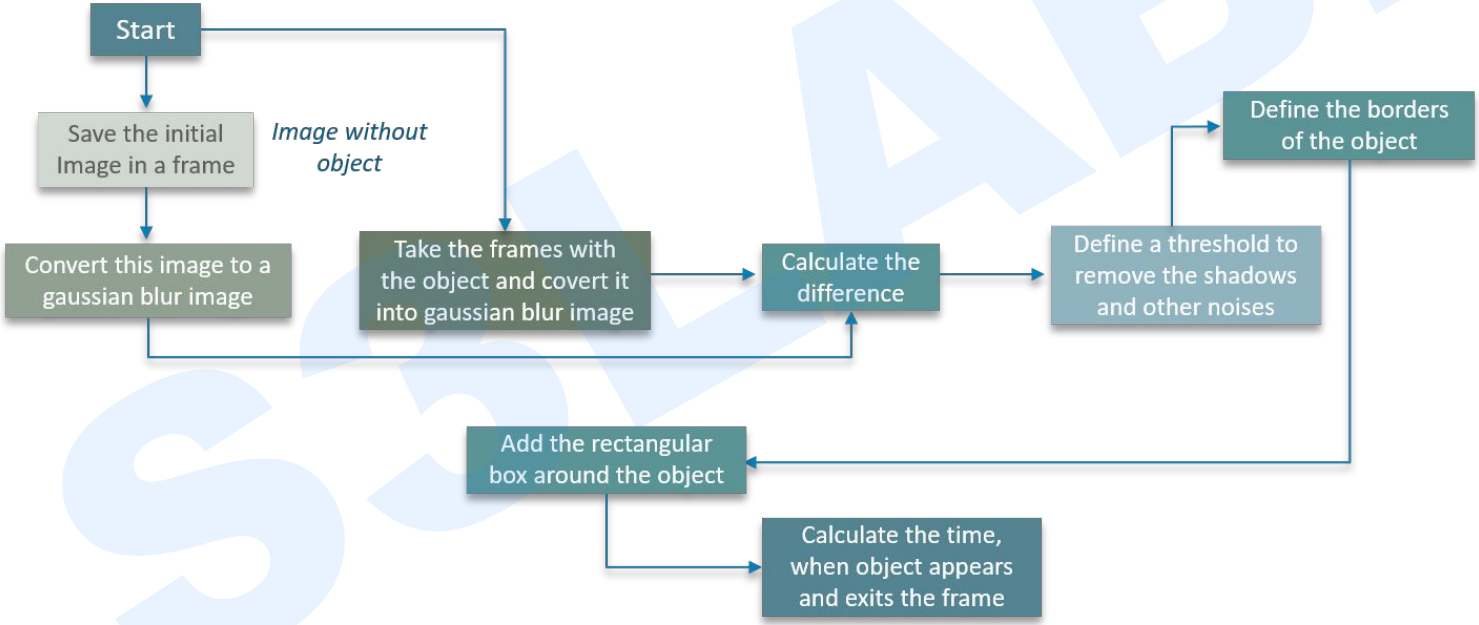
# OpenCV

*Some Examples - Face Detection*



# OpenCV

## *Some Examples - Motion Detection*



# OpenCV

*Top projects 2021*

https://www.youtube.com/watch?v=BFlRmIvqwSA&t=11s

# Android & Native Code

*JNI & Android NDK*

- **JNI**:
  - framework that allows Java code to call, or be called by, native applications/libraries in other languages. E.g., Java code can call a function written in C++, and visa-versa
  - Easiest way to integrate pre-existing code and libraries in other languages (e.g., a pre-existing C encryption library)
  - Good to implement certain functionality which can be written more optimally in C, for example.
  - Native code still runs within the application's VM: Follows AndroidManifest (e.g., need "INTERNET" for sockets)
- **Android NDK**: Toolset integrate/compile JNI in to Android app

# Android & Native Code

*Why write native code*

- To maximize performance:
  - You can improve the performance of an Android application, though only marginally, by implementing the CPU-intensive portions of its business logic in C++.
- To use high-performance APIs:
  - Implementations of API specifications such as Vulkan Graphics and OpenSL ES are a part of the NDK. Therefore, Android game developers tend to use the NDK.

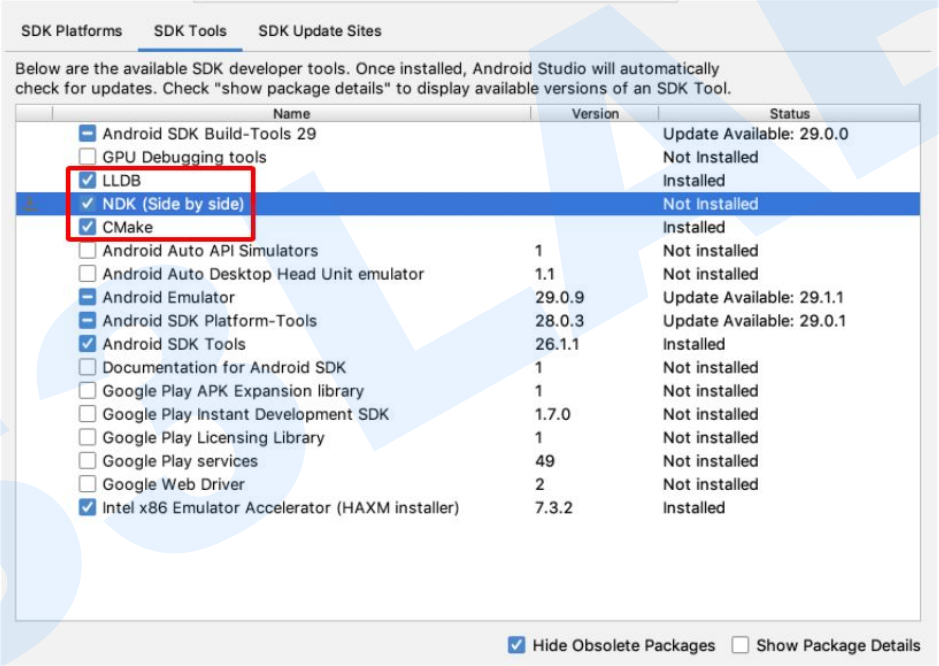
# Android & Native Code

*Why write native code*

- To use popular C/C++ libraries:
  - There are numerous C and C++ libraries out there that have no Java equivalents. If you want to work with them in your Android app, using the NDK is the way to go.
- To reuse code:
  - As long as it doesn't contain any platform-specific dependencies, code written in C++ can be used in both Android and iOS applications, usually with minimal changes. If you are developing a large application and intend to support both the iOS and Android platforms, using C++ might improve your productivity

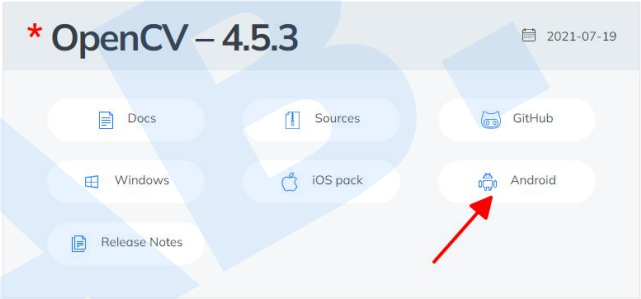
# Android & OpenCV

*How to use - Setup by ourself*



# Android & OpenCV

*How to use - Setup by ourself*




- Download OpenCV SDK for android.
  - https://opencv.org/releases/
- Import OpenCV to Android Studio
  - From File -> New -> Import Module, choose /sdk folder in the unzipped opencv archive. ( leave the default options checked and click next until finish)
- Fixing Gradle Sync Errors
  - Update build.gradle under imported OpenCV module to update 4 fields to match your project build.gradle a) compileSdkVersion b) buildToolsVersion c) minSdkVersion and d) targetSdkVersion.

# Android & OpenCV

*How to use - Setup by ourself*

- Add module dependency
  - File -> Project Structure, and select the Dependencies tab. Click + icon at bottom, choose Module Dependency and select the imported OpenCV module.
- Add Native Libraries
  - Copy libs folder under sdk/native to Android Studio under app/src/main.
- In Android Studio, rename the copied **libs** directory to **jniLibs** and we are done.
- Check SDK management to make sure the NDK tools is available

# Android & OpenCV

*How to use - With Template: OpenCV, JNI, C++*

- https://github.com/VISomers/native-opencv-android-template (latest)
  - Using similar setting about RDK version for Build.gradle (OPENCV)
    - compileSdkVersion
    - minSdkVersion
    - targetSdkVersion
  - Check current version of CMake and NDK match to Build.gradle (APP)
  - Carefully check the opencvsdk=C\:\\Users\\ASUS\\Downloads\\OpenCV-android-sdk on gradle.properties
  - If using Kolin, comment out the MainActivity.java and vice versa.

# Android & OpenCV

*How to use - With Template: OpenCV, JAVA WAPPER*

- https://github.com/quickbirdstudios/opencv-android

# Android & OpenCV

*Homework*

- Create a small application which utilizes OpenCV library to process the real-time video captured by camera in android phone.

# Q & A

## Thank you for listening

*“Coming together is a beginning;*
*Keeping together is progress;*
*Working together is success.”*
- HENRY FORD