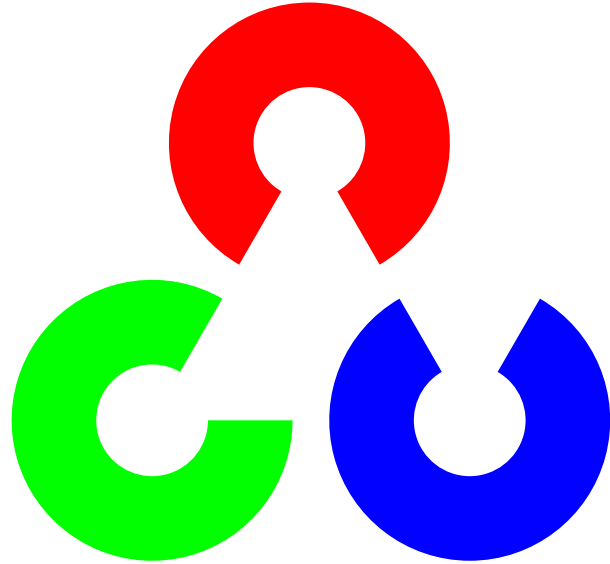


OpenCV ကိုလေ့လာခြင်း



OpenCV

Open Source Computer Vision

သင့်စက်အတွက်အမြင်

ကိုအေး

Yan Naing Aye @ ကိုအေး

Cool Emerald: <http://coolemerald.blogspot.com>

Typeset with Xe_{La}TeX for A4 paper.

ဒုတိယအကြိမ်တည်းဖြတ်ခြင်း - မတ် ၂၀၁၈။

ဤစာအုပ်၏ နောက်ဆုံးတည်းဖြတ်မှုကို

<https://yan9a.github.io/opencv/opencv.pdf> တွင် ရယူ နိုင်ပြီး ဖော်ပြ ထားသော နမူနာများကို <https://github.com/yan9a/opencv> တွင် တွေ့နိုင်သည်။

ဤစာအုပ်ကို လွတ်လပ်စွာ ကူးယူ၊ ဖြန့်ဝေ၊ အသုံးပြု နိုင်သည်။ ခွင့်ပြုထား သော လိုင်စင်မှာ CC-BY-4.0 ဖြစ်သည်။ ထို့ကြောင့် သင့်တင့် လျောက်ပတ်သော အသိအမှတ် ပြုမှု credit နှင့် link ဖော်ပြရန် လိုသည်။ အသေးစိတ်ကို <http://creativecommons.org/licenses/by/4.0/> တွင် တွေ့နိုင်သည်။

ကျေးဇူးတင်လွှာ

OpenCV အကြောင်း ရေးတော့ image processing နဲ့ ပတ်သက်ပြီး အကူအညီ တွေ ပေးခဲ့ ကြတဲ့ မိတ်ဆွေ တွေကို သတိ ရမိ၊ ကျေးဇူး တင်မိ ပါတယ်။ ၂၀၀၉ လောက်က C နဲ့ image processing တွေ လုပ်တော့ စာအုပ် တွေ ပို့ပေး ခဲ့တဲ့ ငွေမိုးအောင်ကို ကျေးဇူး တင်တဲ့ အကြောင်း ပြောချင် ပါတယ်။ Algorithm တွေ နဲ့ ပတ်သက်ပြီး အကြံ ညှဏ်ကောင်း တွေ အမြဲ ပေး ခဲ့တဲ့ ဇော်မင်းဦး ကို အရမ်း ကျေးဇူး တင် ပါတယ်။ ၂၀၁၀ လောက်က OpenCV နဲ့ မိတ်ဆက် ပေး၊ စာအုပ် တွေပါ ပေး ခဲ့တဲ့ ညဏ်ပိုင်စိုး ကိုလည်း အထူး ကျေးဇူး တင် ပါတယ်။ OpenCV နဲ့ ပတ်သက်ပြီး အားပေး၊ တိုက်တွန်းခဲ့ ကြတဲ့ နေဦး၊ သက်မွန်ထွေး တို့ကို လည်း သတိတရ ကျေးဇူး တင်မိ ပါတယ်။ နောက်ဆုံး အနေနဲ့ ကတော့ ကျွန်တော့် အပေါ် ချစ်ခင်၊ သည်းခံ၊ ဖြည့်ဆည်း ပေးတဲ့ ချစ်ဇနီး သီရိ ကို ကျေးဇူး တင်တဲ့ အကြောင်း ပြောပါရစေ။

မာတိကာ

ကျေးဇူးတင်လွှာ	i
မာတိကာ	ii
၁ မိတ်ဆက်	၁
၁.၁ နောက်ခံအကြောင်းအရာ	၁
၁.၂ ဖွဲ့စည်းပုံ	၂
၁.၃ Linux တွင် တပ်ဆင်ခြင်း	၃
၁.၃.၁ ရှိထားရန်လိုအပ်သည့် packages များ	၃
၁.၃.၂ apt ဖြင့်တပ်ဆင်ခြင်း	၄
၁.၃.၃ Source မှ build လုပ်ခြင်း	၄
၁.၃.၄ GCC ၊ CMake တို့ဖြင့် အသုံးပြုခြင်း	၆
၁.၃.၅ Qt ဖြင့် အသုံးပြုခြင်း	၉
၁.၃.၆ Code::Blocks ဖြင့် အသုံးပြုခြင်း	၁၂
၁.၄ OpenCV ကို Windows တွင် တပ်ဆင်ခြင်း	၁၉
၂ အခြေခံများ	၂၉
၂.၁ ရေးဖတ်ခြင်းများ	၂၉
၂.၁.၁ Pixel များကိုသုံးခြင်း	၃၀
၃ တည်ရာနယ်တွင်ပြုစပ်ခြင်းများ	၃၃
၃.၁ ချောမွတ်ရန်ပြုပြင်ခြင်းများ	၃၃
၃.၁.၁ boxFilter	၃၃

၃.၁.၂	blur	၃၄
၃.၁.၃	GaussianBlur	၃၅
၃.၁.၄	medianBlur	၃၅
၃.၂	filter2D	၃၇
၃.၃	Thresholding	၄၁
၃.၄	ပုံလုံးဖော်လုပ်ဆောင်ချက်များ	၄၇
၃.၄.၁	တိုက်စားခြင်း (erosion)	၄၇
၃.၄.၂	ဖောင်းပွခြင်း (dilation)	၄၇
၃.၄.၃	morphologyEx	၄၈
၃.၅	အရောင်နယ်ပြောင်းလဲခြင်းများ	၅၂
၄	ကြိမ်နှုန်းနယ်တွင်ပြုစပ်ခြင်းများ	၅၇
၄.၁	ကြိမ်နှုန်းနယ်ပယ်၏ Gaussian Filter များ	၅၈
၄.၂	ကြိမ်နှုန်းနယ်၏ Butterworth Filter များ	၅၈
၄.၃	Wiener Filtering	၆၅
၅	ဗီဒီယိုပြုစပ်ခြင်း	၇၅
၅.၁	ဗီဒီယို ဖတ်ခြင်းရေးခြင်း	၇၅
၅.၂	Real-time Face Detection	၇၉
၆	ပုံရိပ်ခွဲခြမ်းသုံးသပ်ရှာဖွေခြင်းများ	၈၅
၆.၁	Connected Component Labeling	၈၅
၆.၂	Template Matching	၈၉
၆.၃	Harris Corner Detector	၉၃
၆.၄	မြန်မာနိုင်ငံ၏ဗဟို	၉၄
၇	GUI Applications with wxWidgets	၁၀၁
၇.၁	Linux တွင်တပ်ဆင်ခြင်း	၁၀၂
၇.၁.၁	Source မှ build လုပ်ခြင်း	၁၀၄
၇.၂	OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း	၁၀၅
၇.၃	Code::Blocks တွင်တွဲသုံးခြင်း	၁၀၉

၇.၄	Windows တွင်တပ်ဆင်ခြင်း	၁၁၅
၇.၄.၁	Visual Studio ဖြင့်အသုံးပြုခြင်း	၁၁၅
၇.၄.၂	VS တွင် OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း	၁၁၇
၇.၅	Source ကုဒ်များ	၁၂၃
A	နောက်ဆက်တွဲ ရှင်းလင်းချက်	၁၂၅
၁.၁	စကားလုံးဖွင့်ဆိုချက်များ	၁၂၅
၁.၂	ThresholdTypes	၁၂၆

အခန်း ၁

မိတ်ဆက်

OpenCV သည် ပညာရေး အတွက် သာမက၊ စီးပွားရေး အတွက်ပါ အလကား သုံးစွဲခွင့် ရှိသော free software တစ်ခု ဖြစ်ပြီး၊ BSD license နှင့် ထုတ်သည် [Ope17]။ C/C++ ၊ Python ၊ Java များနှင့် တွဲဖက် အသုံးပြု နိုင်ပြီး၊ Windows ၊ Linux ၊ Mac OS များ အပေါ်တွင် သာမက iOS ၊ Android အစရှိသော mobile platform များ အတွက်ပါ အလုပ် လုပ်သည်။ OpenCV ကို ထိရောက် မြန်ဆန်စွာ တွက်ချက်မှု နှင့် အချိန် နှင့် အမျှအရေးကြီး သော real-time အသုံးချမှု များ အတွက် အဓိက ထားကာ ဒီဇိုင်း ထုဆစ် ပြုလုပ် ထားသည်။ C/C++ သုံး၍ ရေးထား သဖြင့် multi-core processing များ၏ ကောင်းကွက် ကိုလည်း အသုံးချ နိုင်သည်။ OpenCV ကိုအနုပညာ လုပ်ငန်း များမှ အစ၊ မိုင်းရှာခြင်း၊ မြေပုံများ ချိတ်ဆက်ခြင်း၊ အဆင့်မြင့် စက်ရုပ်များ ဖန်တီးခြင်း အထိ နယ်ပယ် အမျိုးမျိုး တွင် ကျယ်ကျယ် ပြန့်ပြန့် အသုံးပြု နေကြ သည်။

၁.၁ နောက်ခံအကြောင်းအရာ

OpenCV ၏ အဓိပ္ပါယ် မှာ Open Source Computer Vision Library ဖြစ်၍ မူရင်းကုဒ် များအား ဖွင့်ပြ ထားသည်။ ကွန်ပျူတာ နှင့် စက်အမြင် ဆိုင်ရာ လုပ်ငန်း များတွင် ကိုးကား အသုံးပြု စရာ ဆော့ဖ်ဝဲ library ဖြစ်သည်။ OpenCV ကို စတင် ဖန်တီးစဉ် မှစ၍ ကွန်ပျူတာ နှင့် စက်အမြင် ဆိုင်ရာ အများ သူငါ အလွယ် တကူ အသုံးပြု နိုင်သည့် ဘုံယန္တရား တစ်ခု ရရန် ရည်ရွယ်သည်။ စီးပွားရေး ထုတ်ကုန် များတွင်ပါ စက်အမြင် ဆိုင်ရာ အပိုင်း များတွင် တိုးတက်မှု ရှိစေရန် ဖြစ်သည်။ BSD-license နှင့် ပေးထားသည့် အတွက် စီးပွားရေး လုပ်ငန်း များတွင် အလွယ် တကူ ယူသုံး၊ ပြုပြင် နိုင်သည်။

ဤ library တွင် အကောင်းဆုံး ရစေရန် ချိန်ညှိ ထားသော နည်းလမ်း (algorithm) ပေါင်း ၂၅၀၀

ကျော် ရှိသည်။ နည်းဟောင်းများ မှစ၍ နောက်ဆုံးပေါ် နည်းလမ်းသစ် များအထိ ပြည့်စုံ နှစ်စဉ် ပါဝင်သည်။ ထို နည်းလမ်း များကို မျက်နှာ များကို ရှာဖွေ သိရှိခြင်း၊ မှတ်မိခြင်း (face detection and recognition)၊ အရာဝတ္ထု များကို ခွဲခြား သိရှိခြင်း (object identification)၊ ဗီဒီယို အတွင်းမှ လူ့ အမူအရာ များကို ခွဲခြား သတ်မှတ်ခြင်း (classification of human actions)၊ ရွေ့လျား နေသော အရာဝတ္ထု များနောက် လိုက်ခြင်း (tracking moving objects)၊ သုံးဘက်မြင် မော်ဒယ် ထုတ်ခြင်း (3D modeling)၊ ရုပ်ပုံ များကို ဆက်ခြင်း (stitching)၊ သိမ်းထား သည့် ပုံများ ထဲမှ တူသော ပုံကို ရွေးထုတ်ခြင်း၊ ရှုခင်း များကို မှတ်မိခြင်း၊ ဖြည့်စွက် အာရုံရိပ် (augmented reality) များ အတွက် အမှတ် အသား များ လုပ်ခြင်း တို့တွင် အသုံးပြု နိုင်သည်။

၁.၂ ဖွဲ့စည်းပုံ

OpenCV ကို အပိုင်း (module) များ ခွဲ၍ တည်ဆောက် ထားသည်။ ထို့ကြောင့် များစွာသော shared သို့မဟုတ် static libraries များ ပါဝင် နေသည်။ အဓိက အသုံး များသော အပိုင်း များမှာ အောက်ပါအတိုင်း ဖြစ်သည် [Ope17i]။

core အခြေခံ အချက်အလက် ပုံစံများ၊ multi-dimensional array Mat နှင့် အခြား အပိုင်းများ အတွက်ပါ လိုအပ်သော အခြေခံ လုပ်ဆောင်ချက် များ (core functionalities) ပါဝင်သည့် ကျစ်လစ် သည့် အခြေခံ ကျသော အပိုင်း ဖြစ်သည်။

imgproc ရုပ်ပုံ များကို ပုံထွက် ပြုပြင်သော ဖယ်တာ (filter) များ၊ ပုံ အရွယ် အစား၊ အမြင် ပြောင်းခြင်း များ၊ အရောင် ပြောင်းခြင်း များ၊ histogram အစ ရှိသည်တို့ ပါဝင်သော ပုံရိပ် ပြုစပ်ခြင်း (image processing) အပိုင်း ဖြစ်သည်။

video ဗီဒီယို များကို ခွဲခြမ်း သုံးသပ်၍ ရွေ့လျားမှု ကို ခန့်မှန်း ခြင်းများ၊ နောက်ခံ မြင်ကွင်း ကို ဖယ်ဖျောက် ခြင်းများ၊ အရာ ဝတ္ထု များ နောက် ခြေရာခံ ခြင်း များ တို့ ပါဝင်သည့် အပိုင်း ဖြစ်သည်။

calib3d ကင်မရာ ချိန်ညှိခြင်း (camera calibration)၊ object pose ခန့်မှန်း ခြင်း၊ stereo correspondence algorithms၊ သုံးဘက် မြင်ကွင်း ပြန် တည်ဆောက် ခြင်း (3D reconstruction) တို့ ပါဝင်သည်။

features2d ပုံများရှိ အဓိက အသွင် အပြင် (salient feature) များကို ရှာခြင်း နှင့် descriptor matcher များ ပါဝင်သည်။

objdetect သတ်မှတ် ထားသော အရာ ဝတ္ထု များကို ရှာခြင်း (ဥပမာ မျက်နှာ (face detection)၊ မျက်လုံး၊ မတ်ခွက်၊ လူ၊ ကား၊ စသည် များ)။

highgui ရိုးရှင်းသည့် UI (user interface) လုပ်ဆောင်မှု များအတွက် အလွယ် တကူ သုံးနိုင်သော in-terface ဖြစ်သည်။

Video I/O ဗွီဒီယို များ ဖမ်းယူ ခြင်း၊ နှင့် video codecs များ ပါဝင် သည်။

gpu OpenCV module များ အတွက် GPU accelerated algorithm များ ပါဝင် သည်။

အခြား အပိုင်း များလည်း ရှိသေး သည်။ လက်ရှိ OpenCV ကို တည်ဆောက် ထားပုံမှာ fully re-entable ဖြစ် သဖြင့် function တစ်ခု ကို မတူညီ သော thread များမှ ပြိုင်တူ သုံးနိုင် သည်။

၁.၃ Linux တွင် တပ်ဆင်ခြင်း

၁.၃.၁ ရှိထားရန်လိုအပ်သည့် packages များ

OpenCV ကို Linux တွင် တပ်ဆင် ရန် ပထမ အဆင့် အနေနှင့် အောက်ပါ packages များ စက်ထဲ တွင် ရှိရန် လိုအပ် သည် [Ope17g]။

1. GCC 4.4.x or later
2. CMake 2.6 or higher
3. Git
4. GTK+2.x or higher, including headers (libgtk2.0-dev)
5. pkg-config
6. Python 2.6 or later and Numpy 1.5 or later with developer packages (python-dev, python-numpy)
7. ffmpeg or libav development packages: libavcodec-dev, libavformat-dev, libswscale-dev
8. [optional] libtbb2 libtbb-dev

9. [optional] libdc1394 2.x

10. [optional] libjpeg-dev, libpng-dev, libtiff-dev, libjasper-dev, libdc1394-22-dev

ထို packages များအား စက်ထဲသို့ ထည့်သွင်း လိုပါက Synaptic Manager သုံး၍ သော်လည်းကောင်း၊ terminal တွင် အောက်ရှိ စာရင်း ၁.၁ နှင့် ၁.၂ ပါ command များ ရိုက်နှိပ်၍ သော်လည်းကောင်း ထည့်နိုင် သည်။

```
1 $ sudo apt-get update
2 $ sudo apt-get install build-essential
3 $ sudo apt-get install cmake git libgtk-3-dev pkg-config libavcodec-dev
   libavformat-dev libswscale-dev
```

စာရင်း ၁.၁: OpenCV အတွက် လိုအပ်သော packages များ ရယူခြင်း။

```
1 $ sudo apt-get install python-dev python-numpy libtbb2 libtbb-dev libjpeg-dev
   libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```

စာရင်း ၁.၂: OpenCV အတွက် optional packages များ ရယူခြင်း။

၁.၃.၂ apt ဖြင့်တပ်ဆင်ခြင်း

OpenCV ကို Linux တွင် ရယူ တပ်ဆင် ရန် လွယ်ကူ ရိုးရှင်း သော နည်းလမ်း တစ်ခု အဖြစ် အောက်ပါ အတိုင်း terminal တွင် ရိုက်ယူ နိုင်သည်။

```
$ sudo apt-get install libopencv-dev
```

၁.၃.၃ Source မှ build လုပ်ခြင်း

ထိုသို့ မဟုတ်ပဲ လက်ရှိ OpenCV အခြေကျ ဗာရှင်း ကို ရယူ တပ်ဆင်လို ပါက [OpenCV for Linux/Mac \(https://opencv.org/\)](https://opencv.org/) တွင် ရယူ ရန်လိုသည်။ ထိုမှ ရလာသော zip ဖိုင်အား Archive Manager သုံး၍ extract လုပ်နိုင်သည်။

ထို အခြေကျ ဗားရှင်း ကို မယူပဲ နောက်ဆုံးထွက် cutting-edge opencv ဗားရှင်း ကို ရယူ မည် ဆိုပါက Git repository ရှိ [OpenCV repository](#) တွင် ရယူ နိုင်သည်။ [OpenCV contrib repository](#) များ ကိုပါ တပ်ဆင် မည် ဆိုပါက လည်း အောက်ပါ အတိုင်း ယူနိုင် သည်။

```
$ cd ~
$ git clone https://github.com/opencv/opencv.git
$ git clone https://github.com/opencv/opencv_contrib.git
```

ဤနေရာတွင် ရလာသည့် folder မှာ opencv ဖြစ်သဖြင့် ထို နေရာသို့ သွား၍ build ဆိုသည့် folder တစ်ခု ဖန်တီးကာ ဖိုင်များ ထုတ်၍ သိမ်းဆည်းရန် အောက်ပါ စာရင်း ၁.၃ ရှိ command များ ရိုက်မည်။ Shared libs ကို unset လုပ်ချင် ပါက စာရင်း ၁.၄ ရှိ option ကို ထည့်နိုင် သည်။

```
1 $ cd ~/opencv
2 $ mkdir build
3 $ cd build
4 $ cmake -D CMAKE_BUILD_TYPE=Release \
5 -D CMAKE_INSTALL_PREFIX=/usr/local \
6 -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules ..
```

စာရင်း ၁.၃: OpenCV ကို build လုပ်ခြင်း။

```
1 -D BUILD_SHARED_LIBS=OFF
```

စာရင်း ၁.၄: Static libs ပြုလုပ်ခြင်း။

ခေတ္တ စောင့်ဆိုင်း ပြီးသည့် အခါ စာရင်း ၁.၅ အတိုင်း build လုပ်၍ install လုပ်မည်။ တပ်ဆင် ပြီးသည့် OpenCV ဗားရှင်း ကို စာရင်း ၁.၆ တွင် ပြထား သော command ဖြင့် ကြည့် နိုင်သည်။

```
1 $ make
2 $ sudo make install
```

စာရင်း ၁.၅: OpenCV ကို install လုပ်ခြင်း။

```
1 $ pkg-config --modversion opencv
```

စာရင်း ၁.၆: OpenCV ဗားရှင်းကို စစ်ခြင်း။

၁.၃.၄ GCC ၊ CMake တို့ဖြင့် အသုံးပြုခြင်း

OpenCV ကို သုံးရန် အလွယ်ဆုံး နည်းမှာ CMake ဖြင့် သုံးခြင်း ဖြစ်သည် [Ope171]။ CMake နှင့် မရင်းနှီးပါက [CMake tutorial \(https://cmake.org/cmake-tutorial/\)](https://cmake.org/cmake-tutorial/) တွင် သွား ရောက် လေ့လာ နိုင်သည်။

ပထမ ခြေလှမ်း အနေ နှင့် ပုံတစ်ပုံ ကို ဖတ်၍ ပြသည့် ရိုးရှင်းသည့် နမူနာ လေးအား စမ်းသပ်ကြည့်မည်။ ထို အတွက် thiri.jpg ဆိုသည့် ဓာတ်ပုံကို home folder တွင် ထားလိုက် မည်။ ထို့နောက် စာရင်း ၁.၇ တွင် ဖော်ပြ ထားသည့် DisplayImage.cpp ဆိုသည့် ပရိုဂရမ် လေးအား ဖန်တီးလိုက်မည်။ ပြီးသည့် အခါ DisplayImage ဟူသည့် folder တစ်ခု ဖွဲ့၍ ထိုထဲ တွင် သိမ်းဆည်းလိုက်မည်။

```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main(int argc, char** argv )
5 {
6     Mat image;
7     image = imread( "/home/yan/thiri.jpg", 1 );
8     if ( !image.data ) {
9         printf("No image data \n");
10        return -1;
11    }
12    namedWindow("Display Image", WINDOW_AUTOSIZE );
13    imshow("Display Image", image);
14    waitKey(0);
15    return 0;
16 }
```

စာရင်း ၁.၇: ပုံ တစ်ပုံ ကို ဖတ်၍ ပြသည့် DisplayImage ပရိုဂရမ်။

ဤနေရာတွင် thiri.jpg အတွက် path မှာ `"/home/yan/thiri.jpg"` ဖြစ်ပြီး သင့်ပုံ ရှိသည့် folder ၊ username တို့နှင့် ကိုက်ညီသည့် path ကို အစားထိုး ရမည်။ `imread` သည် လမ်းကြောင်း ပေးလိုက်သည့် ပုံဖိုင်ကို ဖတ်သည်။ ဒုတိယ argument ဖြစ်သည့် 1 မှာ ကာလာပုံ ဖတ်မည် ဟု ဆိုလိုခြင်း ဖြစ်သည်။ 0 ဆိုပါက အဖြူ အမည်း ပြောင်း၍ ဖတ်မည်။ ပုံကို ဖတ်၍ မရ ပါက message ရိုက်ပြ၍ ထွက်သွားမည် ဖြစ်ပြီး၊ ဖတ်၍ အောင်မြင် ပါက `imshow` ကိုသုံး၍ ပုံကို ထုတ်ပြမည် ဖြစ်သည်။ တဖန် `CMakeLists.txt` ဆိုသည့် ဖိုင်ကို အောက်ပါ စာရင်း ၁.၈ အတိုင်း ဖန်တီးမည်။

```
1 cmake_minimum_required(VERSION 2.8)
2 project( DisplayImage )
3 find_package( OpenCV REQUIRED )
4 add_executable( DisplayImage DisplayImage.cpp )
5 target_link_libraries( DisplayImage ${OpenCV_LIBS} )
```

စာရင်း ၁.၈: CMakeLists.txt

ဤတွင် လိုချင်သည့် ပရိုဂရမ် ကို အောက်ပါ စာရင်း ၁.၉ ရှိ command များအား terminal တွင် ရိုက်ခြင်းဖြင့် ထုတ်လုပ် ရရှိနိုင်၊ run ကြည့်နိုင် သည်။

```
1 $ cd DisplayImage
2 $ cmake .
3 $ make
4 $ ./DisplayImage
```

စာရင်း ၁.၉: DisplayImage ကို CMake ဖြင့် build လုပ်၍ run ခြင်း။

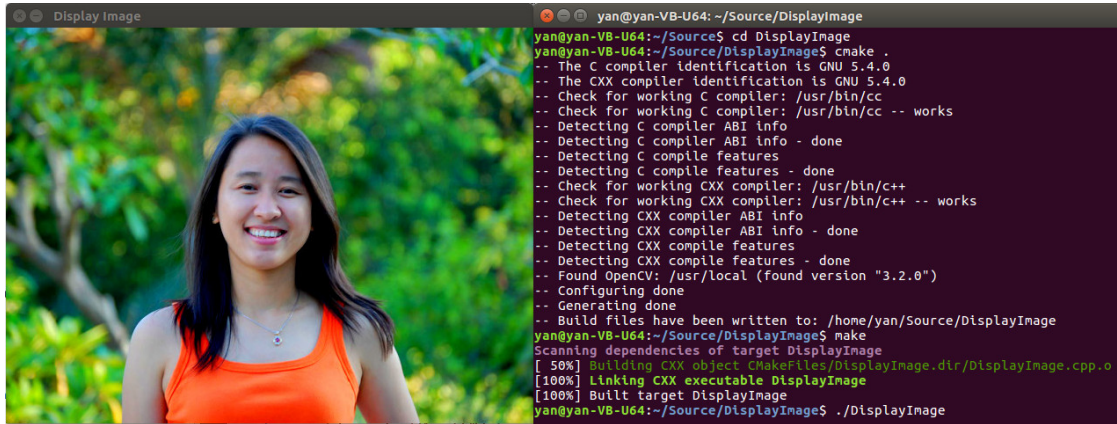
pkg-config

CMake ကို မသုံးပဲ pkg-config ဖြင့် စာရင်း ၁.၁၀ အတိုင်း build လုပ်၍ လည်း run နိုင်သည်။

```
1 $ g++ DisplayImage.cpp `pkg-config --cflags --libs opencv` -o DisplayImage
2 $ ./DisplayImage
```

စာရင်း ၁.၁၀: DisplayImage ကို g++, pkg-config တို့ဖြင့် build လုပ်၍ run ခြင်း။

Run လိုက်သည့် အခါ အောက်ပါ ပုံ ၁.၁ အတိုင်း ရရှိမည်။



ပုံ ၁.၁: DisplayImage ပရိုဂရမ် ၏ ရလဒ်ကို terminal နှင့် ယှဉ်တွဲ ပြထား သည်။

Shared lib အား ရှာ မရ သည့် error ရခဲ့ လျှင် /etc/ld.so.conf.d/opencv.conf ဟူသည့် ဖိုင်အား အောက်ပါ အတိုင်း ဖန်တီး နိုင်သည်။

```
$ sudo nano /etc/ld.so.conf.d/opencv.conf
```

ပြီးသည့် အခါ opencv ကို တပ်ဆင် ထားသည့် နေရာ ပေါ် မူတည်၍

```
/usr/local/opencv/
```

သို့မဟုတ်

```
/usr/local/lib/
```

ကို opencv.conf တွင် ဖြည့်မည်။ ထို့နောက် အောက်ပါ command အား terminal တွင် ရိုက်ထည့် နိုင်သည်။

```
$ sudo ldconfig
```

၁.၃.၅ Qt ဖြင့် အသုံးပြုခြင်း

အသုံးပြုသူများသော IDE တစ်ခု ဖြစ်သည့် Qt နှင့် OpenCV ကို အသုံးပြုခြင်း အကြောင်း ဆွေးနွေးမည်။ Qt ကို ရယူ၍ တပ်ဆင်ရန် အောက်ပါ စာရင်း ၁.၁၁ ရှိ command များကို terminal တွင်ရိုက်နိုင်သည်။

```
1 $ wget http://download.qt.io/official_releases/qt/5.9/5.9.0/qt-opensource-
    linux-x64-5.9.0.run
2 $ chmod +x qt-opensource-linux-x64-5.9.0.run
3 $ ./qt-opensource-linux-x64-5.9.0.run
```

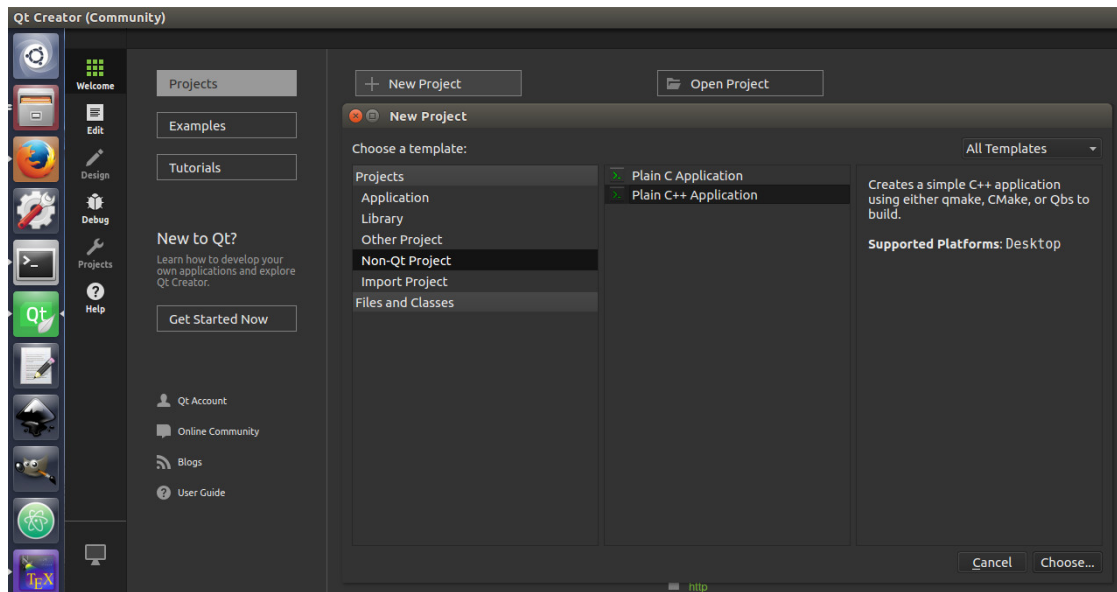
စာရင်း ၁.၁၁: Qt ကို Linux တွင် တပ်ဆင် ခြင်း။

ထို့နောက် အခြား လိုအပ်သော g++ ၊ font configuration ၊ OpenGL များ ရယူ ရန် အောက်ပါ စာရင်း ၁.၁၂ အတိုင်း လုပ်ဆောင် နိုင်သည်။

```
1 $ sudo apt-get install build-essential
2 $ sudo apt-get install libfontconfig1
3 $ sudo apt-get install mesa-common-dev
4 $ sudo apt-get install libglu1-mesa-dev -y
```

စာရင်း ၁.၁၂: Qt အတွက် အခြား လိုအပ်သော packages များရယူ ခြင်း။

တပ်ဆင် ပြီးပါက ရိုးရှင်းသော OpenCV နမူနာ တစ်ခုကို စမ်းသပ်ရန် Qt ကို ဖွင့်မည်။ New Project ကို နှိပ်၍ Non-Qt Project နှင့် Plain C++ Application ကို choose မည် (ပုံ ၁.၂)။



ပုံ ၁.၂: Qt တွင် new project တစ်ခုကို ဖန်တီးခြင်း။

Project Location အတွက် DisplayImageQt စာဖြင့် အမည်ပေး၊ folder ဖွဲ့၍ ရွေးချယ် မည်။ Build System တွင် qmake အတိုင်း ထားမည်။ Kit Selection တွင်လည်း Desktop Qt 5.9.0 GCC 64bit အတိုင်း ထားနိုင် သည်။ Next ထပ်နှိပ် ၍ Project Management တွင် None အတိုင်း ထားမည်။ ထို့နောက် Project အောက်မှ Sources ရှိ `main.cpp` တွင် အောက်ပါ စာရင်း ၁.၁၃ ရှိ ပရိုဂရမ် အတိုင်း ရေးမည်။

```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main(int argc, char** argv )
5 {
6     Mat image,bw_img;
7     image = imread( "/home/yan/thiri.jpg", 1 );
8     if ( !image.data ) {
9         printf("No image data \n");
10        return -1;
11    }
12    namedWindow("Original Image", WINDOW_AUTOSIZE );
13    imshow("Original Image", image);

```

```

14 cvtColor(image, bw_img, COLOR_BGR2GRAY);
15 namedWindow("BW Image", WINDOW_AUTOSIZE );
16 imshow("BW Image", bw_img);
17 waitKey(0);
18 return 0;
19 }

```

စာရင်း ၁.၁၃: ပုံ တစ်ပုံကို ဖတ်ပြ၍ အဖြူ အမည်း ပြောင်းပေးသော ပရိုဂရမ်။

ဤ ပရိုဂရမ် တွင် ပုံကို ဖတ်၍ ရသည့် အခါ cvtColor ကို သုံး၍ အဖြူ အမည်း ပုံ ဖြစ်အောင် ပြောင်းသည်။ တဖန် DisplayImageQt.pro ဟူသည့် ဖိုင်ကို ဖွင့်၍ အောက်ပါ စာရင်း ၁.၁၄ အတိုင်း ဖြစ်အောင် ထပ်ဖြည့် မည်။

```

1 TEMPLATE = app
2 CONFIG += console c++11
3 CONFIG -= app_bundle
4 CONFIG -= qt
5 SOURCES += main.cpp
6 INCLUDEPATH += /usr/local/include/opencv
7 LIBS += -L/usr/local/lib -lopencv_core -lopencv_highgui -lopencv_imgproc -
    opencv_imgcodecs

```

စာရင်း ၁.၁၄: DisplayImageQt.pro

ပြီးနောက် ပရိုဂရမ် အား run ကြည့်ပါက ပုံ ၁.၃ အတိုင်း သီရိ၏ မူရင်း နှင့် အဖြူ အမည်း ပုံများ လှပစွာ ပေါ်ထွက် လာမည်။



ပုံ ၁.၃: ပရိုဂရမ်မှ ထွက်ပေါ်လာသည့် သီရိ၏ မူရင်းပုံ နှင့် အဖြူ အမည်း ပုံများ။

၁.၃.၆ Code::Blocks ဖြင့် အသုံးပြုခြင်း

OpenCV C/C++ ပရိုဂရမ် များ ရေးသားရန် အတွက် Code::Blocks ဆိုသည့် open source လည်းဖြစ်၊ free လည်း ပေးသော IDE နှင့်တွဲသုံး ကြည့်မည်။ ထို IDE သည် extension များ အမြောက် အမြား ရှိပြီး၊ စိတ်ကြိုက် configure လည်း လုပ်နိုင် သည့် အတွက် အလွန် ခေတ်စား သော IDE တစ်ခု ဖြစ်သည်။ Code::Blocks IDE ကို တပ်ဆင် ရန် terminal တွင် စာရင်း ၁.၁၅ အတိုင်း ရိုက်ထည့် နိုင်သည်။

```

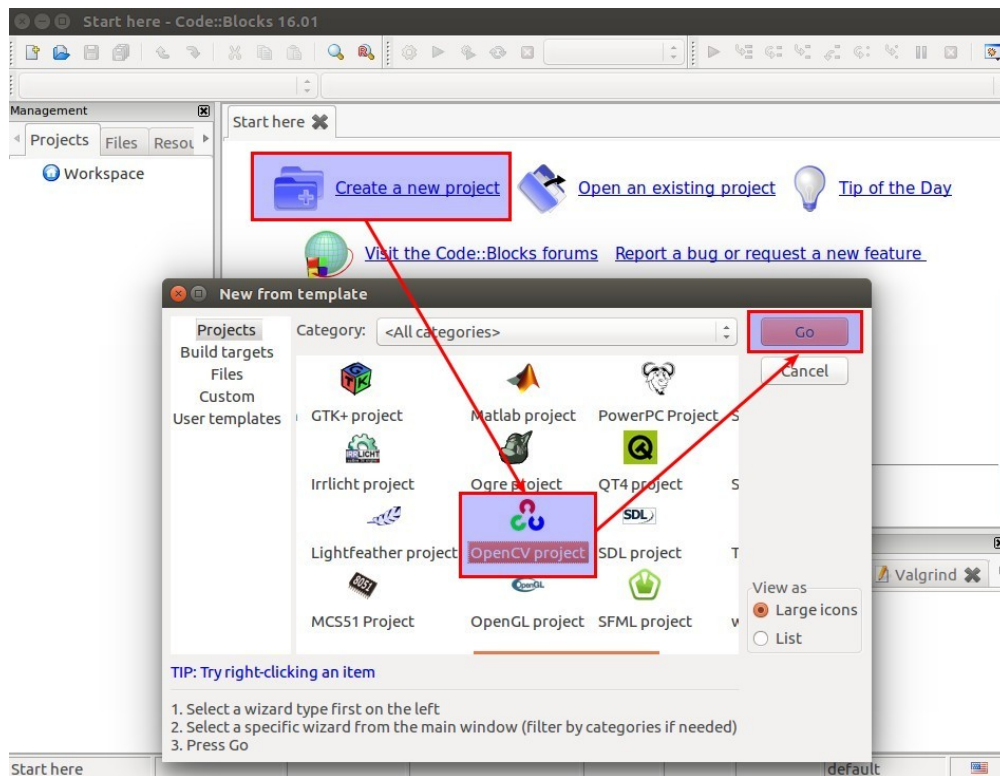
1 $ sudo apt-get install build-essential
2 $ sudo apt-get install libgtk-3-dev
3 $ sudo apt-get install gdb
4 $ sudo add-apt-repository ppa:damien-moore/codeblocks-stable
5 $ sudo apt-get update
6 $ sudo apt-get install codeblocks codeblocks-contrib

```

စာရင်း ၁.၁၅: Code Blocks တပ်ဆင်ခြင်း။

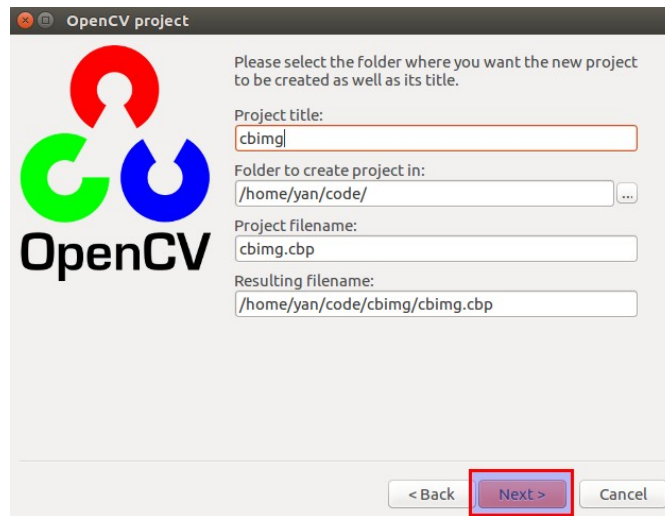
Code::Blocks ကို တပ်ဆင် ပြီးသည့် အခါ သူ့ကို ဖွင့်လိုက်ပြီး ပုံ ၁.၄ တွင် ပြထား သလို File menu → New → Project... သို့မဟုတ် Start here ဆိုသည့် tab ရှိ Create a new project ကို နှိပ်နိုင် သည်။ ပြီးသည့် အခါ ပေါ်လာသည့် ဝင်းဒိုး တွင် project အမျိုးအစား သတ်မှတ်ရန် OpenCV project

ကို ရွေးပြီး Go ကို နှိပ်မည်။

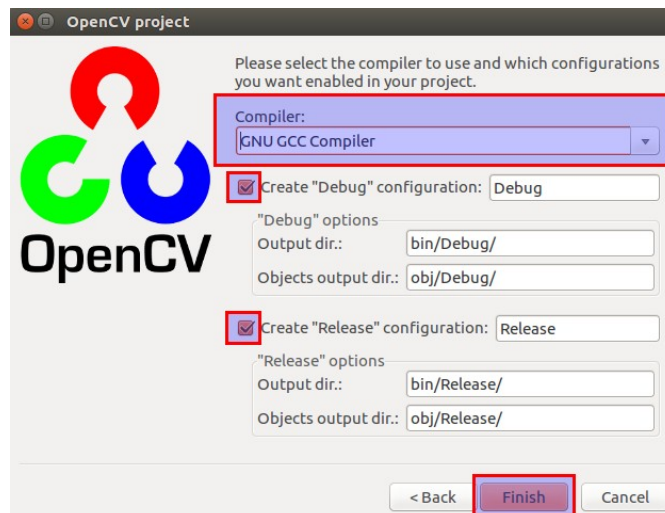


ပုံ ၁.၄: Code::Blocks IDE တွင် OpenCV project အသစ် ဖန်တီးမှု။

ပြီးသည့် အခါ Next ကို နှိပ်ပြီး ပရောဂျက် အမည် နှင့် သိမ်းမည့် နေရာ များ သတ်မှတ် မည်။
ထို့နောက် Compiler ကို GNU GCC compiler ကို ပုံ ၁.၅ အတိုင်း ရွေးပြီး Finish ကို နှိပ်မည်။



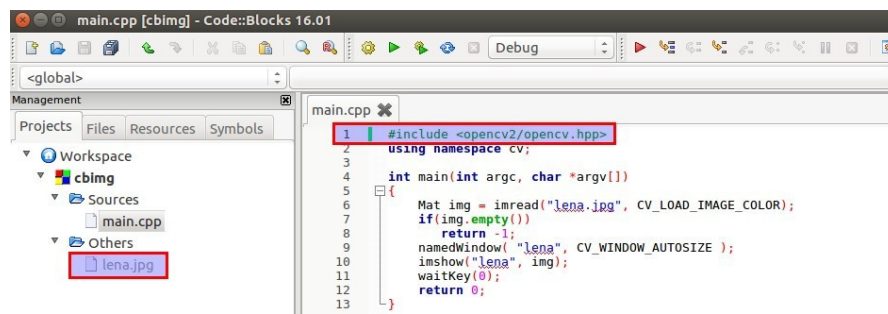
(a) နာမည်နှင့်နေရာ သတ်မှတ်ခြင်း။



(b) Compiler သတ်မှတ်ခြင်း။

ပုံ ၁.၅: Code::Blocks တွင် OpenCV အသုံးပြုခြင်း အဆင့်ဆင့်။

ထိုအခါ ဖန်တီး ရရှိ လာသော ပရောဂျက် တွင် နမူနာ OpenCV ကုဒ်များ နှင့် ထိုကုဒ်တွင် သုံးထားသော lena.jpg ကိုပါ တွေ့ရမည်။ Header အတွက် “#include <opencv2/opencv.hpp>” ဟု ပြင်ပြီး သည့် အခါ ပုံ ၁.၆ နှင့် စာရင်း ၁.၁၆ အတိုင်း တွေ့ရမည်။



ပုံ ၁.၆: Source နမူနာအား ပြုပြင် စမ်းသပ်ခြင်း။

```

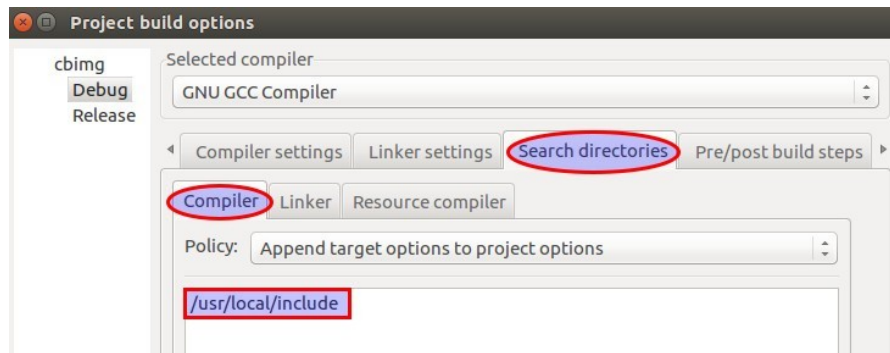
1 #include <opencv2/opencv.hpp>
2 using namespace cv;
3
4 int main(int argc, char *argv[])
5 {
6     Mat img = imread("lena.jpg", CV_LOAD_IMAGE_COLOR);
7     if(img.empty())
8         return -1;
9     namedWindow( "lena", CV_WINDOW_AUTOSIZE );
10    imshow("lena", img);
11    waitKey(0);
12    return 0;
13 }

```

စာရင်း ၁.၁၆: Code::Blocks အတွက် နမူနာ OpenCV ပရိုဂရမ်။

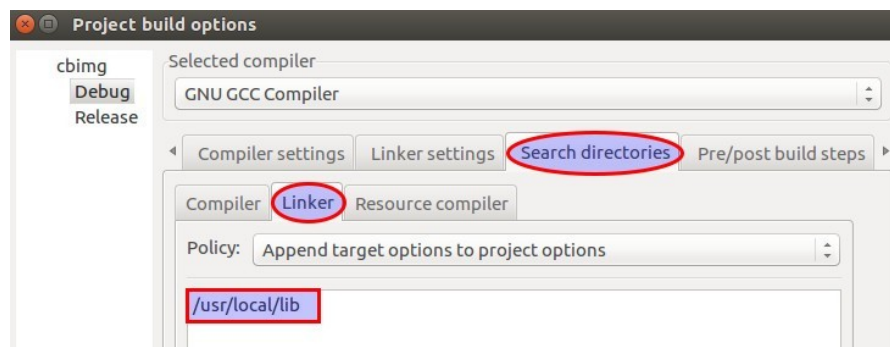
Build Options များသတ်မှတ်ခြင်း

Project Menu → Build Options... ကို နှိပ်ပြီး Search directories tab → Compiler tab တွင် ပုံ ၁.၇ အတိုင်း /usr/local/include ကို Debug နှင့် Release နှစ်ခုလုံး အတွက် သတ်မှတ် နိုင်သည်။

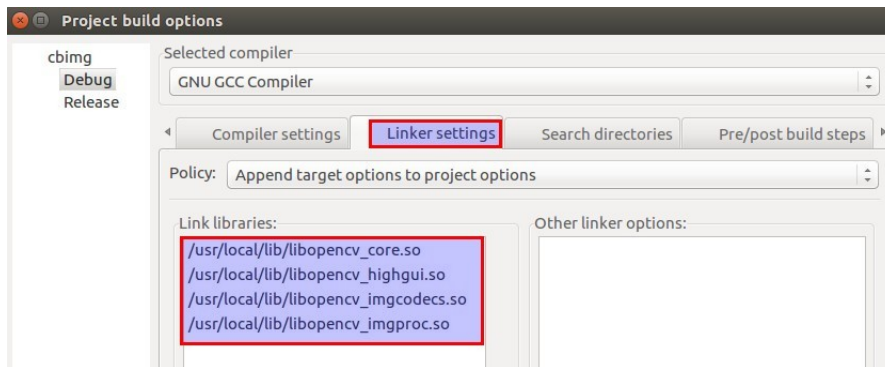


ပုံ ၁.၇: Compiler Search directories များသတ်မှတ်ခြင်း။

ပြီးသည့် အခါ ဘေးဘက် ရှိ Linker tab တွင် ပုံ ၁.၈ အတိုင်း lib များ၏ လမ်းကြောင်း /usr/local/lib ကို သတ်မှတ် မည်။ နောက်တစ်ဖန် Linker settings tab တွင် ပုံ ၁.၉ အတိုင်း Link libraries များအား သတ်မှတ်မည်။

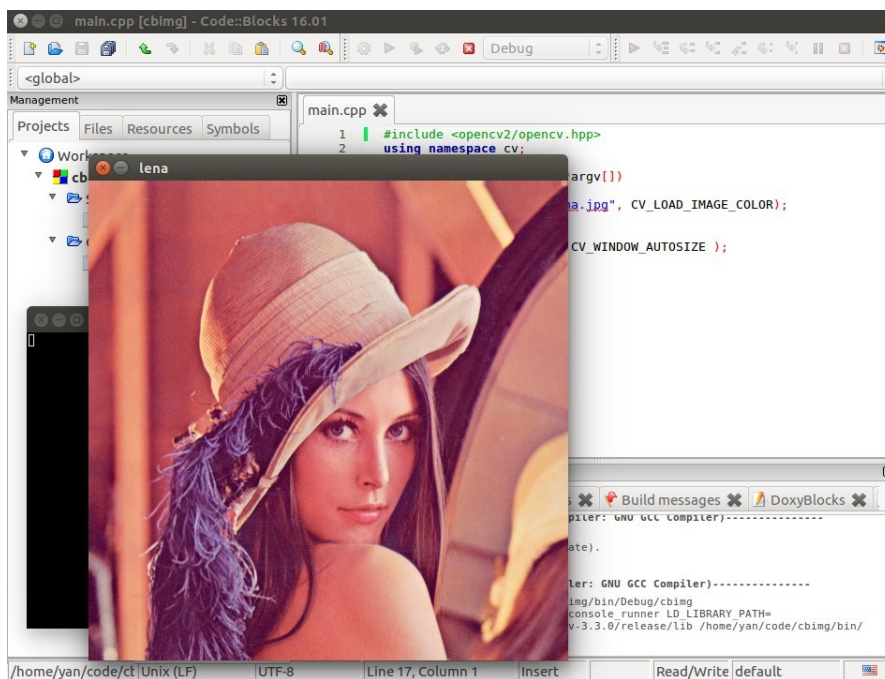


ပုံ ၁.၈: Library ဖိုင်များအတွက် Search directories များသတ်မှတ်ခြင်း။



ပုံ ၁.၉: Link libraries များသတ်မှတ်ခြင်း။

နောက်ဆုံးတွင် F9 ခလုတ်ကို နှိပ်ပြီး Build and run လုပ်လိုက်လျှင် ပုံ ၁.၁၀ တွင် ပြထား သကဲ့သို့ ပရိုဂရမ် ၏ output ကို တွေ့နိုင် မည်။



ပုံ ၁.၁၀: OpenCV နမူနာ ပရိုဂရမ်ကို Code::Blocks တွင် run ခြင်း။

pkg-config သုံးခြင်း

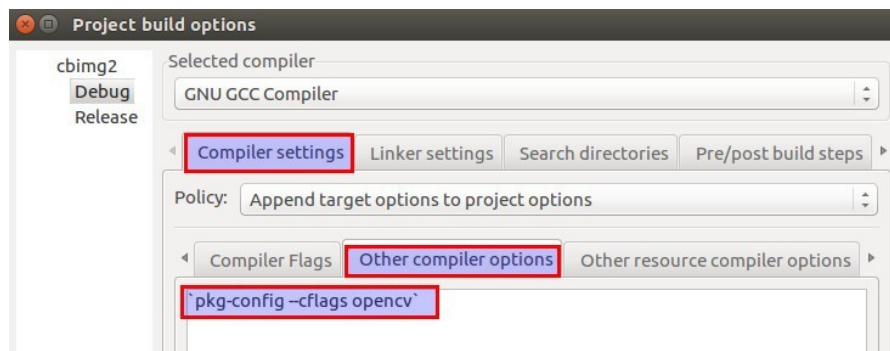
Build Options များကို pkg-config ကို သုံး၍ သတ်မှတ်မည် ဆိုပါက လည်း ရသည်။ Project Menu → Build Options... ကို နှိပ်ပြီး Compiler settings tab → Other compiler options tab တွင် ပုံ ၁.၁၁ အတိုင်း

```
`pkg-config --cflags opencv`
```

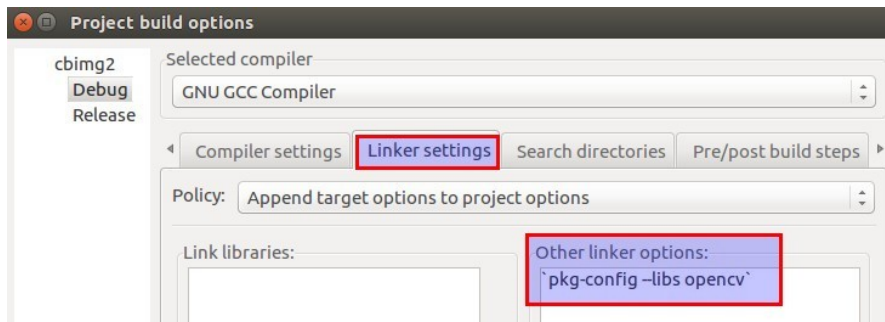
ကို Debug နှင့် Release နှစ်ခုလုံး အတွက် သတ်မှတ် နိုင်သည်။ ထို နည်းတူ Linker settings tab → Other linker options tab တွင် ပုံ ၁.၁၂ အတိုင်း

```
`pkg-config --libs opencv`
```

ကို သတ်မှတ်မည်။



ပုံ ၁.၁၁: pkg-config ကို compiler settings တွင်သတ်မှတ်ခြင်း။



ပုံ ၁.၁၂: pkg-config ကို linker settings တွင်သတ်မှတ်ခြင်း။

ဤ ပရောဂျက် (online) ကို F9 ခလုတ်ကို နှိပ်ပြီး Build and run လုပ်လိုက်လျှင် ပုံ ၁.၁၀ တွင် ပြထား သည့် အတိုင်း တူညီသည့် ရလဒ်ကို ရနိုင်သည်။

၁.၄ OpenCV ကို Windows တွင် တပ်ဆင်ခြင်း

Windows အတွက်မူ pre-built လုပ်ပြီး ဖြစ်သော လက်ရှိ နောက်ဆုံးထွက် OpenCV အခြေကျ ဗာရှင်း ကိုသုံးမည်။ ၎င်းအား OpenCV for Windows <https://opencv.org/> တွင် ယူနိုင်သည်။ ရရှိ လာသော opencv-3.3.0-vc14.exe အား ဤ နမူနာ အတွက် C:\opencv အောက်တွင် opencv330 အနေနှင့် extract လုပ်သည်။

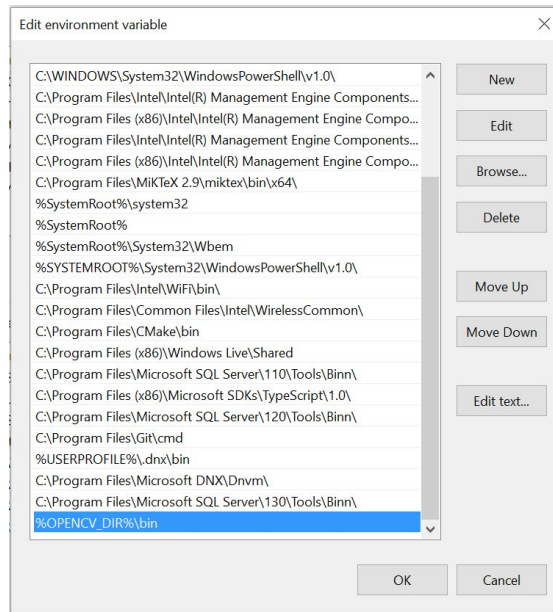
IDE အတွက် Visual Studio Community ကို Visual Studio Downloads <https://www.visualstudio.com/downloads/> မှ ရယူ ချ် install လုပ်မည်။ အောက်ပါ အဆင့် များကို Windows 10 64 bit တွင် Visual Studio 2017 နှင့် OpenCV-3.3.0 အား သုံး၍ စမ်းကြည့် ထားသည်။ အခြား ဗားရှင်းများ အတွက်လည်း ဆင်တူ ယိုးမှား နည်းလမ်း များသာ ဖြစ်သည် [Ope17h]။

64 bit windows နှင့် Visual Studio အတွက် environment variable တစ်ခု အနေနှင့် OPENCV_DIR ကို သတ်မှတ် မည်။ ထို့အတွက် Command window ကို administrator အနေနှင့် run ရန် start menu ကို right click နှိပ်၍ command prompt (admin) ကို ရွေးပြီး လျင် အောက်ပါ စာရင်း ၁.၁၇ အတိုင်း ရိုက်ထည့် နိုင်သည်။

```
1 setx -m OPENCV_DIR C:\opencv\opencv330\build\x64\vc14
```

စာရင်း ၁.၁၇: OPENCV_DIR ကို သတ်မှတ် ခြင်း။

DLL များ သုံးသည့် အခါ bin folder ရှိသည့် နေရာကို system path တွင် ထည့်ရန် အတွက် **Rapid Environment Editor** အစရှိသည့် utility တို့ကိုသုံး၍သော်လည်းကောင်း၊ This PC(My Computer) > Properties > Advanced System Settings > Advanced Tab > Environment variables > Edit System Variables တွင် Path ကို ပုံ ၁.၁၃ တွင် ပြထား သကဲ့သို့ edit လုပ်ပြီး သော်လည်းကောင်း တနည်းနည်း သုံး၍ %OPENCV_DIR%\bin ကို system path အနေနှင့် ထည့်နိုင်သည်။



ပုံ ၁.၁၃: Environment variable အသစ်ထပ်ထည့်ခြင်း။

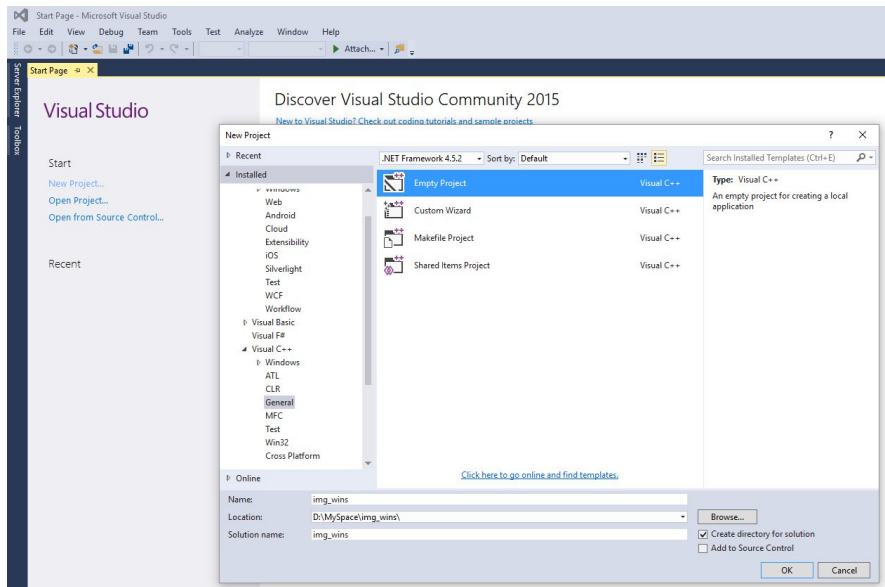
ထိုသို့ install လုပ်ခြင်းများ ပြီးစီး ပါက OpenCV ကို အသုံးပြု၍ နမူနာ တစ်ခု စမ်းသပ်ကြည့် မည်။ Visual Studio IDE ကို ဖွင့်၍ ပုံ ၁.၁၄ ကဲ့သို့ project အသစ် တစ်ခု ဖန်တီး မည်။ Installed > Visual C++ > General > Empty Project ကိုရွေးမည်။

ထို့နောက် ညာဖက် Solution Explorer အတွင်းရှိ Source Files ကို ညာဖက် ကလစ် နှိပ်ကာ Add > New Item... မှတစ်ဆင့် ပေါ်လာသော pop-up window တွင် Installed > Visual C++ > C++ File (.cpp) ကို ရွေးကာ main.cpp အမည်နှင့် ဖိုင် အလွတ် တခုကို ထည့်မည်။ အပေါ် Tool bar ရှိ Solution Platforms တွင် ပုံ ၁.၁၅ ကဲ့သို့ x86 အစား x64 သို့ ပြောင်းလိုက် မည်။

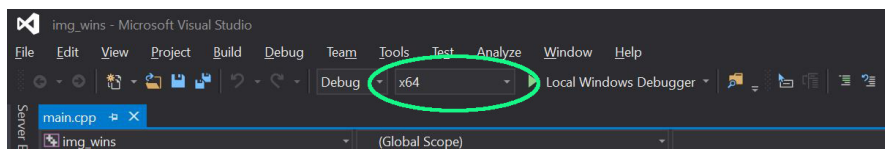
OpenCV နှင့် application များ ရေးရန် လိုအပ်သည့် အလုပ် နှစ်ခုရှိသည်။ OpenCV library များမှာ မည့်သည့် ပုံစံ ဖြစ်သည်ကို compiler အား ပြောရန်နှင့် OpenCV function များ၊ data structure များအား မည့်သည့် နေရာတွင် ယူရ မည်ကို linker အား ပြောရန် ဖြစ်သည်။ ထို့အတွက်

၁.၄. OPENCV ကို WINDOWS တွင် တပ်ဆင်ခြင်း

၂၁

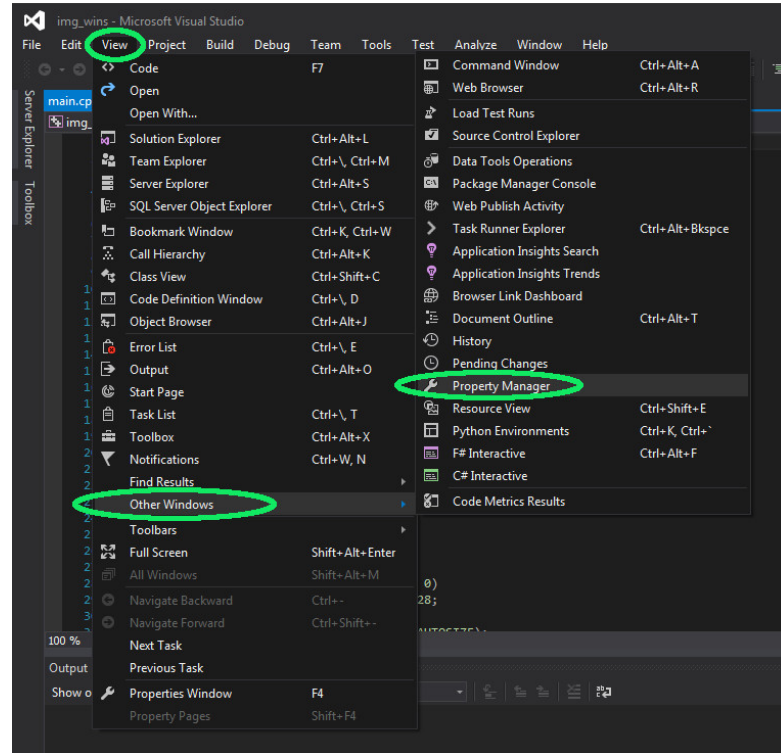


ပုံ ၁.၁၄: Visual Studio တွင် project အသစ် တစ်ခု ဖန်တီးခြင်း။



ပုံ ၁.၁၅: Platform ကို x64 အားရွေးခြင်း။

ပုံ ၁.၁၆ တွင် ပြထား သကဲ့သို့ အပေါ် menu bar ရှိ View > Other Windows > Property Manager ကိုသွား၍ ဖွင့်မည်။



ပုံ ၁.၁၆: Property Manager ကိုဖွင့်ခြင်း။

ပြီးနောက် ညာဖက် Property Manager တွင်လက်ရှိလုပ်မည့် configuration ဖြစ်သော Debug|x64 တွင် ညာဖက် ကလစ်နှိပ်၍ Add New Project Property Sheet... ကိုရွေးကာ 'OpenCV' အမည်ရှိ property sheet တစ်ခု ဖန်တီးမည် (ပုံ ၁.၁၇)။ Property sheet ကို တစ်ကြိမ် ဖန်တီးပြီးပါက နောက် Project များတွင် Add New အစား Add Existing Property Sheet ကို ရွေးကာ ပြန်လည် အသုံးပြု နိုင်သည်။

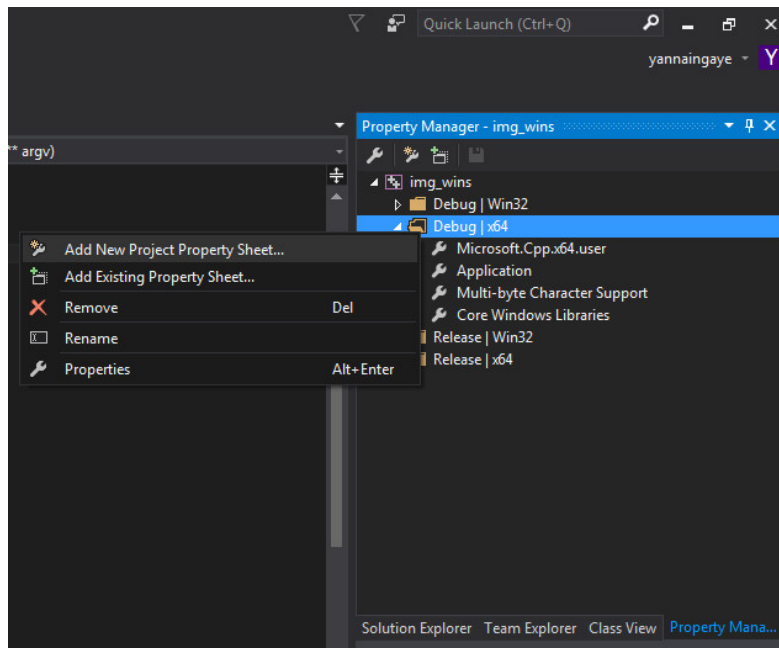
ထို့နောက် Property Manager အတွင်းရှိ OpenCV အမည် ပေါ်တွင် ညာဖက် ကလစ် နှိပ်ကာ Properties ကို ရွေးမည်။ ပေါ်လာသည့် Property Pages တွင် C/C++ > General > Additional Include Directories ၌

`$(OPENCV_DIR)\..\..\include`

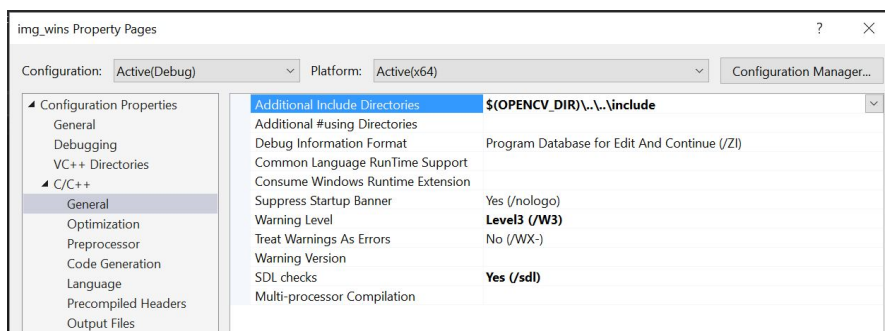
ဟုဖြည့်ပြီး Apply ကို နှိပ်မည် (ပုံ ၁.၁၈)။

၁.၄. OPENCV ကို WINDOWS တွင် တပ်ဆင်ခြင်း

၂၃



ပုံ ၁.၁၇: Property Sheet ထည့်ခြင်း။

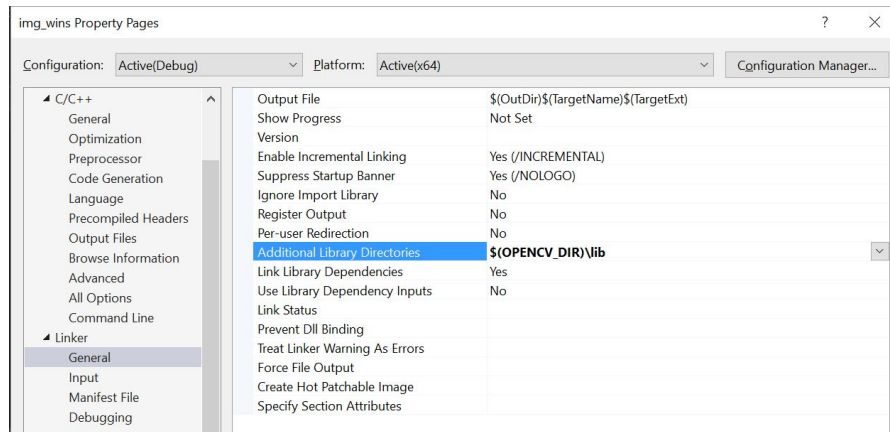


ပုံ ၁.၁၈: Include directory အားသတ်မှတ်ခြင်း။

တဖန် Linker အောက်ရှိ General > Additional Library Directories ဌှိ ပုံ ၁.၁၉ ကဲ့သို့

`$(OPENCV_DIR)\lib`

ဟု ညွှန်ပြီးသည့် အပြင် မည့်သည့် library ဟု သတ်မှတ်ရန် Linker အောက် Input > Additional Dependencies တွင် သုံးမည့် library modules များ၏ အမည်ကို ဖြည့်ရမည်။



ပုံ ၁.၁၉: Library directory အား သတ်မှတ်ခြင်း။

ဤနမူနာတွင် သုံးသည့် OpenCV 3.3.0 အတွက် C:/opencv/opencv330/build/ x64/vc14/lib တွင် opencv_world330.lib နှင့် opencv_world330d.lib ဟူ၍ library module နှစ်ခု တွေ့နိုင်သည်။ Debug mode အတွက် d နှင့် အဆုံးသတ် သည့် module သုံးနိုင်ပြီး၊ Release mode အတွက် ကျန်တစ်ခု ကို သုံးရမည်။ Module နာမည်ရှိ နံပါတ် 330 မှာ opencv version 3.3.0 အတွက် ဖြစ်ပြီး၊ သုံးသည့် version ပေါ် မူတည်၍ ပြောင်းမည်။

ယခု နမူနာ အတွက် အောက်ပါ library အမည်

`opencv_world330d.lib`

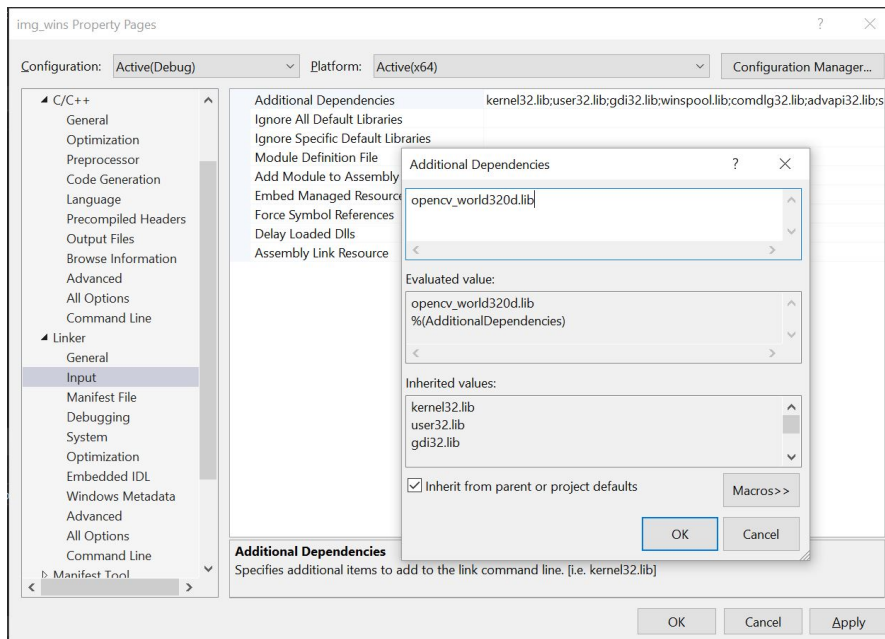
ကို ထည့်မည် (ပုံ ၁.၂၀)။

OpenCV တွင်သုံးသည့် fopen သည် Visual Studio 2017 ဌှိ deprecation ဖြစ်သွား သည်ကို disable လုပ်ရန် C/C++ > Preprocessor > Preprocessor Definitions တွင်

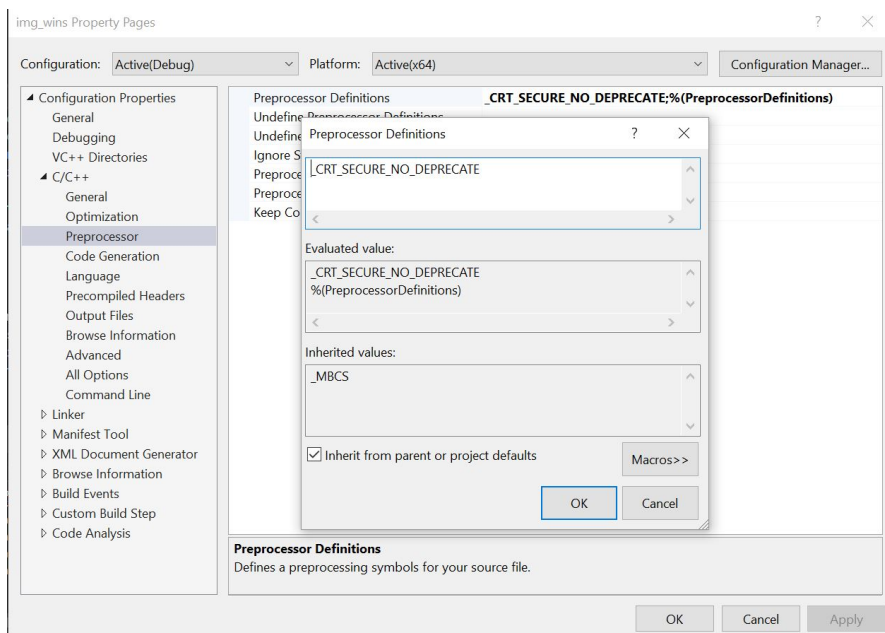
`_CRT_SECURE_NO_DEPRECATED`

ဟု ဖြည့်မည် (ပုံ ၁.၂၁)။

OpenCV စမ်းကြည့်ရန် နမူနာကို အောက်ပါ စာရင်း ၁.၁၈ အတိုင်း ရေးနိုင်သည်။



ပုံ ၁.၂၀: Library အမည်အား သတ်မှတ်ခြင်း။



ပုံ ၁.၂၁: Deprecation ကို disable လုပ်ခြင်း။


```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main(int argc, char** argv)
5 {
6     Mat image;
7     image = imread("C:/opencv/thiri.jpg",0);
8     if (!image.data) {
9         printf("No image data \n");
10        return -1;
11    }
12    namedWindow("Display Image", WINDOW_AUTOSIZE);
13    imshow("Display Image", image);
14
15    //Create a circular mask
16    int h = image.rows;
17    int w = image.cols;
18    Mat mask(h, w, CV_8UC1, Scalar(0, 0, 0));
19    circle(mask, Point(w / 2, h / 2), h / 3, 255, -1);
20
21    //Perform bitwise 'AND' to mask out
22    Mat im2;
23    bitwise_and(image, mask, im2);
24
25    //Manipulate pixels
26    for (int i=0; i < h; i++)
27        for (int j=0; j < w; j++)
28            if (mask.at<uchar>(i, j) == 0)
29                im2.at<uchar>(i, j) = 128;
30    namedWindow("Masked Image", WINDOW_AUTOSIZE);
31    imshow("Masked Image", im2);
32
33    waitKey(0);
34    return 0;
35 }

```

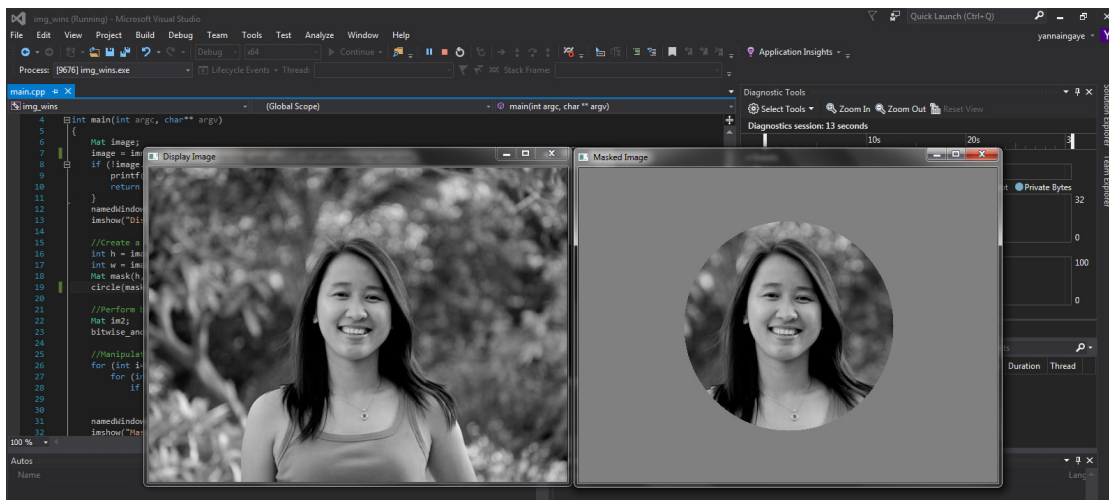
စာရင်း ၁.၁၈: main.cpp

ဤနေရာတွင် thiri ၏ ပုံကို C:/opencv/thiri.jpg တွင် ထားထားပြီး၊ သူရှိသည့် နေရာ full path ကို ရေးဖို့ လိုသည်။ Forward slash ကို လက်ခံသဖြင့် “/” ကို သုံးခြင်းက ရှင်းလင်းသည်။ Backslash ကို သုံးလိုပါက escape character အနေနှင့် “\\” သုံးရန် သတိပြုပါ။

ဤ ပရိုဂရမ်တွင် လမ်းကြောင်း ပေးထားသည့် ပုံကို ဖတ်သည့် အခါ ဒုတိယ argument မှာ 0 ဖြစ်သဖြင့် အဖြူ အမည်း အနေနှင့် ဖတ်သည်။ ထိုနောက် Mat ကို သုံးပြီး mask ဆိုသည့် ပုံ အလွတ်တခုကို အမည်းရောင် (တန်ဖိုး 0 အပြည့်) ဖြင့် ဖန်တီး လိုက်သည်။

ပြီးသည့် အခါ circle ကို သုံး၍ အဖြူရောင် စက်ဝိုင်း တခု mask ၏ အလယ် တွင် ဆွဲလိုက်သည်။ bitwise_and ကို သုံးလိုက်သည့်အခါ mask တွင် 0 ရှိသည့် pixel အားလုံး 0 (အမည်းရောင်) ဖြစ်သွား၍ mask တွင် အဖြူရောင် ရှိသည့် နေရာများ (အလယ် စက်ဝိုင်း) တွင်သာ မူရင်းပုံ ကျန်ခဲ့မည်။

ပြီးလျှင် for loop ကို သုံး၍ mask ၏ အမည်းရောင် pixel တစ်ခုခြင်းစီ ၏ တန်ဖိုးကို မီးခိုးရောင် (128) ဖြစ်အောင် ပြောင်းလိုက်သည်။ ရလဒ် သည့် ပုံကို imshow နှင့်ပြသည်။ ရလဒ်ကို အောက်ပါ ပုံ ၁.၂၂ တွင် မြင်နိုင်သည်။



ပုံ ၁.၂၂: Mask လုပ်၍ရသည့် ရလဒ်။

အကိုးအကားများ

- [Ope17g] OpenCV. Installation in Linux. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html.
- [Ope17h] OpenCV. Installation in Windows. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_install/windows_install.html.
- [Ope17i] OpenCV. Introduction to OpenCV. 2017. url: <http://docs.opencv.org/3.2.0/d1/dfb/intro.html>.
- [Ope17j] OpenCV. Open source computer vision. 2017. url: <http://opencv.org>.
- [Ope17l] OpenCV. Using OpenCV with gcc and CMake. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_gcc_cmake/linux_gcc_cmake.html.

အခန်း ၂

အခြေခံများ

ဤ အခန်းတွင် OpenCV အတွက် အခြေခံ များ ဖြစ်သော ပုံရိပ်များ အသုံးပြုပုံ၊ တည်ဆောက်ပုံ စသည် တို့ကို ဆွေးနွေးမည် [Ope16d]။ အခြေခံ structure များ ဖြစ်သော Mat, Size စသည် တို့နှင့် အခြေခံ လုပ်ဆောင် ချက်များ ဖြစ်သော ပုံရိပ် အရွယ် အစား ပြောင်းခြင်း၊ ပုံစံ များဆွဲခြင်း စသည် တို့ကို ပါ ဖော်ပြ မည် [Ope17b; Ope17d]။

၂.၁ ရေးဖတ်ခြင်းများ

OpenCV တွင် ပုံ တစ်ပုံ ကို ဖိုင်မှ ဖတ်လို ပါက imread ဖန်ရှင် ကို အောက်ပါ အတိုင်း သုံးနိုင် ကြောင်း ရှေ့တွင် ဖော်ပြပြီး ဖြစ်သည်။

```
Mat img = imread(filename);
```

ဖတ်သည့် ဖိုင်မှာ jpg အမျိုးအစား ဖြစ်ပါက ပြာစိမ်း၊နီ channel သုံးခု ပါသည့် ပုံ အနေနှင့် ဖတ်မည်။ အကယ်၍ အဖြူ အမည်း ပုံ အနေနှင့် ဖတ်လို ပါက အောက်ပါ အတိုင်း သတ်မှတ် နိုင်သည်။

```
Mat img = imread(filename,0);
```

ကာလာ ပုံ အတွက် သတ်မှတ်ချက် မှာ 1 ဖြစ်သည်။ ပုံကို ဖိုင် အနေ နှင့် သိမ်းလို ပါက imwrite ကို အောက်ပါ အတိုင်း သုံး၍ ရသည်။

```
imwrite(filename,img);
```

၂.၁.၁ Pixel များကိုသုံးခြင်း

ဥပမာ ပုံရိပ်မှာ ရှိတဲ့ အစက် တစ်ခု ခြင်းစီ ရဲ့ intensity ကို 8 bit unsigned char (8U) နဲ့ သုံးထား ပြီး channel တစ်ခု (C1) ပဲရှိ တဲ့ grayscale ပုံရိပ် (8UC1 အမျိုးအစား)ပုံရိပ် တစ်ခု ရဲ့ တန်း i ၊ တိုင် j မှာ ရှိတဲ့ pixel ကို အောက်ပါ အတိုင်း ဖတ်နိုင် သည်။

```
uchar v=img.at<uchar>(i,j);
```

အခြား နည်းလမ်း တစ်ခု အနေနှင့် x-y coordinates ပုံစံ သုံးပြီး အောက်ပါ အတိုင်း လည်း ဖတ်နိုင်သည်။

```
uchar v=img.at<uchar>(Point(j,i));
```

Channel ၃ ခု ပါသည့် BGR ကာလာ ပုံရိပ် ဆိုပါက အောက်ပါ ပုံစံ ဖြင့် သုံးနိုင်သည်။

```
Vec3b intensity = img.at<Vec3b>(i, j);
uchar blue = intensity.val[0];
uchar green = intensity.val[1];
uchar red = intensity.val[2];
```

အကယ်၍ ပုံရိပ်၏ တန်ဖိုး များမှာ ဒဿမ ကိန်းများ ဖြစ်နေ ပါက အောက်ပါ အတိုင်း ဖတ်၍ ရသည်။

```
Vec3f intensity = img.at<Vec3f>(i, j);
float blue = intensity.val[0];
float green = intensity.val[1];
float red = intensity.val[2];
```

အပြန်အလှန် အနေနှင့် Pixel တန်ဖိုး ကို ပြန်ရေး လိုပါက အောက်ပါ အတိုင်း ရေးနိုင် သည်။

```
img.at<uchar>(i, j) = 128;
```

အကိုးအကားများ

- [Ope16d] OpenCV. Operations with images. 2016. url: https://docs.opencv.org/2.4.13/doc/user_guide/ug_mat.html.
- [Ope17b] OpenCV. Basic Structures. 2017. url: https://docs.opencv.org/2.4/modules/core/doc/basic_structures.html.
- [Ope17d] OpenCV. Drawing Functions. 2017. url: https://docs.opencv.org/2.4/modules/core/doc/drawing_functions.html.

အခန်း ၃

တည်ရာနယ်တွင်ပြုစပ်ခြင်းများ

ဤ အပိုင်းတွင် 2D ပုံရိပ်များ၌ အမျိုးမျိုးသော linear နှင့် non-linear ဖယ်တာ (filter) များအား အသုံးပြုပုံ ကို ဖော်ပြပါမည် [Ope17f]။ ပုံရိပ်တစ်ခု၌ ရှိသော အစက် (pixel) တစ်ခုချင်းစီတွင် သူ့ နေရာ အနီးအနား ရှိသော အခြား အစက်များ၏ တန်ဖိုးများ နှင့် ပေါင်းစပ် ပြောင်းလဲခြင်းများ ပြုလုပ်ခြင်း ဖြစ်သဖြင့် တည်ရာနယ် ဖယ်တာ လုပ်ခြင်း (spatial filtering) ဟုပြောနိုင်သည်။

၃.၁ ချောမွတ်ရန်ပြုပြင်ခြင်းများ

၃.၁.၁ boxFilter

boxFilter သည် ပုံရိပ်ရှိ pixel ဟုခေါ်သည့် အစက် တစ်ခုချင်းအား အနီးရှိ အစက်တန်ဖိုး များနှင့် ပေါင်းစပ် ရောနှောပြီး ပုံရိပ်အား ဝါးစေသည်။ သူ၏ ပုံစံမှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

```
void boxFilter(InputArray src, OutputArray dst, int ddepth,
               Size ksize, Point anchor=Point(-1,-1), bool normalize=true,
               int borderType=BORDER_DEFAULT )
```

ထို parameters များမှာ

src: သည် input image ဟုခေါ်သော မူရင်း input အနေနှင့် ပေးရမည့် ပုံဖြစ်သည်။

dst: သည် output အနေနှင့် ထွက်လာမည့် image ဖြစ်ပြီး src နှင့် အရွယ်အစား၊ အမျိုးအစား တူမည်။

ddepth: သည် ထွက်လာမည့် output image ၏ depth ဖြစ်သည်။ (-1 ဆိုပါက src ၏ depth နှင့် တူမည်။)

ksize: သည် blurring kernel size ဖြစ်သည်။

anchor: သည် anchor point ဟုခေါ်သော kernel ၏ အခြေ နေရာဖြစ်သည်။ Default တန်ဖိုးဖြစ်သော Point(-1,-1) ဆိုပါက အလယ်တည့်တည့် တွင်ရှိမည်။

normalize: သည် kernel ကို သူ၏ area နှင့် ပြန်စားပြီး normalize လုပ်မလုပ် သတ်မှတ်သည်။

borderType: သည် border mode ဟုခေါ်သော ပုံအနားသတ်ရှိ အစက်များ တန်ဖိုးကို တွက်သည့် အခါ မည်သည့် extrapolate နည်းသုံးမည် ကိုသတ်မှတ်သည်။

ဤ boxFilter function ၏ smooth လုပ်သော kernel မှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

$$K = \alpha \begin{bmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{bmatrix} \quad (၃.၁)$$

ဤတွင်

$$\alpha = \begin{cases} \frac{1}{ksize.width \times ksize.height}, & \text{when normalize=true} \\ 1, & \text{otherwise} \end{cases} \quad (၃.၂)$$

ဖြစ်သည်။

၃.၁.၂ blur

Blur သည် normalized box filter ကို သုံး၍ ပုံကို ဝါးစေသည်။ boxFilter ကို ddepth တွင် src ၏ depth အတိုင်း ထားပြီး၊ normalize တွင် true ပေးသည်နှင့် ထပ်တူညီသည်။ Blur filter ၏ ပုံစံမှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

```
void blur(InputArray src, OutputArray dst, Size ksize,
          Point anchor=Point(-1,-1),
          int borderType=BORDER_DEFAULT )
```

၃.၁.၃ GaussianBlur

GaussianBlur သည် Gaussian filter ကို သုံး၍ ပုံကို ဝါးစေသည်။ ညက်ညောစွာ ဝါးစေသဖြင့် ကြမ်းနေသော ပုံများအား ချောမွတ်စေခြင်း၊ noise များအား လျော့ပါးစေခြင်း တို့တွင် အသုံးများသည်။ ဤ function ၏ ပုံစံမှာ

```
void GaussianBlur(InputArray src, OutputArray dst, Size ksize,
    double sigmaX, double sigmaY=0,
    int borderType=BORDER_DEFAULT )
```

ဖြစ်သည်။ Parameters များမှာ အောက်ပါအတိုင်းဖြစ်သည်။

src: သည် input image ဟုခေါ်သော မူရင်း input အနေနှင့် ပေးရမည့် ပုံဖြစ်သည်။ ထိုပုံတွင် channel အရေအတွက် ကြိုက်သလောက် ရှိနိုင်ပြီး သီးသန့်စီ process လုပ်မည် ဖြစ်သည်။ သို့သော် depth အနေနှင့် CV_8U, CV_16U, CV_16S, CV_32F, CV_64F များမှ တစ်မျိုးမျိုးသာ သုံး၍ရမည်။

dst: သည် output အနေနှင့် ထွက်လာမည့် image ဖြစ်ပြီး src နှင့် အရွယ်အစား၊ အမျိုးအစား တူမည်။

ksize: သည် Gaussian kernel size ဖြစ်သည်။ ksize.width နှင့် ksize.height တို့ မတူ၍ ရသော်လည်း မကဏန်း တန်ဖိုးများသာ ဖြစ်ရမည်။ သို့မဟုတ်ပါက သုည ဖြစ်နိုင်ပြီး၊ ထိုတန်ဖိုးများအား sigma မှတွက်ယူ လိမ့်မည်။

sigmaX: သည် Gaussian kernel ၏ X ဝင်ရိုး တလျှောက် standard deviation ဖြစ်သည်။

sigmaY: သည် Gaussian kernel ၏ Y ဝင်ရိုး တလျှောက် standard deviation ဖြစ်သည်။ သုည ဖြစ်ပါက sigmaX အတိုင်း ယူမည်။ နှစ်ခုလုံး သုည ဖြစ်နေပါက ksize.width နှင့် ksize.height မှတွက်ယူ လိမ့်မည်။

borderType: သည် border mode ဟုခေါ်သော ပုံအနားသတ်ရှိ အစက်များ တန်ဖိုးကို တွက်သည့် အခါ မည်သည့် extrapolate နည်းသုံးမည် ကိုသတ်မှတ်သည်။

၃.၁.၄ medianBlur

Median filter ကို သုံးပြီး ပုံကို ဝါးစေလိုပါက medianBlur အား သုံးနိုင်သည်။ Gaussian filter နှင့် ယှဉ်လျှင် ပုံ၏ ပြတ်သားမှုကို ထိခိုက်မှု ပိုနည်းသည်။ Salt and Pepper noise များအား ဖယ်ရှားရာတွင် အသုံးတည့်သည်။ ဤ filter ၏ပုံစံမှာ

```
void medianBlur(InputArray src, OutputArray dst, int ksize)
```

ဖြစ်သည်။ Parameters များမှာ အောက်ပါ အတိုင်းဖြစ်သည်။

src: သည် input image ဟုခေါ်သော မူရင်း input အနေနှင့် ပေးရမည့် ပုံဖြစ်သည်။ ထိုပုံတွင် channel အရေအတွက် ၁ ခု၊ ၃ ခု၊ သို့မဟုတ် ၄ ခု ရှိနိုင်သည်။ ksize ၃ သို့မဟုတ် ၅ ဆိုပါက depth တွင် CV_8U, CV_16U, CV_32F တို့မှ တစ်မျိုးမျိုး သာ သုံး၍ ရမည်။ ပိုကြီးသော ksize များအတွက် CV_8U တမျိုးသာ ရမည်။

dst: သည် output အနေနှင့် ထွက်လာမည့် image ဖြစ်ပြီး src နှင့် အရွယ်အစား၊ အမျိုးအစား တူမည်။

ksize: သည် aperture linear size ဖြစ်ပြီး ၁ ထက်ကြီးသော မ ဂဏန်း တန်ဖိုး များသာ ဖြစ်ရမည်။ ဥပမာ ၃, ၅, ၇, ...

ဖော်ပြခဲ့သော smoothing filter များအား အောက်ပါ စာရင်း ၃.၁ တွင် ပြထားသော ပရိုဂရမ်ကို သုံး၍ စမ်းကြည့် နိုင်သည်။

```
1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main()
5 {
6     //Read original image
7     Mat image;
8     image = imread("C:/opencv/star.jpg",0);
9     if (!image.data) {
10         printf("No image data \n");
11         return -1;
12     }
13     namedWindow("Original Image", WINDOW_AUTOSIZE);
14     imshow("Original Image", image);
15
16     //Box filter
17     Mat im_box;
18     boxFilter(image, im_box, -1, Size(7, 7), Point(-1, -1), true, BORDER_DEFAULT)
19         ;
20     namedWindow("Box filter", WINDOW_AUTOSIZE);
```

```

20 imshow("Box filter", im_box);
21
22 //Gaussian filter
23 Mat im_gaussian;
24 GaussianBlur(image, im_gaussian, Size(7, 7), 0, 0);
25 namedWindow("Gaussian filter", WINDOW_AUTOSIZE);
26 imshow("Gaussian filter", im_gaussian);
27
28 //Median filter
29 Mat im_median;
30 medianBlur(image, im_median, 7);
31 namedWindow("Median filter", WINDOW_AUTOSIZE);
32 imshow("Median filter", im_median);
33 waitKey(0);
34 return 0;
35 }

```

စာရင်း ၃.၁: Smoothing filter များ အတွက် နမူနာ ပရိုဂရမ်။

ပရိုဂရမ်၏ ရလဒ်များအား ပုံ ၃.၁ တွင်နှိုင်းယှဉ် ဖော်ပြထားသည်။

၃.၂ filter2D

ပုံတစ်ခုကို kernel တစ်ခုနှင့် convolve လုပ်ပေးသည် [Ope14a]။ မိမိစိတ်ကြိုက် custom kernel အသုံးပြု၍ ရသည်မှာ အားသာချက်ဖြစ်သည်။ ဤ filter ၍ပုံစံမှာ

```

void filter2D(InputArray src, OutputArray dst,
int ddepth, InputArray kernel,
Point anchor=Point(-1,-1), double delta=0,
int borderType=BORDER_DEFAULT )

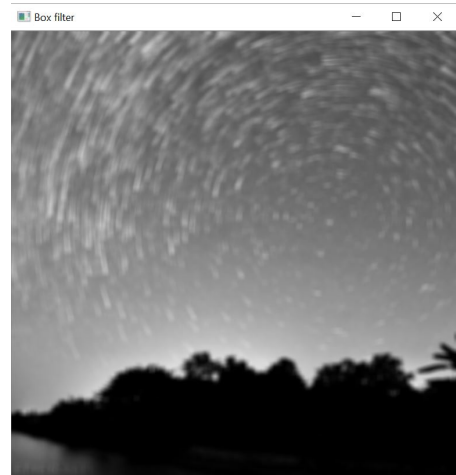
```

ဖြစ်သည်။ ထိုတွင် သုံးရမည့် parameters များမှာ အောက်ပါအတိုင်း ဖြစ်သည်။

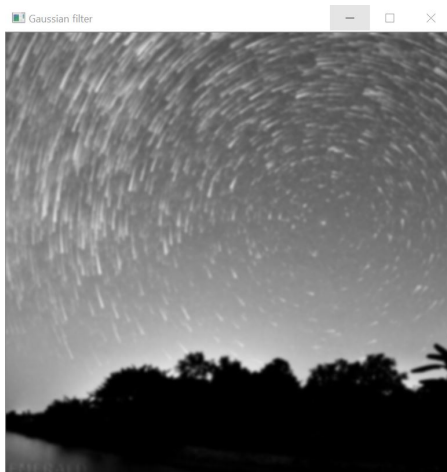
src: သည် input image ဟုခေါ်သော မူရင်း input အနေနှင့် ပေးရမည့် ပုံဖြစ်သည်။



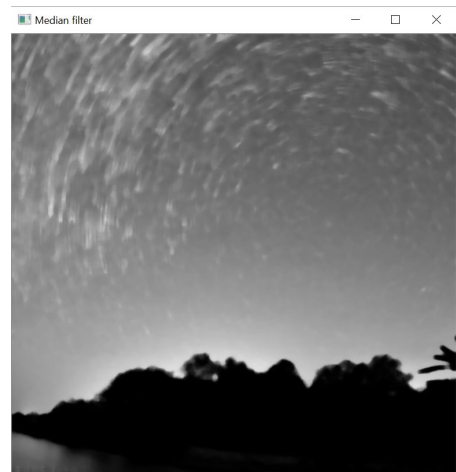
(a) မူရင်းပုံ။



(b) box filter ၏ ရလဒ်။



(c) Gaussian filter ၏ ရလဒ်။



(d) median filter ၏ ရလဒ်။

ပုံ ၃.၁: Smoothing filter များကို နှိုင်းယှဉ်ဖော်ပြခြင်း။

dst: သည် output အနေနှင့် ထွက်လာမည့် image ဖြစ်ပြီး src နှင့် အရွယ်အစား၊ အမျိုးအစား တူမည်။

ddepth: သည် ထွက်လာမည့် output image ၏ depth ဖြစ်သည်။ (-1 ဆိုပါက src ၏ depth နှင့် တူမည်။) အောက်တွင် ပြထားသော အတွဲများ ရသည်။

- `src.depth() = CV_8U, ddepth = -1/CV_16S/CV_32F/CV_64F`
- `src.depth() = CV_16U/CV_16S, ddepth = -1/CV_32F/CV_64F`
- `src.depth() = CV_32F, ddepth = -1/CV_32F/CV_64F`
- `src.depth() = CV_64F, ddepth = -1/CV_64F`

kernel: သည် convolution kernel ဖြစ်သည်။ single-channel floating point matrix ဖြစ်သည်။ မတူသော channel များအား မတူသော kernel သုံးချင်ပါက ပုံကို `split()` သုံးပြီး သတ်သတ်စီ ခွဲသုံးရမည် ဖြစ်သည်။

anchor: သည် anchor point ဟုခေါ်သော kernel ၏ အခြေ နေရာဖြစ်သည်။ Default တန်ဖိုးဖြစ်သော `Point(-1,-1)` ဆိုပါက အလယ်တည့်တည့် တွင်ရှိမည်။

delta: filter လုပ်၍ရသည့် pixel တန်ဖိုးတွင် ပေါင်းထည့် ပေးမည့် optional value ဖြစ်သည်။

borderType: သည် border mode ဟုခေါ်သော ပုံအနားသတ်ရှိ အစက်များ တန်ဖိုးကို တွက်သည့် အခါ မည်သည့် extrapolate နည်းသုံးမည် ကိုသတ်မှတ်သည်။

အမှန်တွင် ဤ function တွက်ပေးသည်မှာ convolution မဟုတ်ပဲ correlation သာ ဖြစ်သည်။

$$dst(x, y) = \sum_{i=0}^{(kernel.cols-1)} \sum_{j=0}^{(kernel.rows-1)} kernel(i, j) * src(x + i - anchor.x, y + j - anchor.y) \quad (၃.၃)$$

Convolution အစစ် သုံးလိုပါက kernel ကို `flip()` ဆိုသည့် လုပ်ဆောင်ချက် အရင် သုံးပေး နိုင်သည်။ `filter2D` အတွက် နမူနာ ပရိုဂရမ်ကို အောက်ပါ စာရင်း ၃.၂ တွင် ဖော်ပြ ထားသည်။

```
1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main()
```

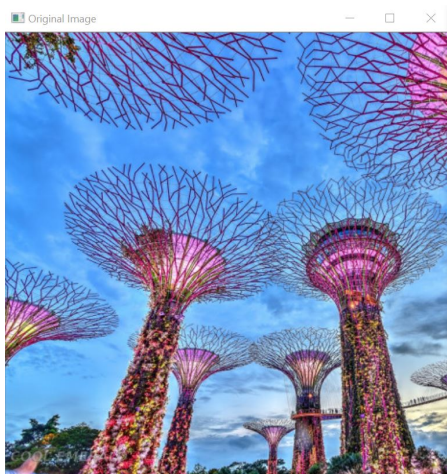
```

5 {
6 //Read original image
7 Mat image;
8 image = imread("C:/opencv/grove.jpg",1);
9 //image = imread("/home/yan/grove.jpg",1);
10 if (!image.data) {
11 printf("No image data \n");
12 return -1;
13 }
14 namedWindow("Original Image", WINDOW_AUTOSIZE);
15 imshow("Original Image", image);
16
17 //create kernel
18 Mat kernel;
19 int kernel_size = 9;
20 kernel = Mat::ones(kernel_size, kernel_size, CV_32F) / (float)(kernel_size*
    kernel_size);
21
22 //filter2D filter
23 Mat im_filter;
24 filter2D(image, im_filter, -1, kernel, Point(-1, -1), 0, BORDER_DEFAULT);
25 namedWindow("filter2D", WINDOW_AUTOSIZE);
26 imshow("filter2D", im_filter);
27 waitKey(0);
28 return 0;
29 }

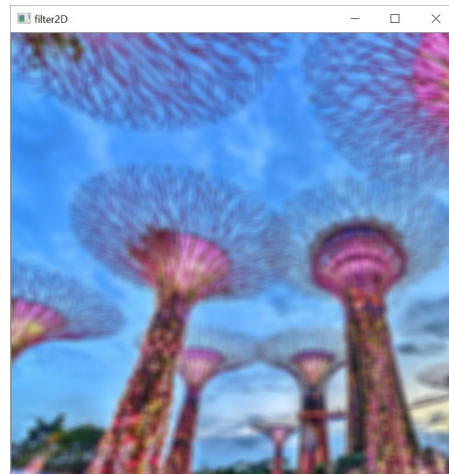
```

စာရင်း ၃.၂: filter2D အတွက် နမူနာ ပရိုဂရမ်။

ပရိုဂရမ်၏ ရလဒ်အား အောက်ပါ ပုံ ၃.၂ တွင်မူရင်းနှင့်နှိုင်းယှဉ် ဖော်ပြထားသည်။



(a) မူရင်းပုံ။



(b) filter2D ၏ ရလဒ်။

ပုံ ၃.၂: filter2D ၏ရလဒ်များကို နှိုင်းယှဉ်ဖော်ပြခြင်း။

၃.၃ Thresholding

Thresholding သည် အခြေခံ အကျဆုံး၊ အရိုးရှင်းဆုံး segmentation လုပ်ဆောင်သော နည်းဖြစ်သည်။ ပုံတစ်ပုံ အတွက် ပိုင်းချမည့် threshold တစ်ခု သတ်မှတ်ကာ ထိုထက် ပိုများ၊ ပိုတောက်ပပါက အဖြူ၊ နည်းပါက အမည်း စသည်ဖြင့် ပိုင်းချလိုက်ခြင်းဖြစ်သည်။ ထိုအတွက် အသုံးပြုနိုင်သော threshold function ၏ ပုံစံကို အောက်တွင် ဖော်ပြထားသည် [Ope16a]။

```
double cv::threshold (InputArray src,
OutputArray dst,
double thresh,
double maxval,
int type
)
```

Parameters များမှာ အောက်ပါအတိုင်းဖြစ်သည်။

src: သည် input array ဖြစ်ပြီး single-channel သို့ 8-bit သို့ 32-bit floating point တစ်ခုခု ဖြစ်နိုင်သည်။

dst: သည် output array ဖြစ်ပြီး src နှင့် အရွယ်တူ၊ အမျိုးအစားတူ ဖြစ်သည်။

thresh: သည် threshold တန်ဖိုး ဖြစ်သည်။

maxval: သည် maximum တန်ဖိုး ဖြစ်သည်။ သတ်မှတ်ချက် ပြေလည်ပါက ထားရှိမည့် pixel ၏တန်ဖိုး ဖြစ်သည်။

type: သည် threshold ပိုင်းချမည့် လုပ်ဆောင်မှု အမျိုးအစား ဖြစ်ပြီး အောက်ပါတို့မှ တစ်ခုခု ဖြစ်နိုင်သည် (နောက်ဆက်တွဲ ၁.၂ ကိုကြည့်ပါ)။

- THRESH_BINARY
- THRESH_BINARY_INV
- THRESH_TRUNC
- THRESH_TOZERO
- THRESH_TOZERO_INV
- THRESH_MASK
- THRESH_OTSU
- THRESH_TRIANGLE

Threshold လုပ်မည့် ပုံတွင် ကြိမ်နှုန်းနိမ့် (low frequency) အလင်းရောင် အပြောင်း အလဲများ ပါရှိ ပါက အထက်ပါ global thresholding နည်း မသုံးခင် ကြိမ်နှုန်းနိမ့် အပိုင်း များအား ဖယ်ထုတ်ရန် လိုမည်။ သို့မဟုတ်ပါက အောက်တွင် ဖော်ပြထားသည့် adaptive thresholding ကိုသုံးနိုင် သည် [Ope16c]။

```
void cv::adaptiveThreshold ( InputArray  src,
OutputArray  dst,
double  maxValue,
int  adaptiveMethod,
int  thresholdType,
int  blockSize,
double  C
)
```

Parameters များမှာ အောက်ပါအတိုင်းဖြစ်သည် [Ope17a]။

src: သည် 8-bit single-channel image ဖြစ်နိုင်သည်။

dst: သည် destination image ဖြစ်ပြီး src နှင့် အရွယ်တူ၊ အမျိုးအစားတူ ဖြစ်သည်။

maxval: သည် သတ်မှတ်ချက်ပြေလည်ပါက ထားရှိမည့် pixel ၏တန်ဖိုး ဖြစ်သည်။

adaptiveMethod: သည် သုံးမည့် Adaptive thresholding algorithm ဖြစ်ပြီး အောက် ပါတို့မှ တစ်ခုခု ဖြစ်နိုင်သည်။

- ADAPTIVE_THRESH_MEAN_C
- ADAPTIVE_THRESH_GAUSSIAN_C

thresholdType: သည် threshold ပိုင်းချမည့် လုပ်ဆောင်မှု အမျိုးအစား ဖြစ်ပြီး အောက် ပါတို့မှ တစ်ခုခု ဖြစ်နိုင်သည်။

- THRESH_BINARY
- THRESH_BINARY_INV

blockSize: သည် pixel တစ်ခု၏ threshold ကိုရှာသည့်အခါ သုံးမည့် အနီးအနား ဝန်းကျင် အရွယ်အစား ဖြစ်ပြီး 3, 5, 7, စသည့် တန်ဖိုး တစ်ခုခု ဖြစ်သည်။

C: သည် mean သို့ weighted mean မှ နုတ်ပေးမည့် ကိန်းသေ ဖြစ်သည်။ ပုံမှန်အားဖြင့် အပေါင်းကိန်းဖြစ်သော်လည်း သုည သို့ အနုတ် ကိန်းများလည်း ဖြစ်နိုင်သည်။

Thresholding ပြုလုပ်သည့် နမူနာ ဂရိုဂရမ် တစ်ခု ကို အောက်ပါ စာရင်း ၃.၃ တွင် တွေ့နိုင်သည်။

```

1 #include "opencv2/imgproc.hpp"
2 #include "opencv2/imgcodecs.hpp"
3 #include "opencv2/highgui.hpp"
4 using namespace cv;
5
6 Mat src, src_gray, dst;
7 int threshold_value = 128;
8 int threshold_type = 0;
9 int const max_BINARY_value = 255;

```

```

10 void Threshold_changed(int, void*);
11 const char* window_name = "Thresholding";
12
13 int main()
14 {
15     //src = imread("/home/yan/ka.jpg", IMREAD_COLOR);
16     src = imread("c:/opencv/ka.jpg", IMREAD_COLOR);
17     if (src.empty()) { return -1;}
18
19     namedWindow("Original", WINDOW_AUTOSIZE);
20     imshow("Original",src);
21
22     cvtColor(src, src_gray, COLOR_BGR2GRAY); // Convert the image to Gray
23     namedWindow(window_name,WINDOW_NORMAL); // Create a window to display results
24
25     // Create Trackbar to choose type of Threshold
26     //Type: 0= Binary, 1= Binary Inverted,
27     // 2= Truncate, 3= To Zero, 4= To Zero Inverted,
28     // 5= Adaptive threshold using ADAPTIVE_THRESH_GAUSSIAN_C
29     int const max_type = 5;
30     createTrackbar("Type",
31 window_name, //window for trackbar
32 &threshold_type, // value to be updated according to trackbar
33 max_type, //range
34 Threshold_changed); //function to be called everytime trackbar is changed
35
36     // Create Trackbar to choose Threshold value
37     int const max_value = 255;
38     createTrackbar("Value",
39 window_name, &threshold_value,
40 max_value, Threshold_changed);
41
42     Threshold_changed(0, 0); // Call the function to initialize
43
44     //loop until 'Esc' key is pressed
45     for (;;) {

```

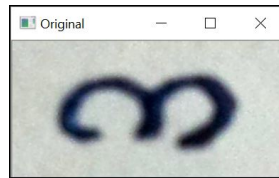
```

46 char c = (char)waitKey(20);
47 if (c == 27) { break;}
48 }
49
50 //save the resulting binary image
51 imwrite("c:/opencv/ka_binary.jpg",dst);
52 return 0;
53 }
54 void Threshold_changed(int, void*)
55 {
56 if (threshold_type == 5) {
57 adaptiveThreshold(src_gray, dst, max_BINARY_value,
58 ADAPTIVE_THRESH_GAUSSIAN_C, THRESH_BINARY, 11, 2);
59 }
60 else {
61 threshold(src_gray, dst, threshold_value,
62 max_BINARY_value, threshold_type);
63 }
64 imshow(window_name, dst);
65 }

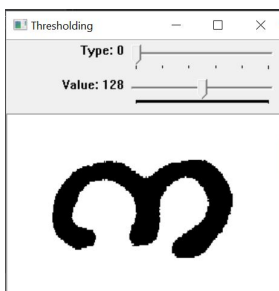
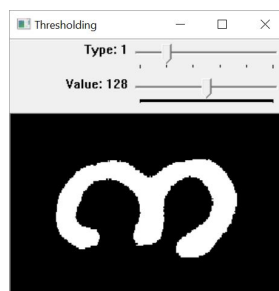
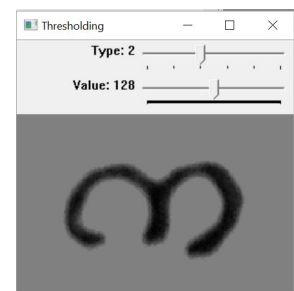
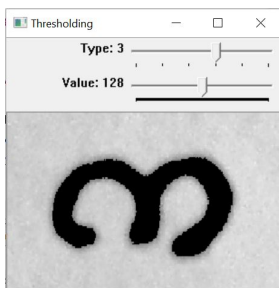
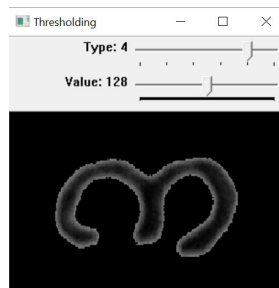
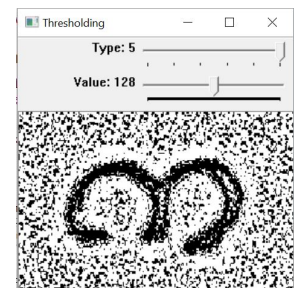
```

စာရင်း ၃.၃: Thresholding နမူနာ ပရိုဂရမ်။

ပရိုဂရမ်၏ ရလဒ်အား အောက်ပါ ပုံ ၃.၃ တွင်မူရင်းနှင့်နှိုင်းယှဉ် ဖော်ပြထားသည်။



(a) မူရင်းပုံ။

(b) Binary thresholding
၏ ရလဒ်။(c) Inverse binary ၏
ရလဒ်။(d) THRESH_TRUNC ၏
ရလဒ်။(e) THRESH_TO_ZERO ၏
ရလဒ်။(f) THRESH_TO-
ZERO_INV ၏ ရလဒ်။(g) adaptive Threshold
၏ ရလဒ်။

ပုံ ၃.၃: Thresholding ရလဒ်များကို နှိုင်းယှဉ်ဖော်ပြခြင်း။

၃.၄ ပုံလုံးဖော်လုပ်ဆောင်ချက်များ

အခြေခံတည်ဆောက်ပုံစံ (structuring element) တစ်ခု၏ သဏ္ဌာန် (shape) ကို အခြေခံ၍ ပုံကို ပြုပြင်ခြင်းအား ပုံလုံးဖော်လုပ်ဆောင်ချက်များ (morphological operation) ဟုခေါ်မည်။ Noise အစက်အပြောက်များအား ဖယ်ရှားခြင်း၊ ထိဆက်နေသော အပိုင်း များအား ခွဲခြားခြင်း၊ အစရှိသော နေရာများတွင် အသုံးပြု ကြသည် [Ope17f]။

Structuring element ကို `getStructuringElement()` ကို အသုံးပြု၍ ရယူနိုင်သည်။

```
Mat cv::getStructuringElement (int shape,
Size ksize, Point anchor = Point(-1,-1) )
```

Parameters များမှာ အောက်ပါအတိုင်းဖြစ်သည်။

shape: သည် MORPH_RECT, MORPH_CROSS, MORPH_ELLIPSE တို့မှ တစ်ခုခု ဖြစ်နိုင်သည်။

ksize: သည် structuring element ၏ အရွယ်ဖြစ်သည်။

anchor: သည် anchor point ဟုခေါ်သော element ၏ အခြေ နေရာဖြစ်သည်။ Default တန်ဖိုးဖြစ်သော `Point(-1,-1)` ဆိုပါက အလယ်တည့်တည့် တွင်ရှိမည်။

၃.၄.၁ တိုက်စားခြင်း (erosion)

ပုံသဏ္ဌာန်တစ်ခု ၏ အဖြူရောင် အပိုင်းကို ကျုံ့သွား၊ ပါးသွားအောင် စားပေးသည်။

```
void erode(InputArray src, OutputArray dst, InputArray kernel,
Point anchor=Point(-1,-1), int iterations=1,
int borderType=BORDER_CONSTANT,
const Scalar& borderValue=morphologyDefaultBorderValue() )
```

၃.၄.၂ ဖောင်းပွခြင်း (dilation)

ပုံသဏ္ဌာန်တစ်ခု ၏ အဖြူရောင် အပိုင်းကို ပွလာ၊ ပြန့်ကားလာ အောင် ကြီးစေသည်။

```
void dilate(InputArray src, OutputArray dst, InputArray kernel,
```

```
Point anchor=Point(-1,-1), int iterations=1,
int borderType=BORDER_CONSTANT,
const Scalar& borderValue=morphologyDefaultBorderValue() )
```

၃.၄.၃ morphologyEx

အခြားပို၍ ရှုပ်ထွေးသော morphological operation များအား သုံးလိုပါက morphologyEx ကို သုံးနိုင်သည်။

```
void morphologyEx(InputArray src, OutputArray dst, int op,
InputArray kernel, Point anchor=Point(-1,-1),
int iterations=1, int borderType=BORDER_CONSTANT,
const Scalar& borderValue=morphologyDefaultBorderValue() )
```

ထိုတွင် op မှာ morphological operation အမျိုးအစား ဖြစ်ပြီး အောက်ပါတို့မှ တစ်ခု ဖြစ်နိုင်သည်။

- MORPH_OPEN - an opening operation
- MORPH_CLOSE - a closing operation
- MORPH_GRADIENT - a morphological gradient
- MORPH_TOPHAT - “top hat”
- MORPH_BLACKHAT - “black hat”
- MORPH_HITMISS - “hit and miss”

နမူနာ ပရိုဂရမ် တစ်ခုကို အောက်ပါ စာရင်း ၃.၄ တွင် ဖော်ပြ ထားသည်။

```
1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 using namespace cv;
4 int main()
5 {
6     //Read original image
7     Mat image;
```

```
8 image = imread("C:/opencv/ka_ori.jpg", 0);
9 //image = imread("/home/yan/ka_ori.jpg", 0);
10 if (!image.data) {
11     printf("No image data \n");
12     return -1;
13 }
14 namedWindow("Original Image", WINDOW_AUTOSIZE);
15 imshow("Original Image", image);
16
17 Mat e_struct = getStructuringElement(MORPH_RECT, Size(7, 7), Point(3, 3));
18
19 //erosion
20 Mat im_e;
21 erode(image, im_e, e_struct);
22 namedWindow("Erosion", WINDOW_AUTOSIZE);
23 imshow("Erosion", im_e);
24
25 //dilation
26 Mat im_d;
27 dilate(image, im_d, e_struct);
28 namedWindow("Dilation", WINDOW_AUTOSIZE);
29 imshow("Dilation", im_d);
30
31 //opening
32 Mat im_o;
33 morphologyEx(image, im_o, MORPH_OPEN, e_struct);
34 namedWindow("Opening", WINDOW_AUTOSIZE);
35 imshow("Opening", im_o);
36
37 //closing
38 Mat im_c;
39 morphologyEx(image, im_c, MORPH_CLOSE, e_struct);
40 namedWindow("Closing", WINDOW_AUTOSIZE);
41 imshow("Closing", im_c);
42
43 //opening and closing to remove noise
```



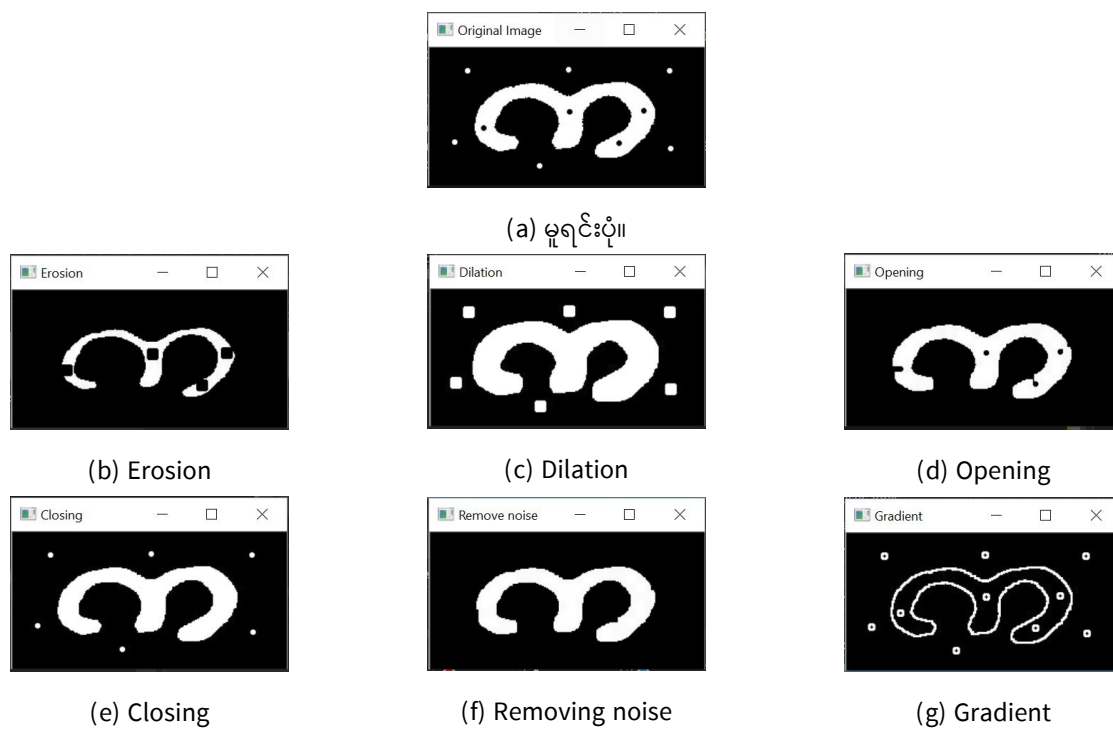
```

44 Mat im_r;
45 morphologyEx(image, im_o, MORPH_OPEN, e_struct);
46 morphologyEx(im_o, im_r, MORPH_CLOSE, e_struct);
47 namedWindow("Remove noise", WINDOW_AUTOSIZE);
48 imshow("Remove noise", im_r);
49
50 //gradient
51 Mat g_struct = getStructuringElement(MORPH_RECT, Size(3, 3), Point(1, 1));
52 Mat im_g;
53 morphologyEx(image, im_g, MORPH_GRADIENT, g_struct);
54 namedWindow("Gradient", WINDOW_AUTOSIZE);
55 imshow("Gradient", im_g);
56
57 waitKey(0);
58 return 0;
59 }

```

စာရင်း ၃.၄: ပုံသဏ္ဌာန်ဆိုင်ရာပြောင်းလဲခြင်းများ အတွက် နမူနာ ပရိုဂရမ်။

ပရိုဂရမ်၏ ရလဒ်အား အောက်ပါ ပုံ ၃.၄ တွင်မူရင်းနှင့်နှိုင်းယှဉ် ဖော်ပြထားသည်။



ပုံ ၃.၄: ပုံလုံးဖော်လုပ်ဆောင်ချက်များ၏ ရလဒ်များကို နှိုင်းယှဉ်ဖော်ပြခြင်း။ Opening တွင် အဖြူရောင် အစက်များ ပျောက်သွား သည်။ Closing တွင် အမည်း ရောင် အစက်များ ပျောက်သွား သည်။

၃.၅ အရောင်နယ်ပြောင်းလဲခြင်းများ

အရောင်နယ် (Color space) ပြောင်းခြင်းနှင့် ပတ်သက်၍ RGB မှ HSV သို့ပြောင်းသုံးသည့် နမူနာတစ်ခု ကို ဆွေးနွေးပါမည် [Gar+15; CE14]။ HSV သည် အရောင်ကို အခြေခံသော ပြုပြင်ခြင်း၊ ရှာဖွေခြင်း၊ မှတ်မိခြင်း များအတွက်များစွာ အဆင်ပြေ အသုံးဝင်သည်။ ပုံတစ်ပုံရှိ အပြာ ရောင် အပိုင်းများကို ရွေးချယ်၍ အစိမ်းရောင် ပြောင်းပေးသည့် နမူနာ ဂရိုဂရမ်ကို အောက်ပါ စာရင်း ၃.၅ တွင်တွေ့နိုင်သည်။

```

1 //File: color.cpp
2 //Description: Selective Color Modification
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 #include <stdio.h>
8 #include <opencv2/opencv.hpp>
9 using namespace cv;
10 int main(int argc, char** argv)
11 {
12     Mat image;
13     image = imread("C:/opencv/lane.jpg", 1);
14     if (!image.data) {
15         printf("No image data \n");
16         return -1;
17     }
18     namedWindow("Original Image", WINDOW_AUTOSIZE);
19     imshow("Original Image", image);
20
21     //Convert to hsv
22     Mat hsv;
23     cvtColor(image, hsv, COLOR_BGR2HSV);
24
25     //select pixels
26     //the range of H channel is 0-179.
27     //blue at 240 deg corresponds to 120 in H channel

```

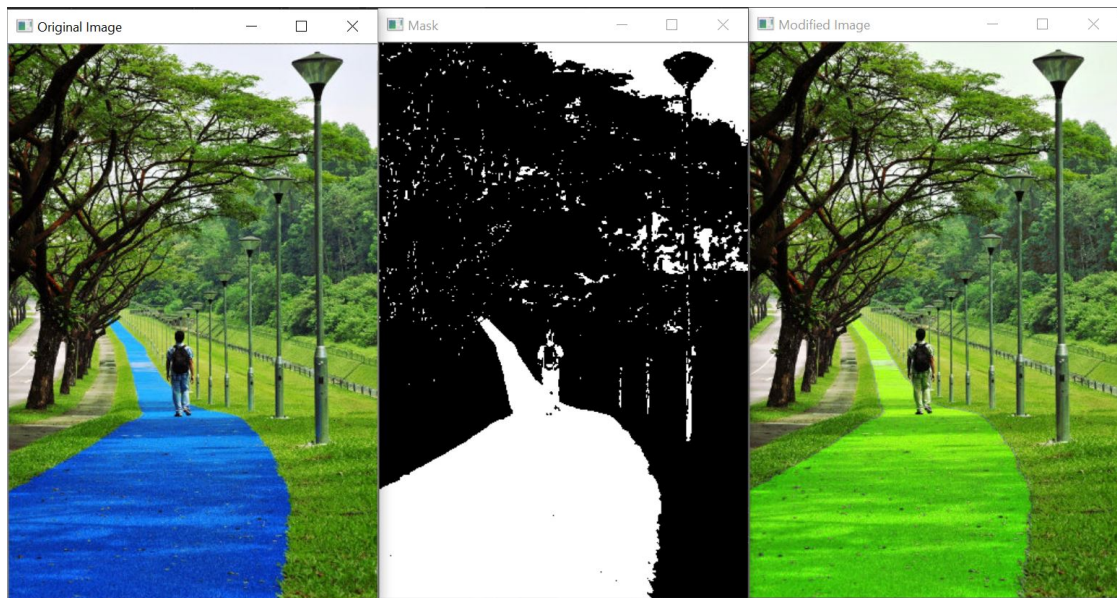
```

28 Mat bw;
29 inRange(hsv, Scalar(80, 0, 0), Scalar(140, 255, 255), bw);
30 namedWindow("Mask", WINDOW_AUTOSIZE);
31 imshow("Mask", bw);
32
33 //Manipulate pixels
34 for (int i = 0; i < image.rows; i++)
35 for (int j = 0; j < image.cols; j++)
36 if (bw.at<uchar>(i, j) > 128)
37 hsv.at<Vec3b>(i, j)[0] = (hsv.at<Vec3b>(i, j)[0] + 120) % 180;
38
39 //Convert to bgr
40 Mat im2;
41 cvtColor(hsv, im2, COLOR_HSV2BGR);
42
43 namedWindow("Modified Image", WINDOW_AUTOSIZE);
44 imshow("Modified Image", im2);
45
46 waitKey(0);
47 return 0;
48 }

```

စာရင်း ၃.၅: အရောင်ရွေးချယ်ပြောင်းလဲခြင်း။

ဤပရိုဂရမ်တွင် cvtColor ကိုသုံး၍ RGB မှ HSV သို့ပြောင်းသည်။ Opencv တွင် Blue channel သည် အရင်ဖြစ်သည်။ HSV space တွင် အပြာရောင်မှာ 240° တွင်ရှိရာ Opencv ၏ hue channel ၏ range 0 - 180 နှင့် အချိုးချ ပါက 120 ဝန်းကျင်ဖြစ်သည်။ ထို့ကြောင့် inRange ကို သုံး၍ 80 - 140 အတွင်း အရောင်များကို mask အနေနှင့် ထုတ်လိုက်သည်။ အပြာကို အစိမ်းရောင် ပြောင်းရန် 240° လှည့်ရန် လိုအပ်သည့်အတွက် pixel တစ်ခုချင်းစီအား 120 ပေါင်းကာ အများဆုံး h တန်ဖိုး 180 ကျော်ပါက color space ၏ အစဘက်မှ ပြန်စ ပေးရန်လိုသည်။ ထိုပုံအား RGB color space သို့ ပြန်ပြောင်း သည့် အခါ ရလာသော ရလဒ်များကို အောက်ပါ ပုံ ၃.၅ တွင် မြင်နိုင်သည်။



ပုံ ၃.၅: မူရင်းပုံ၊ mask နှင့် အရောင်ပြောင်းပြီးသည့် ပုံများ။

အကိုးအကားများ

- [CE14] Cool-Emerald. “Selective Color Modification”. In: (2014). url: <http://coolemerald.blogspot.sg/2014/08/selective-color-modification.html>.
- [Gar+15] Gloria Bueno Garcia et al. Learning Image Processing with OpenCV. Ed. by Julian Ursell et al. First. Packt Publishing Ltd., 2015.
- [Ope14a] OpenCV. Making your own linear filters. 2014. url: http://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/filter_2d/filter_2d.html.
- [Ope16a] OpenCV. Basic Thresholding Operations. 2016. url: http://docs.opencv.org/3.2.0/db/d8e/tutorial_threshold.html.
- [Ope16c] OpenCV. Image Thresholding. 2016. url: http://docs.opencv.org/3.2.0/d7/d4d/tutorial_py_thresholding.html.

- [Ope17a] OpenCV. adaptiveThreshold. 2017. url: http://docs.opencv.org/3.2.0/d7/d1b/group__imgproc__misc.html#ga72b913f352e4a1b1b397736707afcde3.
- [Ope17f] OpenCV. Image filtering. 2017. url: http://docs.opencv.org/3.2.0/d4/d86/group__imgproc__filter.html.

အခန်း ၄

ကြိမ်နှုန်းနယ်တွင်ပြုစပ်ခြင်းများ

ဤ အပိုင်းတွင် 2D ပုံရိပ်များကို ကြိမ်နှုန်းနယ် (frequency domain) တွင် ဖော်ပြခြင်း၊ ပြုစပ်ခြင်းများ အကြောင်း ဆွေးနွေးမည် [Ope15]။ ကြိမ်နှုန်း (Frequency) ကို အခြေခံသော ခွဲခြမ်း စိတ်ဖြာ ခြင်းများ၊ ပြုပြင်ခြင်း များသည် ပုံရိပ်များအား ပိုမို ကောင်းမွန် စေခြင်း၊ ပုံရိပ်များအား ပြန်လည် တည်ဆောက် ခြင်းများ အတွက် အဓိက သော့ချက်ဖြစ်သည်။ တည်ရာနယ် (spatial domain) မှ ကြိမ်နှုန်းနယ် သို့ ပြောင်းလဲရန် Discrete Fourier Transform (DFT) ကို သုံးနိုင်သည်။ အရွယ်အစား $M \times N$ ရှိသော ပုံရိပ်တစ်ခု အတွက် 2 dimensional DFT နှင့် သူ၏ ပြောင်းပြန် (inverse) DFT တို့ကို အောက်ပါ ညီမျှခြင်း များနှင့် ဖော်ပြ နိုင်သည် [GWE04]။

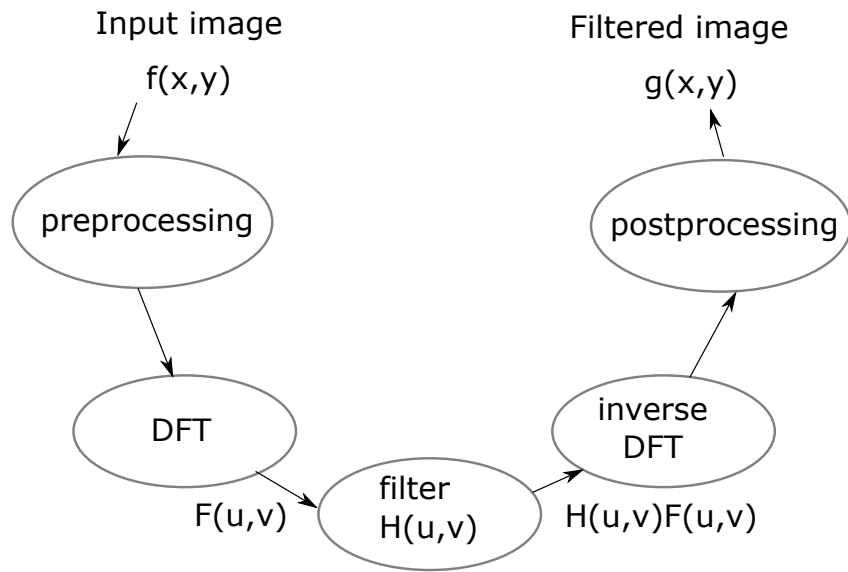
$$F(u, v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(\frac{ux}{M} + \frac{vy}{N})} \quad (၄.၁)$$

$$f(u, v) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(x, y) e^{j2\pi(\frac{xu}{M} + \frac{yv}{N})} \quad (၄.၂)$$

ကြိမ်နှုန်းနယ် တွင် filtering လုပ်သည့် အခါသုံးသည့် အခြေခံ အဆင့် များအား အောက်ပါ ပုံ ၄.၁ တွင် ဖော်ပြ ထားသည်။

$H(u, v)$ ဖြင့် ကိုယ်စားပြု ဖော်ပြသည့် Filter များအား ကြိမ်နှုန်းနယ်တွင် တိုက်ရိုက် ပြုလုပ်နိုင်ပြီး၊ သူတို့၏ အလယ်မှတ် အား origin အနေနှင့် သတ်မှတ်ပါက၊ origin မှ pixel တစ်ခုအထိ အကွာအဝေး $D(u, v)$ အား အောက်ပါအတိုင်း ရှာနိုင်သည်။

$$D(u, v) = \sqrt{u^2 + v^2} \quad (၄.၃)$$



ပုံ ၄.၁: ကြိမ်နှုန်းနယ် တွင် filtering လုပ်ခြင်း။

၄.၁ ကြိမ်နှုန်းနယ်၏ Gaussian Filter များ

Gaussian low pass filter (GLPF) ၏ cutoff frequency အား standard deviation နှင့်တန်ဖိုးတူ σ ဟု သတ်မှတ်ပါက $H(u, v)$ အား အောက်ပါအတိုင်း ဖော်ပြနိုင်သည်။

$$H(u, v) = e^{-\frac{u^2+v^2}{2\sigma^2}} \quad (၄.၄)$$

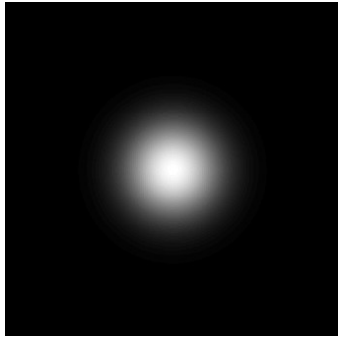
အောက်ပါပုံ ၄.၂ တွင် အလယ်မှတ်အနီး ကြိမ်နှုန်းနိမ့်သည့် အပိုင်းတွင် ပြင်းအားများ၍၊ ကြိမ်နှုန်းမြင့်သည့်အပိုင်းတွင် ပြင်းအားလျော့သွားသည်ကို မြင်နိုင်သည်။

ထိုနည်းတူ Gaussian high pass filter (GHPF) ဆိုပါက အောက်ပါ အတိုင်း ဖော်ပြ နိုင်သည်။

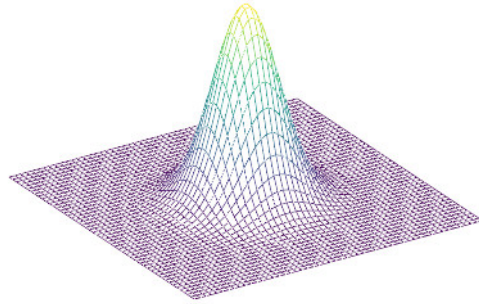
$$H(u, v) = 1 - e^{-\frac{u^2+v^2}{2\sigma^2}} \quad (၄.၅)$$

၄.၂ ကြိမ်နှုန်းနယ်၏ Butterworth Filter များ

Butterworth filter များ၏ ရလဒ်များမှာ ချောမွေ့သဖြင့် သူတို့အား maximally flat magnitude filter များ ဟုလည်း ခေါ်ကြသည်။ $H(u, v)$ ၏ cutoff frequency အား origin မှ အကွာအဝေး D_0



(a) အလင်းအမှောင်။



(b) အနိမ့်အမြင့်။

ပုံ ၄.၂: GLPF ၏ ပမာဏ အား အလင်း ပြင်းအား၊ အနိမ့် အမြင့် တို့ဖြင့် ဖော်ပြ ထားပုံ။

ဟု သတ်မှတ်ပါက၊ order n ရှိသော Butterworth low pass filter (BLPF) အား အောက်ပါအတိုင်း ဖော်ပြနိုင်သည်။

$$H(u, v) = \frac{1}{1 + \left[\frac{u^2 + v^2}{D_0^2} \right]^n} \quad (၄.၆)$$

ထိုနည်းတူ Butterworth high pass filter (BHPF) ဆိုပါက အောက်ပါ အတိုင်း ဖော်ပြ နိုင်သည်။

$$H(u, v) = \frac{1}{1 + \left[\frac{D_0^2}{u^2 + v^2} \right]^n} \quad (၄.၇)$$

ကြိမ်နှုန်းနယ်ပယ်တွင် ပုံတစ်ပုံကို ပြုပြင်ပြီး၊ noise ထည့်ပေးသည့် နမူနာတစ်ခုကို အောက်ပါ frequencyfilter.cpp (စာရင်း ၄.၁) တွင် တွေ့နိုင်သည်။

```

1 //File: freqfilter.cpp
2 //Description: filtering in frequency domain
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
5 //Copyright (c) 2017 Yan Naing Aye
6
7 #include <stdio.h>
8 #include <iostream>
9 #include <opencv2/opencv.hpp>
10 using namespace std;
11 using namespace cv;

```

```

12
13 //Create an image of complex number type (2 channels to store
14 // real part and imaginary part) from an input grayscale image
15 // with an optimal size for DFT
16 // src : single channel grayscale image input
17 // dst : two channel complex image output
18 void i2z(Mat src, Mat& dst)
19 {
20     //get optimal size for dft
21     int m = getOptimalDFTSize(src.rows);
22     int n = getOptimalDFTSize(src.cols);
23
24     //change the image to optimal size, padding 0 values at bottom and right
25     Mat im_padded;
26     copyMakeBorder(src, im_padded, 0, m - src.rows, 0, n - src.cols,
27         BORDER_CONSTANT, Scalar::all(0));
28
29     //convert the image to float type, create another one filled with zeros,
30     //and make an array of these 2 images
31     Mat im_array[] = { Mat_<float>(im_padded), Mat::zeros(im_padded.size(),
32         CV_32F) };
33
34     //combine as a 2 channel image to represent a complex number type image
35     Mat im_complex; merge(im_array, 2, im_complex);
36
37     im_complex.copyTo(dst); //copy to destination
38 }
39
40 // convert a 2 channel complex image to a single channel grayscale image
41 // by getting magnitude, change scale, and normalize
42 // src : two channel complex image input
43 // dst : single channel grayscale image output
44 // LogScale : change to log scale or not
45 void z2i(Mat src, Mat& dst, bool LogScale = false)
46 {
47     Mat im_tmp[2]; split(src, im_tmp); //split the complex image to 2

```

```

46  Mat im_f; magnitude(im_tmp[0], im_tmp[1], im_f); //get absolute value
47  if (LogScale) {
48      im_f += Scalar::all(1); //make minimum 1
49      log(im_f, im_f); //log scale
50  }
51  Mat im;
52  normalize(im_f, im, 0, 255, NORM_MINMAX, CV_8U); //normalize
53      //im = Mat_<uchar>(im_f);
54  im.copyTo(dst); //copy to destination
55  }
56
57  // rearrange the quadrants of a Fourier image
58  void fftshift(Mat src) {
59      // make number of rows or columns even
60      src = src(Rect(0, 0, src.cols & -2, src.rows & -2));
61      int cx = src.cols / 2;
62      int cy = src.rows / 2;
63      Mat q1(src, Rect(cx, 0, cx, cy)); // Top-Right
64      Mat q2(src, Rect(0, 0, cx, cy)); // Top-Left
65      Mat q3(src, Rect(0, cy, cx, cy)); // Bottom-Left
66      Mat q4(src, Rect(cx, cy, cx, cy)); // Bottom-Right
67      Mat tmp;
68      q2.copyTo(tmp);
69      q4.copyTo(q2);
70      tmp.copyTo(q4);
71      q1.copyTo(tmp);
72      q3.copyTo(q1);
73      tmp.copyTo(q3);
74  }
75
76  //Produce Gaussian filter in frequency domain
77  // height : height of the filter
78  // width : width of the filter
79  // std : standard deviation where intensity is 60% (like cutoff frequency
80      )
81  // highpass: true=highpass, false=lowpass

```

```

81 Mat GaussianFilter(int height, int width, float std, bool highpass = false)
82 {
83     Mat im_filter = Mat::ones(Size(width, height), CV_32F);
84     float u, v, t;
85     int k = 0;
86     for (long i = 0; i < height; i++)
87         for (long j = 0; j < width; j++) {
88             u = i - height / 2;
89             v = j - width / 2;
90             t = exp(-(u*u + v*v) / (2.0*std*std));
91             if (highpass) t = 1 - t; //if high pass
92             im_filter.at<float>(i, j) = t;
93         }
94     return im_filter;
95 }
96
97 //Produce Butterworth filter in frequency domain
98 // height : height of the filter
99 // width : width of the filter
100 // cutoffFreq: cutoff frequency where intensity is 50%
101 // order : order n
102 // highpass: true=highpass, false=lowpass
103 Mat ButterworthFilter(int height, int width, float cutoffFreq, int order,
104     bool highpass = false)
105 {
106     Mat im_filter = Mat::ones(Size(width, height), CV_32F);
107     float u, v, t;
108     int k = 0;
109     for (long i = 0; i < height; i++)
110         for (long j = 0; j < width; j++) {
111             u = i - height / 2;
112             v = j - width / 2;
113             t = (u*u + v*v) / (cutoffFreq*cutoffFreq);
114             if (highpass) t = 1 / t; //if high pass
115             for (long k = 1; k < order; k++) t *= t; //for order n
116             t = 1 / (1 + t);

```

```

116     im_filter.at<float>(i, j) = t;
117 }
118 return im_filter;
119 }
120
121 //Add Gaussian noise
122 // src : single channel grayscale image input
123 // dst : single channel grayscale image output
124 // mean: mean
125 // std: standard deviation
126 void GaussianNoise(Mat src, Mat& dst, float mean, float std)
127 {
128     Mat noise = Mat(src.size(), CV_32F);
129     randn(noise, 0, 0.05);
130     Mat im;
131     normalize(src, im, 0.0, 1.0, NORM_MINMAX, CV_32F);
132     im += noise;
133     normalize(im, im, 0, 255, NORM_MINMAX, CV_8U); //normalize
134     im.copyTo(dst); //copy to destination
135 }
136
137 int main(int argc, char** argv)
138 {
139     //Get an image
140     Mat image = imread("C:/opencv/lake.jpg", IMREAD_GRAYSCALE);
141     if (!image.data) {
142         printf("No image data \n");
143         return -1;
144     }
145     namedWindow("Original Image", WINDOW_AUTOSIZE);
146     imshow("Original Image", image);
147
148     Mat im_complex; i2z(image, im_complex); //convert to complex type
149     dft(im_complex, im_complex); //perform discrete fourier transform
150
151     //display Fourier spectrum

```

```

152 Mat im_f; z2i(im_complex, im_f, true);//convert to grayscale
153 fftshift(im_f);//shift quadrants to make the center origin
154 namedWindow("Freq domain Image", WINDOW_AUTOSIZE);
155 imshow("Freq domain Image", im_f);
156
157 //Get a filter (freq domain)
158 //Mat im_filter = ButterworthFilter(im_complex.rows, im_complex.cols, 50,2,
    true);
159 Mat im_filter = GaussianFilter(im_complex.rows, im_complex.cols, 50, false)
    ;
160 namedWindow("Filter", WINDOW_AUTOSIZE);
161 imshow("Filter", im_filter);
162
163 fftshift(im_filter);//shift quadrants for matching
164 //multiply in freq domain (convolve in spatial domain)
165 Mat im_array[2]; split(im_complex, im_array);
166 im_array[0] = im_array[0].mul(im_filter);
167 im_array[1] = im_array[1].mul(im_filter);
168 merge(im_array, 2, im_complex);//combine these 2 into a complex image
169
170 dft(im_complex, im_complex, DFT_INVERSE);//perform inverse discrete fourier
    transform
171 Mat im_hp; z2i(im_complex, im_hp);//get grayscale
172 GaussianNoise(im_hp, im_hp, 0, 0.05);//add noise
173 namedWindow("Filtered Image", WINDOW_AUTOSIZE);
174 imshow("Filtered Image", im_hp);
175
176 //save the result
177 imwrite("C:/opencv/blur.jpg", im_hp);
178 waitKey(0);
179 return 0;
180 }

```

စာရင်း ၄.၁: ကြိမ်နှုန်းနယ်တွင် ဖယ်တာခြင်း။

ပရိုဂရမ်ရှိ $i2z$ ဆိုသည့် ဖန်ရှင်မှာ grayscale ပုံရိပ်တစ်ခု ကို DFT ပြုလုပ်ရန်အတွက် complex number အမျိုးအစား ပြောင်းပေးသည်။ $z2i$ မှာ complex image ကို grayscale ပြန်ပြောင်းပေး သည့် ဖန်ရှင် ဖြစ်သည်။ တွက်၍ ရသော fourier image တွင် သုည ကြိမ်နှုန်းမှာ ဘယ်ဖက်အပေါ်ဖြစ်သဖြင့် origin အမှတ်ကို အလယ်သို့ ရွေ့ရန် fftshift ဖန်ရှင်ကို သုံးနိုင်သည်။ ပရိုဂရမ်၏ ရလဒ်ကို အောက်ပါပုံ ၄.၃ တွင်တွေ့နိုင်သည်။



(a) အဖြူအမည်း မပြောင်းခင်မူရင်းပုံ။



(b) ဝါး၍ noise ပေါင်းထည့်ခြင်း။

ပုံ ၄.၃: နမူနာအနေနှင့် channel တစ်ခုထဲကို ကြိမ်နှုန်းနယ်တွင်ဖယ်တာခြင်း။

၄.၃ Wiener Filtering

ပုံများ ဝါးသွားသည် ဆိုသည်မှာ နီးကပ် နေသော အစက်များ အချင်းချင်း ပေါင်းစပ် ရော နှော သွားသည်ဟု ဆိုနိုင်သည်။ ဖိုးကပ် မပြတ်၍ ဓာတ်ပုံ ဝါးသွား သည်မှာ အပါချာ အဝန်း သဏ္ဌာန် အနီးအနား အစက်များ ရောသွားခြင်း ဟု မြင်နိုင်သည်။ ရွေ့လျား၍ ဝါးသွား (motion blur) သည်မှာ ရွေ့လျားသည့် ဦးတည်ရာ တလျှောက် (များသော အားဖြင့် ဘေးတိုက်) အစက်များ ရောကုန်ခြင်း ဟုမြင်နိုင်သည်။ လက်တုန်၍ ဝါးသွား ပါကလည်း ပုံရိ အလင်း အစက် များ၏ အရေး အကြောင်းကို ကြည့်၍ ဝါးစေသည့် လှိုင်းတွန့်ပုံစံ (point spread function) ကို ခန့်မှန်း နိုင်သည်။ တနည်း အားဖြင့် မူရင်း ပြတ်သားသော ပုံကို၊ degradation function, $H(u, v)$, ဖြင့် convolution လုပ်ထား သည်နှင့် တူညီ သည်။

တည်ရာနယ် (spatial domain) တွင် စဉ်းစားပါက ဝါးနေသော ပုံကို ပြတ်သားအောင် လုပ်ရန်မှာ ပေါင်းစပ် နေသော အစက် တန်ဖိုး များကို ပြန်ခွဲ ထုတ်သည် နှင့်တူသည်။ နမူနာ စဉ်းစား ကြည့်ပါက ပေါင်းလဒ် ကိန်းတစ်ခု ကိုတွေ့ပါက ထိုပေါင်းလဒ်မှာ မည်သည့် ကိန်းများ၏ ပေါင်းလဒ် ဖြစ်မည်နည်းဟု ပြန်ရှာ ရခြင်းနှင့် တူသဖြင့် မသိကိန်း များကို ပြန်ရဖို့ မဖြစ်နိုင်ပါ။ Unsharp mask အစရှိသည့် sharpening filter များမှာလည်း ကြိမ်နှုန်း မြင့် အပိုင်းများကို မြှင့်ပေးရုံ ယေဘုယျ နည်းများသာ ဖြစ်သဖြင့် အမျိုးမျိုးသော အခြား ဝါးပုံ ဝါးနည်းများ အတွက် ထိရောက်မှု မရှိပါ။

ကြိမ်နှုန်းနယ် (frequency domain) တွင်မူ convolution လုပ်သည်မှာ မြှောက်ခြင်း နှင့် ညီမျှသဖြင့် နဂို မူရင်းကို ပြန်လိုချင် ပါက $H(u, v)$ ဖြင့် ပြန်စား လိုက်ရုံပင် ဖြစ်သည်။ ဝါးနေသောပုံ (degraded image) ၏ DFT မှာ $G(u, v)$ ၊ ဝါးစေသော degradation function ၏ DFT မှာ $H(u, v)$ ဆိုပါက ခန့်မှန်း မူရင်းပုံ၏ DFT ကို အောက်ပါအတိုင်း inverse function အနေနှင့် ပြနိုင်သည်။

$$F \cap (u, v) = \frac{G(u, v)}{H(u, v)} \quad (၄.၈)$$

ဤနည်းမှာ ရိုးရှင်း သလို ရှိသော်လည်း noise များ၊ $H(u, v)$ ၏ တိကျမှုများတွင် များစွာ မူတည်လွန်း သဖြင့် ရလဒ် မကောင်း ခြင်းများ၊ ပိုင်းခြေ $H(u, v)$ ၏ သုည နားနီး တန်ဖိုး များတွင် စားလဒ်မှာ သတ်မှတ်၍ မရ တော့သော ပြဿနာ ဖြစ်ခြင်း များရှိသည်။

Wiener Filtering မှာမူ statistical error ကို အနည်းဆုံးဖြစ်အောင် ခန့်မှန်းနိုင်ပြီး၊ အစောဆုံး နှင့် လူကြိုက် အများဆုံး ဖယ်တာ တစ်ခု ဖြစ်သည်။ သူ၏ ကြိမ်နှုန်းနယ် ဖော်ပြ ချက်မှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v) \left[|H(u, v)|^2 + \frac{S_n(u, v)}{S_f(u, v)} \right]} \right] G(u, v) \quad (၄.၉)$$

ထိုတွင် $H(u, v)$ = degradation function ဖြစ်၍၊ $|H(u, v)|^2 = H^*(u, v)H(u, v)$ ဖြစ်သည်။ တဖန် $H^*(u, v)$ သည် $H(u, v)$ ၏ complex conjugate ဖြစ်သည်။ $S_n(u, v) = |N(u, v)|^2$ = noise ၏ power spectrum ဖြစ်၍၊ $S_f(u, v) = |F(u, v)|^2$ = undegraded ပုံရိပ် ၏ power spectrum ဖြစ်သည်။ အချိုး $\frac{S_n(u, v)}{S_f(u, v)}$ ကို noise to signal ratio (NSR) ဟုလည်း ခေါ်သည်။

Wiener Filtering နမူနာကို အောက်ပါ deblur.cpp (စာရင်း ၄.၂) တွင် တွေ့နိုင် သည်။

```
1 //File: deblur.cpp
2 //Description: Wiener filtering
3 //WebSite: http://cool-emerald.blogspot.com
4 //MIT License (https://opensource.org/licenses/MIT)
```

```

5 //Copyright (c) 2017 Yan Naing Aye
6
7 #include <stdio.h>
8 #include <iostream>
9 #include <opencv2/opencv.hpp>
10 using namespace std;
11 using namespace cv;
12
13 //Create an image of complex number type (2 channels to store
14 // real part and imaginary part) from an input grayscale image
15 // with an optimal size for DFT
16 // src : single channel grayscale image input
17 // dst : two channel complex image output
18 void i2z(Mat src, Mat& dst)
19 {
20     //get optimal size for dft
21     int m = getOptimalDFTSize(src.rows);
22     int n = getOptimalDFTSize(src.cols);
23
24     //change the image to optimal size, padding 0 values at bottom and right
25     Mat im_padded;
26     copyMakeBorder(src, im_padded, 0, m - src.rows, 0, n - src.cols,
27         BORDER_CONSTANT, Scalar::all(0));
28
29     //convert the image to float type, create another one filled with zeros,
30     //and make an array of these 2 images
31     Mat im_array[] = { Mat_<float>(im_padded), Mat::zeros(im_padded.size(),
32         CV_32F) };
33
34     //combine as a 2 channel image to represent a complex number type image
35     Mat im_complex; merge(im_array, 2, im_complex);
36
37     im_complex.copyTo(dst); //copy to destination
38 }
39
40 // convert a 2 channel complex image to a single channel grayscale image

```

```

39 // by getting magnitude, change scale, and normalize
40 // src : two channel complex image input
41 // dst : single channel grayscale image output
42 // LogScale : change to log scale or not
43 void z2i(Mat src, Mat& dst, bool LogScale = false)
44 {
45     Mat im_tmp[2]; split(src, im_tmp); //split the complex image to 2
46     Mat im_f; magnitude(im_tmp[0], im_tmp[1], im_f); //get absolute value
47     if (LogScale) {
48         im_f += Scalar::all(1); //make minimum 1
49         log(im_f, im_f); //log scale
50     }
51     Mat im;
52     normalize(im_f, im, 0, 255, NORM_MINMAX, CV_8U); //normalize
53                                     //im = Mat_<uchar>(im_f);
54     im.copyTo(dst); //copy to destination
55 }
56
57 // rearrange the quadrants of a Fourier image
58 void fftshift(Mat src) {
59     // make number of rows or columns even
60     src = src(Rect(0, 0, src.cols & -2, src.rows & -2));
61     int cx = src.cols / 2;
62     int cy = src.rows / 2;
63     Mat q1(src, Rect(cx, 0, cx, cy)); // Top-Right
64     Mat q2(src, Rect(0, 0, cx, cy)); // Top-Left
65     Mat q3(src, Rect(0, cy, cx, cy)); // Bottom-Left
66     Mat q4(src, Rect(cx, cy, cx, cy)); // Bottom-Right
67     Mat tmp;
68     q2.copyTo(tmp);
69     q4.copyTo(q2);
70     tmp.copyTo(q4);
71     q1.copyTo(tmp);
72     q3.copyTo(q1);
73     tmp.copyTo(q3);
74 }

```

```
75
76 //Produce Gaussian filter in frequency domain
77 // height : height of the filter
78 // width  : width of the filter
79 // std    : standard deviation where intensity is 60% (like cutoff frequency
80           )
81 // highpass: true=highpass, false=lowpass
82 Mat GaussianFilter(int height, int width, float std, bool highpass = false)
83 {
84     Mat im_filter = Mat::ones(Size(width, height), CV_32F);
85     float u, v, t;
86     int k = 0;
87     for (long i = 0; i < height; i++)
88         for (long j = 0; j < width; j++) {
89             u = i - height / 2;
90             v = j - width / 2;
91             t = exp(-(u*u + v*v) / (2.0*std*std));
92             if (highpass) t = 1 - t; //if high pass
93             im_filter.at<float>(i, j) = t;
94         }
95     return im_filter;
96 }
97
98 //Produce Butterworth filter in frequency domain
99 // height : height of the filter
100 // width  : width of the filter
101 // cutoffFreq: cutoff frequency where intensity is 50%
102 // order   : order n
103 // highpass: true=highpass, false=lowpass
104 Mat ButterworthFilter(int height, int width, float cutoffFreq, int order,
105                       bool highpass = false)
106 {
107     Mat im_filter = Mat::ones(Size(width, height), CV_32F);
108     float u, v, t;
109     int k = 0;
110     for (long i = 0; i < height; i++)
```

```

109     for (long j = 0; j < width; j++) {
110         u = i - height / 2;
111         v = j - width / 2;
112         t = (u*u + v*v) / (cutoffFreq*cutoffFreq);
113         if (highpass) t = 1 / t; //if high pass
114         for (long k = 1; k < order; k++) t *= t; //for order n
115         t = 1 / (1 + t);
116         im_filter.at<float>(i, j) = t;
117     }
118     return im_filter;
119 }
120
121 int main(int argc, char** argv)
122 {
123     //Get an image
124     Mat image = imread("C:/opencv/blur.jpg", IMREAD_GRAYSCALE);
125     if (!image.data) {
126         printf("No image data \n");
127         return -1;
128     }
129     namedWindow("Original Image", WINDOW_AUTOSIZE);
130     imshow("Original Image", image);
131
132     Mat im_complex; i2z(image, im_complex); //convert to complex type
133     dft(im_complex, im_complex); //perform discrete fourier transform
134     Mat im_co; im_complex.copyTo(im_co); //keep original DFT copy
135
136     //Estimate Signal power spectrum, take low freq part as signal
137     //Get a low pass filter (freq domain)
138     Mat im_filter = ButterworthFilter(im_complex.rows, im_complex.cols, 100, 2,
        false);
139     fftshift(im_filter); //shift quadrants for matching
140     //multiply in freq domain (convolve in spatial domain)
141     Mat im_array[2]; split(im_complex, im_array);
142     im_array[0] = im_array[0].mul(im_filter);
143     im_array[1] = im_array[1].mul(im_filter);

```

```

144 merge(im_array, 2, im_complex); //combine these 2 into a complex image
145
146 dft(im_complex, im_complex, DFT_INVERSE); //perform inverse discrete fourier
    transform
147 Mat im_signal; z2i(im_complex, im_signal); //get grayscale
148
149 //Estimate noise
150 Mat imsp = Mat_<float>(im_signal); //convert image signal to float
151 Mat imor = Mat_<float>(image); //convert original image to float
152 Mat imnps = imor - imsp; //get noise
153
154 //Signal to noise ratio
155 imsp = imsp.mul(imsp); //get signal power spectrum
156 imnps = imnps.mul(imnps); //get noise power spectrum
157 float signalpower = sum(imsp)[0];
158 float noisepower = sum(imnps)[0];
159 float snr = signalpower / noisepower;
160 printf("snr: %f \n", snr);
161
162 //Get the known degrading function
163 Mat im_df = GaussianFilter(im_complex.rows, im_complex.cols, 50, false);
164 fftshift(im_df); //shift quadrants for matching
165
166 //Wiener filtering
167 Mat h2 = im_df.mul(im_df); //magnitude square
168 Mat dbH = (h2 / (h2 + (1 / snr))) / im_df;
169 //multiply in freq domain (convolve in spatial domain)
170 Mat im_coar[2]; split(im_co, im_coar);
171 im_coar[0] = im_coar[0].mul(dbH);
172 im_coar[1] = im_coar[1].mul(dbH);
173 merge(im_coar, 2, im_co); //combine these 2 into a complex image
174
175 //Convert the DFT result into grayscale
176 dft(im_co, im_co, DFT_INVERSE); //perform inverse discrete fourier
    transform
177 Mat im_deblur; z2i(im_co, im_deblur); //get grayscale

```

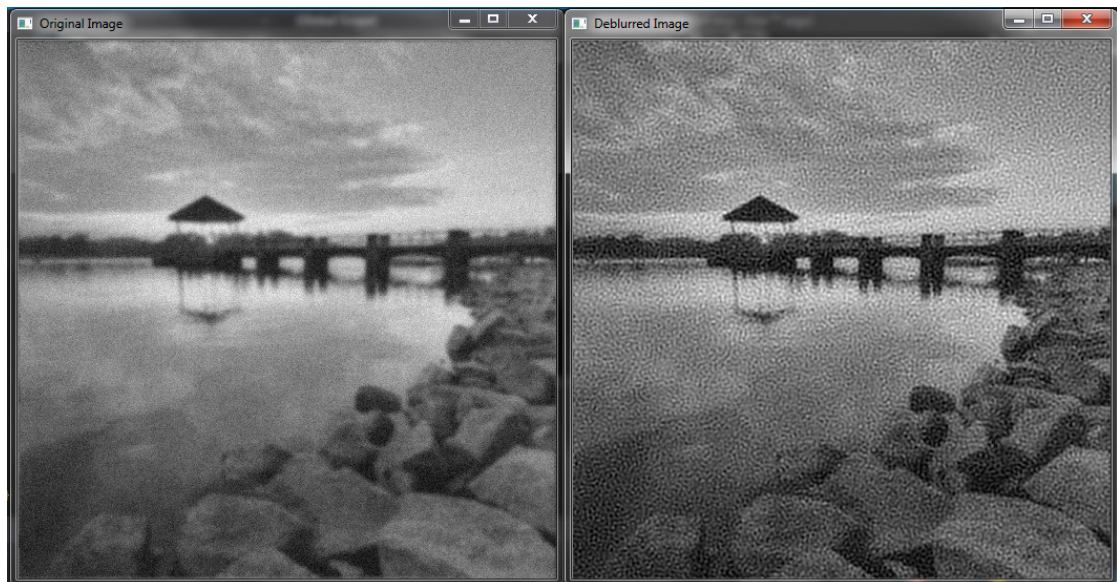
```

178 namedWindow("Deblurred Image", WINDOW_AUTOSIZE);
179 imshow("Deblurred Image", im_deblur);
180
181 //save the result
182 imwrite("C:/opencv/deblur.jpg", im_deblur);
183 waitKey(0);
184 return 0;
185 }

```

စာရင်း ၄.၂: Wiener Filtering

ပရိုဂရမ်၏ ရလဒ်ကို အောက်ပါပုံ ၄.၄ တွင်တွေ့နိုင်သည်။



ပုံ ၄.၄: Noise ထ၍ ဝါးနေသောပုံကို Wiener filter ဖြင့် deblur လုပ်ပြီးသည့် အခါ ရလာသော ရလဒ်။

အကိုးအကားများ

- [GWE04] R.C. Gonzalez, R.E. Woods, and S.L. Eddins. Digital Image Processing Using MATLAB. Pearson Prentice Hall, 2004. isbn: 9780130085191. url: <https://books.google.co.in/books?id=YYuJQgAACAAJ>.

- [Ope15] OpenCV. Discrete Fourier Transform. 2015. url: http://docs.opencv.org/3.1.0/d8/d01/tutorial_discrete_fourier_transform.html.

အခန်း ၅

ဗွီဒီယိုပြုစပ်ခြင်း

ဤ အခန်း တွင် ဗွီဒီယို များကို ဖတ်ခြင်း၊ ရေးခြင်း၊ နှင့် အချိန် နှင့် တပြေးညီ ခွဲခြမ်း၊ ရှာဖွေခြင်း များ အကြောင်း တင်ပြ မည်။ ဗွီဒီယို တစ်ခု ကို ပုံရိပ် များအား အစီအစဉ် လိုက် ပေါင်းစပ် ပြုလုပ် ထားသည် ဟု ဆိုနိုင် သည်။ ထိုသို့ ဖွဲ့စည်း ထားသည့် ပုံရိပ် တစ်ခု ခြင်းစီ ကို ပြကွက် (frame) ဟု ခေါ်ပြီး၊ တစ် စက္ကန့် အတွင်း ပြောင်းလဲ သွားသည့် ပြကွက် အရေ အတွက် ကို ပြန့်နှံ့ (frame rate) ဟု ခေါ်မည် [Lag14]။

၅.၁ ဗွီဒီယို ဖတ်ခြင်း၊ ရေးခြင်း

OpenCV တွင် ဗွီဒီယို များကို ဖမ်းယူရန် သို့မဟုတ် ဖတ်ရန် VideoCapture ဟု ခေါ်သည့် class ပါရှိ သည် [Ope16e]။ ဗွီဒီယို ဖိုင်များ ရေးသားရန် အတွက် ကိုမူ VideoWriter Class ကို သုံးနိုင် သည် [Ope17m]။ စက်ထဲ တွင် သိမ်းထား သည့် ဗွီဒီယို ဖိုင် တစ်ခု ကို ဖတ်ကာ၊ ပြကွက် တစ်ခု ခြင်းစီ ထုတ်ယူ ပြီး၊ လက်ရှိ အချိန် ကို စတုရံ ကွက် နောက်ခံ နှင့် ထို ပြကွက် ပေါ် ထပ်ရေး မည်။ ရလာ သည့် ရလဒ် ကို ဗွီဒီယို ဖိုင် အနေနှင့် ပြန်ရေး သည့် vcap ဆိုသည့် နမူနာ တစ်ခု ကို အောက်ပါ စာရင်း ၅.၁ တွင် ဖော်ပြ ထားသည်။ ပုံ ပေါ်တွင် စာသား များ ရေးခြင်း အတွက် putText [Ope14b] ကို သုံးသည်။

```
1 #include <stdio.h>
2 #include <iostream>
3 #include <opencv2/opencv.hpp>
4 #include <string>
5 using namespace std;
6 using namespace cv;
7 std::string to_string(int i)
```

```

8 {
9 std::stringstream ss;
10 ss << i;
11 return ss.str();
12 }
13 int main(int argc, char** argv)
14 {
15 //VideoCapture cap(0); //Default camera
16 VideoCapture cap("/home/yan/sensor.mp4"); //open video file
17 //VideoCapture cap("c:/opencv/sensor.mp4"); //open video file
18 if (!cap.isOpened()) {
19 printf("Video is not opened. \n");
20 return -1;
21 }
22 else {
23 printf("Video is opened. \n");
24 }
25
26 union { int v; char c[5]; } uEx;
27 uEx.v = static_cast<int>(cap.get(CV_CAP_PROP_FOURCC));
28 uEx.c[4] = '\0';
29 printf("Codec: %s \n", uEx.c);
30
31 Size S = Size((int)cap.get(CV_CAP_PROP_FRAME_WIDTH), (int)cap.get(
    CV_CAP_PROP_FRAME_HEIGHT));
32 printf("Frame size: %d x %d \n", S.width, S.height);
33
34 double rate = cap.get(CV_CAP_PROP_FPS); //Frame rate
35 printf("Frame rate: %f \n", rate);
36 int dperiod = 1000 / rate;
37
38 int c = cap.get(CV_CAP_PROP_FRAME_COUNT);
39 printf("Frame count: %d \n", c);
40 namedWindow("Frame", 1);
41
42 int ex = CV_FOURCC('M', 'J', 'P', 'G');

```

```

43 //int ex = CV_FOURCC('X', 'V', 'I', 'D');//https://www.xvid.com/
44 //int ex = -1;//pop up window to choose
45 //const string vpath="c:/opencv/vdowr.avi";
46 const string vpath="/home/yan/vdowr.avi";
47 VideoWriter outputVideo(vpath, ex , rate, S, true);
48 if (!outputVideo.isOpened())
49 {
50 cout << "Could not open the output video to write."<< endl;
51 waitKey(5000);
52 return -1;
53 }
54
55 long ms,p;
56 Mat frame,img;
57 string str;
58 for (int i = 0;; i++) {
59 if (!cap.read(frame)) break;
60 p = cap.get(CV_CAP_PROP_POS_FRAMES);
61 //printf("Frame: %ld \n", p);
62 ms = cap.get(CV_CAP_PROP_POS_MSEC);
63 str = to_string(ms) + " ms";
64 rectangle(frame, Point(10, 10), Point(200, 60), CV_RGB(255, 0,0), CV_FILLED);
65 putText(frame, str, Point(25, 45),FONT_HERSHEY_PLAIN,2.0, CV_RGB(0, 255, 0),
66         2.0);
67 outputVideo << frame;
68 imshow("Frame", frame);
69 if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
70 }
71 cap.release();
72 outputVideo.release();
73 //waitKey(5000);
74 return 0;
75 }

```

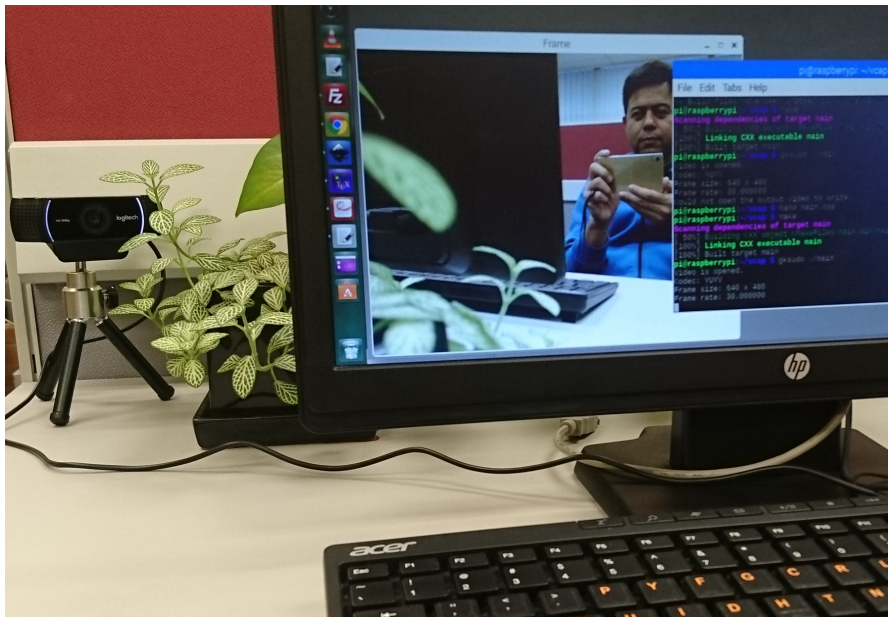
စာရင်း ၅.၁: ဗွီဒီယိုဖတ်ခြင်းရေးခြင်း။

ပရိုဂရမ် အစ ရှိ VideoCapture အတွက် object တစ်ခု ကို initialize ပြုလုပ် ရာတွင် ဗွီဒီယို ဖိုင် တစ်ခု ၏ path ကို ထည့်ပေး ပါက ဖိုင်ကို ဖွင့်ဖတ် ပေးမည် ဖြစ်ပြီး၊ 0 ကို အသုံးပြု ပါက စက်၍ default ကင်မရာ ကို ဖတ် မည် ဖြစ်သည်။ ထို့နောက် ဗွီဒီယို ဖွင့်ခြင်း အဆင်ပြေ မပြေ isOpened ဆိုသည့် method ဖြင့် စစ်ဆေး နိုင်သည်။

လက်ရှိ ဗွီဒီယို ၏ setting များကို get ဟူသည့် method သုံးကာ ဖတ်ကြည့် နိုင်ပြီး၊ set ကို သုံး၍ ပြုပြင် နိုင်သည်။ ဗွီဒီယို ၏ Codec ကို get(CV_CAP_PROP_FOURCC) ဟု ဖတ်နိုင် သည်။ ထို get method ၏ return အမျိုးအစား မှာ double ဖြစ်သည်။ Codec မှာ character လေးလုံး သုံး သဖြင့် union ကို သုံး၍ ဖတ်နိုင် သည်။ ထို့နောက် ပြကွက် အရွယ် အစား၊ ပြန့်နှံး တို့ကို CV_CAP_PROP_FRAME_WIDTH , CV_CAP_PROP_FRAME_HEIGHT , CV_CAP_PROP_FPS တို့ ဖြင့် ဖတ်သည်။

ဆက်လက်၍ ကင်မရာ မှ ဖတ်ရှု ရရှိသော ပုံရိပ် များကို ဗွီဒီယို ဖိုင် အနေဖြင့် သိမ်းရန် VideoWriter တစ်ခု ကို ဖန်တီး သည်။ Initialize ပြုလုပ် ရာတွင် ဖိုင်ကို သိမ်းဆည်း မည့် path ကို ထည့်ပေး နိုင်သည်။ ထို့နောက် codec အတွက် fourcc ဟု ခေါ်သည့် 4-character code ကို ထည့်ပေး သည်။ ဤ နမူနာ တွင် codec ကို motion-jpeg codec သုံးရန် CV_FOURCC('M', 'J', 'P', 'G') ဟု ရယူ သည်။ ထိုသို့ မဟုတ်ပဲ <https://www.xvid.com> တွင် free ရယူ နိုင် သော XVID စသည့် codec တို့ကို CV_FOURCC('X', 'V', 'I', 'D') အစ ရှိသဖြင့် သတ်မှတ် နိုင်သည်။ အကယ်၍ ထို argument အတွက် -1 ကို သုံးပါက GUI ဖြင့် ရွေးချယ်ရန် pop-up window ပေါ်လာ မည်။

နောက်ပိုင်း loop ထဲတွင် VideoCapture ၏ read ဖြင့် frame များကို တစ်ခု ခြင်း ဖတ်၍ VideoWriter ဖြင့် ရေးသည်။ waitKey သည် သတ်မှတ် ထားသည့် milliseconds အချိန် ရပ်စောင့်၍ ထည့်သွင်း ရိုက်ထည့် သည့် key တစ်ခု ကို စောင့်ဖတ် ပေးသည်။ ထိုသို့ ဖတ်နိုင် ရန် imshow ဖြင့် ပြထား သည့် ပြကွက် ဝင်းဒိုး မှာ active ဖြစ်နေရန် လို သည်။ Escape key ၏ တန်ဖိုး 27 ကို ဖတ်၍ ရပါက loop မှ ထွက်၍ ပရိုဂရမ် အဆုံးသတ် သည်။ Webcam တစ်ခု ကို Raspberry Pi တွင် opencv ဖြင့် ချိတ်ဆက် အသုံးပြု နေပုံ ကို ပုံ ၅.၁ တွင် ဖော်ပြ ထားသည်။



ပုံ ၅.၁: Webcam ကို Raspberry Pi ပေါ်တွင် OpenCV ဖြင့် ချိတ်ဆက် အသုံးပြုခြင်း။

၅.၂ Real-time Face Detection

ဗီဒီယို အတွင်း မှ မျက်နှာ များအား အချိန် နှင့် တပြေးညီ ရှာဖွေခြင်း ကို ပြုလုပ် မည်။ ထိုအတွက် **Cascade Classifier [Ope17c]** ကို သုံးနိုင် သည်။ နမူနာ အနေနှင့် **facetedetection** ဆိုသည့် ပရိုဂရမ် ကို အောက်ပါ စာရင်း ၅.၂ တွင် တွေ့နိုင် သည်။

```

1 #include <stdio.h>
2 #include <opencv2/opencv.hpp>
3 #include<string>
4 using namespace std;
5 using namespace cv;
6
7 String face_cascade_name = "c:/opencv/haarcascade_frontalface_alt.xml";
8 String eyes_cascade_name = "c:/opencv/haarcascade_eye_tree_eyeglasses.xml";
9 CascadeClassifier face_cascade;
10 CascadeClassifier eyes_cascade;
11
12 void detectAndDisplay(Mat frame)

```

```

13 {
14     std::vector<Rect> faces;
15     Mat frame_gray;
16
17     cvtColor(frame, frame_gray, CV_BGR2GRAY);
18     equalizeHist(frame_gray, frame_gray);
19
20     //-- Detect faces
21     face_cascade.detectMultiScale(frame_gray, faces, 1.1, 2, 0 |
        CV_HAAR_SCALE_IMAGE, Size(120, 120));
22
23     for (size_t i = 0; i < faces.size(); i++)
24     {
25         Point center(faces[i].x + faces[i].width*0.5, faces[i].y + faces[i].
            height*0.5);
26         ellipse(frame, center, Size(faces[i].width*0.5, faces[i].height*0.5), 0,
            0, 360, Scalar(255, 0, 255), 4, 8, 0);
27
28         Mat faceROI = frame_gray(faces[i]);
29         std::vector<Rect> eyes;
30
31         //-- In each face, detect eyes
32         eyes_cascade.detectMultiScale(faceROI, eyes, 1.1, 2, 0 |
            CV_HAAR_SCALE_IMAGE, Size(30, 30));
33
34         for (size_t j = 0; j < eyes.size(); j++)
35         {
36             Point center(faces[i].x + eyes[j].x + eyes[j].width*0.5, faces[i].y +
                eyes[j].y + eyes[j].height*0.5);
37             int radius = cvRound((eyes[j].width + eyes[j].height)*0.25);
38             circle(frame, center, radius, Scalar(255, 0, 0), 4, 8, 0);
39         }
40     }
41     //-- Show what you got
42     imshow("Frame", frame);
43 }

```

```
44
45 int main(int argc, char** argv)
46 {
47     //Load the cascades
48     if (!face_cascade.load(face_cascade_name)) { printf("--(!)Error loading\n")
49         ; return -1; };
49     if (!eyes_cascade.load(eyes_cascade_name)) { printf("--(!)Error loading\n")
50         ; return -1; };
51
52     VideoCapture cap(0); //Default camera
53     //VideoCapture cap("/home/yan/faces.mp4"); //open video file
54     //VideoCapture cap("c:/opencv/face.mp4"); //open video file
55     if (!cap.isOpened()) { printf("Video is not opened. \n"); return -1; }
56     else { printf("Video is opened. \n"); }
57
58     union { int v; char c[5]; } uEx;
59     uEx.v = static_cast<int>(cap.get(CV_CAP_PROP_FOURCC));
60     uEx.c[4] = '\0';
61     printf("Codec: %s \n", uEx.c);
62
63     cap.set(CAP_PROP_FRAME_WIDTH, 1280);
64     cap.set(CAP_PROP_FRAME_HEIGHT, 720);
65     cap.set(CAP_PROP_FPS, 30);
66     Size S = Size((int)cap.get(CV_CAP_PROP_FRAME_WIDTH), (int)cap.get(
67         CV_CAP_PROP_FRAME_HEIGHT));
68     printf("Frame size: %d x %d \n", S.width, S.height);
69
70     double rate = cap.get(CV_CAP_PROP_FPS); //Frame rate
71     printf("Frame rate: %f \n", rate);
72     int dperiod = 15;
73
74     long p = 0;
75     Mat frame, img;
76     string str;
77     for (int i = 0;; i++) {
```



```

77     if (!cap.read(frame)) break;
78     detectAndDisplay(frame);
79     if (waitKey(dperiod) == 27) break; //if 'Esc' key is pressed
80 }
81 cap.release();
82 //waitKey(5000);
83 return 0;
84 }

```

စာရင်း ၅.၂: Real-time face-detection

ဤ ပရိုဂရမ် တွင် မျက်နှာကို ရှာဖွေရန် “haarcascade_frontalface_alt.xml” နှင့် မျက်လုံးများအား ရှာဖွေရန် “haarcascade_eye_tree_eyeglasses.xml” ဆိုသည့် ဖိုင်များ အား သုံးပါမည်။ ထို ဖိုင်များ သည် “C:/opencv/sources/data/haarcascades” သို့မဟုတ် “/home/username/opencv/data/haarcascades” အောက်တွင် ရှိသည်။ မျက်နှာ နှင့် မျက်လုံးများ အတွက် CascadeClassifier object နှစ်ခုကို ကြေငြာ ပြီး၊ load ကိုသုံး၍ အဆိုပါ ဖိုင် များအား ဖတ်သည်။

အချိန် နှင့် တပြေးညီ ဗီဒီယို ရိုက်ရန် VideoCapture object ကို ကြေငြာ သည့် အခါ ဖိုင် နာမည် အစား 0 ကို သုံးသည့် အတွက် သင့် စက် တွင် တပ်ဆင် ထားသော ကင်မရာ ကို သုံးသည်။ VideoCapture ၏ get ကို သုံး၍ လက်ရှိ ဗီဒီယို ၏ setting များကို ဖတ်နိုင်ပြီး၊ set ကို သုံး၍ လိုသလို ပြင်နိုင် သည်။ ဤ ပရိုဂရမ် တွင် ပြကွက် အရွယ် အစား နှင့် ပြန့်နှံ့မှုကို set နှင့် ပြင်လိုက် သည်။ ထို့နောက် ပြကွက် တစ်ခုခြင်း ဖတ်၍ detectAndDisplay ဖန်ရှင်ကို သုံးကာ မျက်နှာ ကို ရှာ၍ ဝိုင်း ပြသည်။

detectAndDisplay ဖန်ရှင် တွင် ပုံရိပ်ကို အဖြူ အမည်း ပြောင်း၍ detectMultiScale ကို သုံး၍ ရှာသည်။ သူ၏ ပုံစံမှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

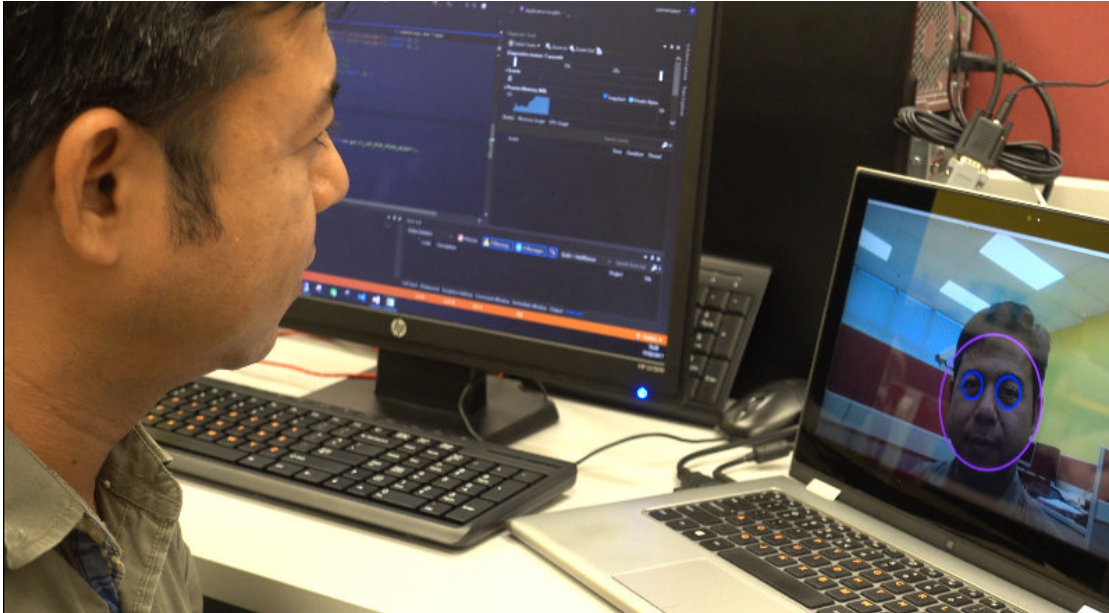
```

void CascadeClassifier::detectMultiScale(const Mat& image,
vector<Rect>& objects, double scaleFactor=1.1,
int minNeighbors=3, int flags=0,
Size minSize=Size(), Size maxSize=Size())

```

ပထမ Parameters ဖြစ်သည့် image မှာ ရှာလို သည့် ပုံရိပ် ဖြစ်ပြီး၊ ဒုတိယ objects တွင် ရှာတွေ့ သည့် နေရာ များ၏ စတုရံ များအား သိမ်းပေး မည်။ ထို့နောက် ရှာတွေ့သည့် နေရာ နှင့် အရွယ် များကို

သုံး၍ circle နှင့် စက်ဝိုင်း များ ဆွဲ၍ ပုံကို imshow နှင့် ပြပေး သည်။ အောက်ပါ ပုံ ၅.၂ တွင် တချိန်နှင့် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း နှင့် ရလာ သည့် ပုံတို့ ကို ပြထား သည်။



ပုံ ၅.၂: တချိန် နှင့် တပြေးညီ မျက်နှာ ရှာဖွေ ခြင်း။

အကိုးအကားများ

- [Lag14] Robert Laganieri. OpenCV Computer Vision Application Programming Cookbook. 2nd. Packt Publishing, 2014. isbn: 1782161481, 9781782161486.
- [Ope14b] OpenCV. putText: Draws a text string. 2014. url: http://docs.opencv.org/3.0-beta/modules/imgproc/doc/drawing_functions.html#puttext.
- [Ope16e] OpenCV. VideoCapture Class Reference. 2016. url: http://docs.opencv.org/3.2.0/d8/dfe/classcv_1_1VideoCapture.html.
- [Ope17c] OpenCV. Cascade Classifier. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/objdetect/cascade_classifier/cascade_classifier.html.
- [Ope17m] OpenCV. VideoWriter Class Reference. 2017. url: http://docs.opencv.org/trunk/dd/d9e/classcv_1_1VideoWriter.html.

အခန်း ၆

ပုံရိပ်ခွဲခြမ်းသုံးသပ်ရှာဖွေခြင်းများ

ဤအခန်းတွင် ပုံရိပ် များအား ခွဲခြမ်းစိစစ်ခြင်း၊ သုံးသပ်ခြင်း၊ ရှာဖွေခြင်း များအား ပြုလုပ်မည်။

၆.၁ Connected Component Labeling

Connected component labeling သည် binary image တစ်ခုတွင်ရှိသော ဆက်စပ်နေသည့် region များကို မတူညီသည့် label များတပ်ပေးသည်။ ဤ algorithm သည် ပုံရိပ်တစ်ခုကို နှစ်ခေါက်သာ စစ်ရန်လိုသဖြင့် floodfill စသည့် နည်းများနှင့် နှိုင်းယှဉ်လျင် ထိရောက်မြန်ဆန်မှု များစွာသာလွန်သည်။ သူ၏ ပုံစံမှာ အောက်ပါအတိုင်းဖြစ်သည် [Ope16b]။

```
int cv::connectedComponents ( InputArray image,
OutputArray labels,
int connectivity = 8,
int ltype = CV_32S
)
```

ဆက်စပ်နေသည့် region အရေအတွက် စုစုပေါင်း N ကို return ပြန်ပေး သည်။ Label များမှာ $[0, N - 1]$ ဖြစ်ပြီး၊ ၀ မှာ နောက်ခံ အပိုင်းအတွက် label ဖြစ်သည်။ နောက်တစ်မျိုးမှာ connectedComponentsWithStats ဖြစ်ပြီး၊ ဆက်စပ်နေသော region များအား ရှာပေး သည့် အပြင်၊ သူတို့၏ ဂျီဩမေတြီ ဗဟို (centroid)၊ အကျယ်အဝန်း (area)၊ နေရာ စသည်တို့ကို ပါတွက်ချက် ပေးသည်။ သူ၏ ပုံစံမှာ အောက်ပါအတိုင်းဖြစ်သည်။

```
int cv::connectedComponentsWithStats ( InputArray  image,
OutputArray  labels,
OutputArray  stats,
OutputArray  centroids,
int  connectivity = 8,
int  ltype = CV_32S
)
```

stats ကို stats(label, COLUMN) ဖြင့် ရယူနိုင်ပြီး column မှာ

```
CC_STAT_LEFT
CC_STAT_TOP
CC_STAT_WIDTH
CC_STAT_HEIGHT
CC_STAT_AREA
CC_STAT_MAX
```

တို့မှ တစ်ခုခု ဖြစ်သည်။ သူတို့၏ နမူနာကို အောက်ပါ စာရင်း ၆.၁ တွင် တွေ့နိုင်သည်။

```
1 #include <stdio.h>
2 #include <iostream>
3 #include <opencv2/opencv.hpp>
4
5 using namespace std;
6 using namespace cv;
7
8 int main()
9 {
10 //Read original image
11 Mat src;
12 src = imread("C:/opencv/ka_ori.jpg", 0);
13 if (!src.data) { printf("No image data \n"); return -1; }
14
15 // scale - to make it convenient to print out and see
16 Mat ims; resize(src, ims, Size(16, 8));
```

```

17 Mat imb; threshold(ims, imb, 128, 255, CV_8U); // make binary
18
19 namedWindow("Original Image", WINDOW_AUTOSIZE);
20 imshow("Original Image", imb);
21 cout << "Original" << endl << imb << endl << endl;
22
23 //Connected components
24 Mat imr; int N=connectedComponents(imb, imr);
25 cout << "CCL" << endl << imr << endl << endl;
26 cout<<"Number of components: "<< N-1 << endl; // 0 is background
27
28 //Connected components with stats
29 Mat cc, sts, ctd; N=connectedComponentsWithStats(imb, cc, sts, ctd, 8);
30
31 //get area column except background
32 Mat A(sts, Rect(CC_STAT_AREA, 1, 1, sts.rows - 1));
33
34 // Localizing the biggest component
35 double minVal, maxVal;
36 Point minLoc, maxLoc;
37 minMaxLoc(A, &minVal, &maxVal, &minLoc, &maxLoc);
38 printf("Max area is %lf at label %d .\n", maxVal, (maxLoc.y + 1));
39
40 printf("Centroid of the biggest component: [%f,%f] \n", ctd.at<double>(maxLoc
    .y + 1, 0), ctd.at<double>(maxLoc.y + 1, 1));
41
42 waitKey(0);
43 return 0;
44 }

```

စာရင်း ၆.၁: ဆက်စပ်နေသော နယ်မြေ များအား ရှာခြင်း။

အစတွင် ပုံရိပ် အစက်တန်ဖိုး များကို ရိုက်ပြကာ အလွယ်အကူ ပြန်စစ်နိုင်ရန် အတွက်၊ သေးငယ်သည့် အရွယ်အစား 16 x 8 အဖြစ် `resize` ကိုသုံး၍ ပြောင်းလိုက်သည်။ ထို့နောက် binary ပုံရိပ် ရစေရန် `threshold` နှင့် ပြောင်းပြီး မူရင်းပုံ ကို ပြသည်။ 16 x 8 အရွယ် သာဖြစ်သဖြင့်

```

C:\Users\DELL\Desktop\space\OpenCV_Eg\OpenCV320\x64\...
Original
[ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
  0, 0, 0, 0, 255, 255, 255, 0, 0, 255, 255, 255, 0, 0, 0;
  0, 0, 0, 255, 0, 0, 0, 255, 255, 0, 0, 0, 0, 0, 0;
  0, 0, 0, 255, 0, 0, 0, 0, 255, 0, 0, 0, 255, 0, 0;
  0, 255, 0, 255, 255, 0, 0, 255, 255, 0, 255, 255, 0, 0, 0;
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

CCL
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0;
 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0;
 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 3, 0, 0;
 0, 2, 0, 1, 1, 0, 0, 1, 1, 0, 3, 3, 0, 0, 0;
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0;
 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Number of components: 3
Max area is 15.000000 at label 1 .
Centroid of the biggest component: [6.400000,3.266667]

```

ပုံ ၆.၁: ဆက်စပ်နေသော နယ်မြေ များအား အစက်တန်ဖိုး တစ်ခုခြင်း အနေနှင့် တွေ့နိုင်ပြီး၊ သူတို့၏ label များ၊ အရေအတွက်၊ အကြီးဆုံး နယ်မြေ၏ အကျယ်အဝန်း၊ ဗဟိုတို့ကို ဖော်ပြထားသည်။

အလွန်သေးငယ်သည်။ ထို့နောက် အစက်တစ်ခု ခြင်းစီ၏ တန်ဖိုးကို cout နှင့် ရိုက်ပြသည့် အတွက် console တွင် ဆက်စပ်နေသော အစက်များကို မျက်စိဖြင့် မြင်နိုင်သည်။

connectedComponents ကိုသုံး၍ ရရှိလာသော label များကို cout ၏ အတွက်တွင် မြင်နိုင်သည်။ Region များ၏ အရေအတွက်ကို တွေ့နိုင်ပြီး၊ stats ၏ area များကို ထုတ်ယူသည်။ ထို့နောက် **min-MaxLoc** ဖြင့် အကြီးဆုံး နယ်မြေကို ရှာကာ သူ၏ centroid များရှာသည်။ ပရိုဂရမ်၏ ရလဒ်ကို ပုံ ၆.၁ တွင်တွေ့နိုင်သည်။

၆.၂ Template Matching

ပုံရိပ်တစ်ခုတွင် သတ်မှတ်ထားသော ပုံစံ နမူနာနှင့် ကိုက်ညီသည့် နေရာကို ရှာဖွေခြင်းအား Template Matching ဟုခေါ်မည်။ ထိုသို့ ရှာဖွေရန် **matchTemplate** [Ope17k] ကိုသုံးနိုင်သည်။ သူ၏ ပုံစံမှာ အောက်ပါအတိုင်းဖြစ်သည်။

```
void cv::matchTemplate ( InputArray image,
InputArray templ,
OutputArray result,
int method,
InputArray mask = noArray()
)
```

Template matching methodမှာ အောက်ပါတို့မှ တစ်ခုခု ဖြစ်နိုင်သည်။

```
TM_SQDIFF
TM_SQDIFF_NORMED
TM_CCORR
TM_CCORR_NORMED
TM_CCOEFF
TM_CCOEFF_NORMED
```

နမူနာ ပရိုဂရမ်ကို အောက်ပါ စာရင်း ၆.၂ တွင် တွေ့နိုင်သည်။

```
1 #include "opencv2/highgui/highgui.hpp"
2 #include "opencv2/imgproc/imgproc.hpp"
```



```

3 #include <iostream>
4 #include <stdio.h>
5
6 using namespace std;
7 using namespace cv;
8
9 /// Global Variables
10 Mat img; Mat templ; Mat result;
11 int match_method;
12
13 void MatchingMethod(int, void*)
14 {
15     // Source image to display with rectangle overlay
16     Mat img_display;
17     img.copyTo(img_display);
18
19     // Do the Matching and Normalize
20     matchTemplate(img, templ, result, match_method);
21     normalize(result, result, 0, 1, NORM_MINMAX);
22
23     // Localizing the best match with minMaxLoc
24     double minVal, maxVal;
25     Point minLoc, maxLoc, matchLoc;
26     minMaxLoc(result, &minVal, &maxVal, &minLoc, &maxLoc, Mat());
27
28     // For SQDIFF and SQDIFF_NORMED, the best matches are lower values.
29     // For all the other methods, the higher the better
30     if (match_method == CV_TM_SQDIFF || match_method == CV_TM_SQDIFF_NORMED) {
31         matchLoc = minLoc; }
32     else { matchLoc = maxLoc; }
33
34     // Show me what you got
35     Point matchLoc2(matchLoc.x + templ.cols, matchLoc.y + templ.rows);
36     rectangle(img_display, matchLoc, matchLoc2, Scalar::all(0), 2, 8, 0);
37     rectangle(result, matchLoc, matchLoc2, Scalar::all(0), 2, 8, 0);
38     imshow("Source", img_display);

```

```

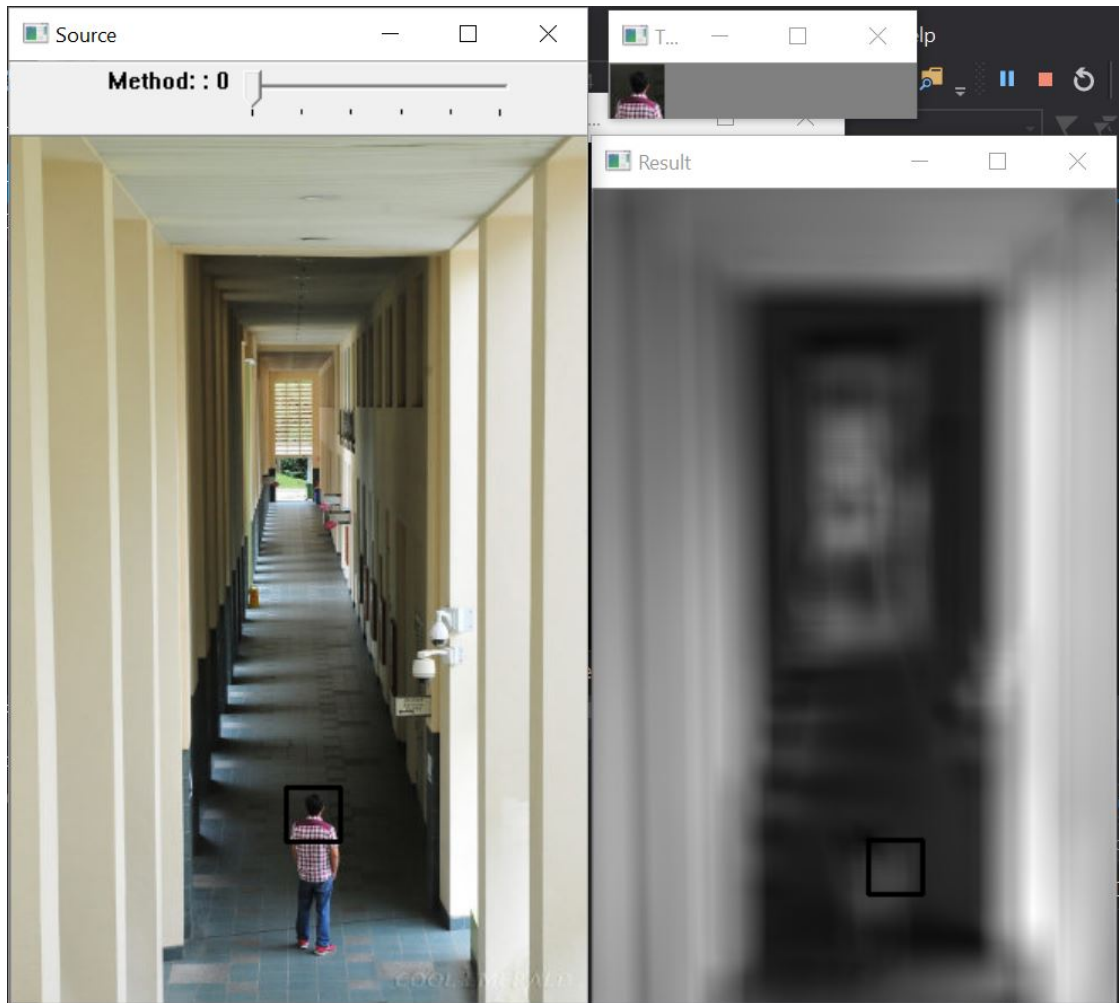
38 imshow("Result", result);
39 }
40
41 int main()
42 {
43     /// Load image and template
44     img = imread("c:/opencv/stand.jpg");
45     templ = imread("c:/opencv/template.jpg");
46
47     /// Create windows
48     namedWindow("Source", CV_WINDOW_AUTOSIZE);
49     namedWindow("Result", CV_WINDOW_AUTOSIZE);
50     namedWindow("Template", CV_WINDOW_AUTOSIZE);
51     imshow("Template", templ);
52     /// Create Trackbar
53     createTrackbar("Method: ", "Source", &match_method, 5, MatchingMethod);
54     MatchingMethod(0, 0);
55
56     waitKey(0);
57     return 0;
58 }

```

စာရင်း ၆.၂: သတ်မှတ် ထားသော ပုံစံ တစ်ခု နှင့် ကိုက်ညီသော နေရာ အား ရှာဖွေခြင်း။

ဤ ပရိုဂရမ်တွင် မူရင်းပုံနှင့် ရှာဖွေလိုသည့် ပုံစံ (template) တို့ကို ဖတ်ပြီး၊ trackbar မှ ရွေးချယ်သည့် matching method အတိုင်း template နှင့် အကိုက်ညီဆုံး နေရာ ကို မူရင်းပုံတွင် ရှာသည်။ Template matching တွက်၍ရသည့် တန်ဖိုးများကို result window တွင် ပြသည်။

ရလဒ်များမှာ CV_TM_SQDIFF နှင့် CV_TM_SQDIFF_NORMED နည်းများအတွက် တန်ဖိုးအငယ်ဆုံး မှာ ကိုက်ညီမှု အရှိဆုံး နေရာ ဖြစ်ပြီး၊ ကျန်သော နည်းများ အတွက်မူ တန်ဖိုး အကြီးဆုံးမှာကိုက်ညီမှု အရှိဆုံး နေရာဖြစ်သည်။ ထိုအတွက် minMaxLoc ကို သုံးသည်။ အကိုက်ညီဆုံးနေရာတွင် မူရင်းပုံ နှင့် ရလဒ်များ၌ စတုရန်းပုံ နှင့်ပြသည်။ ပရိုဂရမ် ၏ ရလဒ် ကို ပုံ ၆.၂ တွင် တွေ့နိုင်သည်။



ပုံ ၆.၂: Template matching လုပ်၍ရသော ပုံ။

၆.၃ Harris Corner Detector

ဤ အပိုင်းတွင် အသွင်အပြင်များ ရှာဖွေခြင်း (feature detection) နမူနာ အနေနှင့် `cornerHarris` ကိုသုံးကာ Harris-Stephens method နှင့် ပုံရိပ်တစ်ခုရှိ ထောင့်များအား ရှာဖွေခြင်း အကြောင်း တင်ပြမည် [Ope17e]။ `cornerHarris` ၏ ပုံစံ မှာ အောက်ပါ အတိုင်း ဖြစ်သည်။

```
void cornerHarris(InputArray src, OutputArray dst,
    int blockSize, int ksize, double k,
    int borderType=BORDER_DEFAULT )
```

နမူနာ ပရိုဂရမ်ကို စာရင်း ၆.၃ တွင်ဖော်ပြထားသည်။

```
1 #include "opencv2/highgui/highgui.hpp"
2 #include "opencv2/imgproc/imgproc.hpp"
3 #include <iostream>
4 #include <stdio.h>
5
6 using namespace std;
7 using namespace cv;
8
9 int main()
10 {
11     /// Load image and template
12     Mat img = imread("c:/opencv/stand.jpg",0);
13     namedWindow("Source", CV_WINDOW_AUTOSIZE);
14     imshow("Source", img);
15     // Detect Harris Corners
16     Mat cornerStrength;
17     cornerHarris(img,cornerStrength,3,3,0.01);
18     normalize(cornerStrength, cornerStrength, 0, 255, NORM_MINMAX, CV_8U);
19     namedWindow("Corner Strength",CV_WINDOW_AUTOSIZE);
20     imshow("Corner Strength", cornerStrength);
21     // threshold the corner strengths
22     Mat harrisCorners;
23     threshold(cornerStrength, harrisCorners,24, 255, cv::THRESH_BINARY);
24     namedWindow("Corners", CV_WINDOW_AUTOSIZE);
```

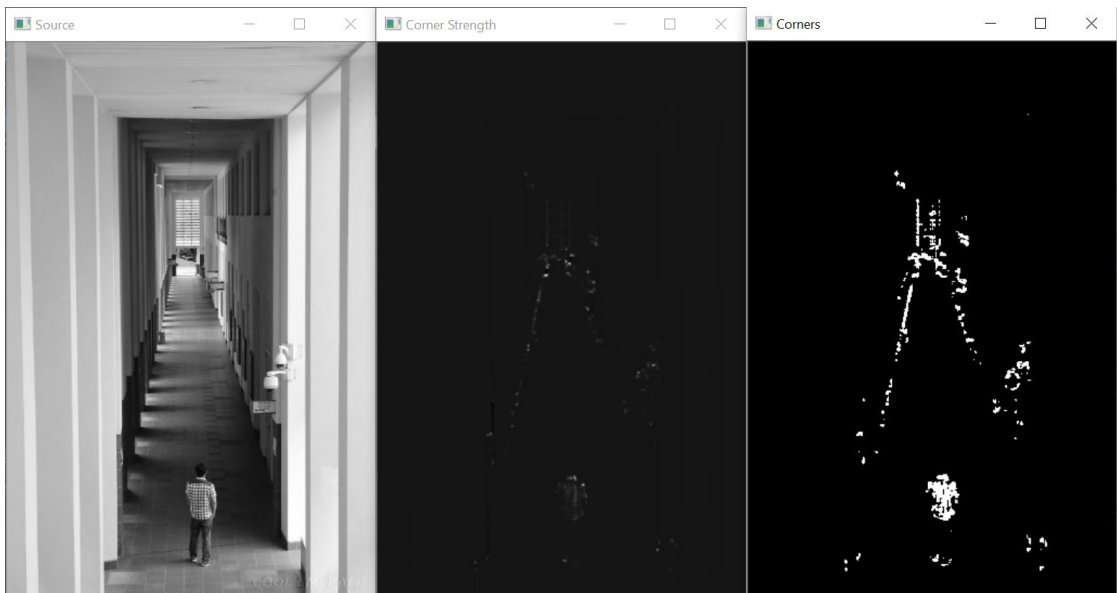
```

25  imshow("Corners", harrisCorners);
26  waitKey(0);
27  return 0;
28  }

```

စာရင်း ၆.၃: ထောင့်အသွင်အပြင် များအား ရှာဖွေခြင်း။

ပရိုဂရမ် တွင် ပုံရိပ်တစ်ခု ကို ဖတ်ကာ cornerHarris နှင့် ထောင့် အသွင်အပြင်များ ၏ တွေ့ရှိမှု ပြင်းအား (corner strength) ကိုတွက်သည်။ ထို့နောက် threshold နှင့် ဖြတ်ချကာ ထောင့်များကို ဖော်ပြသည်။ ပရိုဂရမ်၏ ရလဒ်ကို ပုံ ၆.၃ တွင် တွေ့နိုင်သည်။



ပုံ ၆.၃: ထောင့် အသွင်အပြင် များကို ရှာဖွေခြင်း။

၆.၄ မြန်မာနိုင်ငံ၏ဗဟို

ပုံသဏ္ဌာန်တစ်ခု၏ geometric center ကို centroid, $\mathbf{C} = [C_X \ C_Y]^T$, ဟုခေါ်ပြီး အောက်ပါ ညီမျှခြင်း များဖြင့် ရှာနိုင်သည်။

$$C_X = \frac{\sum(x \times i(x, y))}{\sum i(x, y)} \quad (6.၁)$$

$$C_Y = \frac{\sum(y \times i(x, y))}{\sum i(x, y)} \quad (၆.၂)$$

ထိုတွင် x နှင့် y မှာ pixel တစ်ခုခြင်းစီ၏ coordinates နေရာဖြစ်ပြီး၊ $i(x, y)$ မှာ ထို သက်ဆိုင် ရာ pixel ၏ intensity တန်ဖိုးဖြစ်သည်။

နမူနာ အနေနှင့် မြန်မာနိုင်ငံ မြေပုံ၏ အလယ်ဗဟို ကို ရှာမည်။ ထိုအတွက် Google Maps တွင် မြန်မာ နိုင်ငံ၏ မြေပုံကို screen shot ရယူကာ GIMP တွင် edit လုပ်၍ အဖြူအမည်း ပြောင်းလိုက်မည် (ပုံ ၆.၄)။ ပုံအတွက် crop ဖြတ်လိုက်သော border လတ္တီကျု၊ လောင်ဂျီကျု များမှာ



(a) Google maps ရှိ မြန်မာနိုင်ငံ မြေပုံ။



(b) GIMP ကို သုံး၍ အဖြူအမည်း ပြောင်းပြီးပုံ။

ပုံ ၆.၄: မြန်မာနိုင်ငံ မြေပုံအား Google maps မှရယူလိုက်သော မူရင်းနှင့်၊ GIMP ကိုသုံး၍ အစွန်း များနှင့် အတိ crop လုပ်၊ အဖြူအမည်း ပြောင်းထားပုံ။

- မြောက်ဘက် ၂၈.၅၅°

- တောင်ဘက် ၉.၈၄°
- အရှေ့ဘက် ၁၀၁.၁၇°
- အနောက်ဘက် ၉၂.၁၇°

တို့ဖြစ်သည်။ Google maps မှာ Mercator projection သုံးထားသဖြင့် ညီမျှခြင်း (၆.၁) နှင့် (၆.၂) တို့တွင် အစက်တစ်ခုခြင်း စီ၏ normalized ဧရိယာ မှာ အီကွေတာ အတွက် 1 ဆိုပါက လတ္တီကျု ϕ အလိုက် ဧရိယာ မှာ $\cos(\phi)$ ဖြစ်သည်။ မြန်မာပြည် မြေပုံ၏ အလယ်ဗဟိုအား ရှာဖွေသည့် ပရိုဂရမ်ကို စာရင်း ၆.၄ တွင်ဖော်ပြထားသည်။

```

1 #include <stdio.h>
2 #include <math.h>
3 #include <opencv2/opencv.hpp>
4 #include <iostream>
5 using namespace std;
6 using namespace cv;
7 int main()
8 {
9     Mat image;
10    image = imread("C:/opencv/MyanmarMap6.PNG", IMREAD_GRAYSCALE);
11    if (!image.data) { printf("No image data \n"); return -1; }
12    namedWindow("Display Image", WINDOW_AUTOSIZE);
13    imshow("Display Image", image);
14    double latNorth = 28.55;
15    double latSouth = 9.84;
16    double logEast = 101.17;
17    double logWest = 92.17;
18    double sxi = 0, syi = 0; //for image
19    double slat = 0, slog = 0; //for latitude and longitude
20    double cslat = 0, cslog = 0; //with compensation for
21    // Universal Transverse Mercator (UTM) coordinate system
22    double si = 0;
23    double ilat, jlog;
24    double ci = 0; //compensated intensity
25    double sci = 0; //summation of ci
26    const double PI = 2 * acos(0.0); //get PI

```

```

27 //Manipulate pixels
28 long h = image.rows;
29 long w = image.cols;
30 long ithreshold = 138;
31 for (long i = 0; i < h; i++)
32 for (long j = 0; j < w; j++)
33 if (image.at<uchar>(i, j) > ithreshold) {
34 si += 1; //uniform pixel intensity
35 sxi += j;
36 syi += i;
37 ilat = latNorth - (latNorth - latSouth)*double(i) / double(h - 1);
38 jlog = logWest + (logEast - logWest)*double(j) / double(w - 1);
39 slog += jlog;
40 slat += ilat;
41 ci = cos(ilat*PI / 180.0);
42 sci += ci;
43 cslog += jlog*ci;
44 cslat += ilat*ci;
45 }
46
47 //image centroid
48 double cx = sxi / si;
49 double cy = syi / si;
50
51 //latitude and longitude centroid
52 double lat = slat / si;
53 double log = slog / si;
54
55 //compensated latitude and longitude centroid
56 double clat = cslat / sci;
57 double clog = cslog / sci;
58
59 printf("Centroid row col: [ %f , %f ] for image pixels. \n", cx, cy);
60 printf("Centroid lat log: [ %f , %f ] \n", lat, log);
61 printf("Centroid lat log: [ %f , %f ] after compensation for Mercator
    projection.\n", clat, clog);

```



```

62 waitKey(0);
63 return 0;
64 }

```

စာရင်း ၆.၄: မြန်မာပြည် မြေပုံ၏ အလယ်ဗဟိုအား ရှာဖွေခြင်း။

ပရိုဂရမ်မှ တွက်ပေးသော မြန်မာနိုင်ငံ၏ အလယ်ဗဟိုမှာ Mercator projection အား ထည့်စဉ်းစား ပါက လတ္တီကျု ၂၀.၈၆ နှင့် လောင်ဂျီကျု ၉၆.၄၈ တို့ဖြစ်သည်။ Google maps တွင် 20.86,96.48 ဟုရှာပါက အောက်ပါ ပုံ ၆.၅ အတိုင်း ဗဟိုနေရာအား တွေ့နိုင် သည်။



ပုံ ၆.၅: မြန်မာပြည် မြေပုံ၏ အလယ်ဗဟို။

အကိုးအကားများ

[Ope16b] OpenCV. Connected component labeling. 2016. url: http://docs.opencv.org/3.2.0/d3/dc0/group__imgproc__shape.html.

- [Ope17e] OpenCV. Harris corner detector. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/features2d/trackingmotion/harris_detector/harris_detector.html.
- [Ope17k] OpenCV. Template matching. 2017. url: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html.

အခန်း ၇

GUI Applications with wxWidgets

[wxWidgets](#) က Windows ၊ Linux နဲ့ Mac OSX အစရှိတဲ့ ပလက်ဖောင်း အမျိုးမျိုး ပေါ်မှာ GUI applications တွေ ရေးဖို့ အတွက် C++ library ပါ။ သူနဲ့ GUI code တွေ ရေးပြီး ရင် ပလက်ဖောင်း အမျိုးမျိုး ပေါ်မှာ ကုဒ် ကို သိပ်ပြင် စရာ မလိုပဲ တန်းပြီး compile လုပ်၊ run လုပ်လို့ ရပါတယ်။ ဒီ အခန်းမှာ OpenCV ကို wxWidgets နဲ့ တွဲသုံးတဲ့ အကြောင်း ဆွေးနွေး ပါမယ်။

wxWidgets က free လည်းပေး open source လည်း ဖြစ် တဲ့ software ပါ။ ကိုယ်ပိုင် ဆော့ဝဲ တွေ ထုတ်မယ် ဆိုရင် လည်း ဘာမှ ကန့်သတ်ချက် တွေ မရှိ ပါဘူး [[wxW98](#)]။ အဲဒါက Qt နဲ့ အဓိက ကွာခြားချက် ပါ။ Qt က LGPLv3 လိုင်စင် ကို free ပေးထားပြီး၊ ကိုယ်ပိုင် စီးပွားရေး အတွက် သုံးမယ် ဆိုရင် ကန့်သတ်ချက် တချို့ ရှိတာကြောင့် လိုင်စင် ဝယ်ဖို့ လိုကောင်း လိုနိုင် ပါတယ် [[Qt17](#)]။

Native platform ကို တတ်နိုင် သလောက် သုံးထား တာမို့ wxWidgets သုံးလို့ ရလာတဲ့ GUI တွေဟာ သုံးတဲ့ platform နဲ့ လိုက်ဖက်ပြီး ပင်ကို အမြင် အတိုင်း တသားထဲ ကျ တာကို ခံစား ရမှာပါ [[wxW12](#)]။ Standard C++ ကိုပဲ သုံးထားပြီး Qt တို့လို အထူး extension တွေ သုံး မထား တဲ့ အတွက် ရှုပ်ထွေးမှု နည်းတာ ကလည်း ကောင်းတဲ့ အချက် တစ်ခုပါ။

wxWidgets နဲ့ ရလာတဲ့ binary application တွေဟာ သေးငယ် ပေါ့ပါး တာမို့ embedded system တွေအတွက် အထူး သင့်တော် ပါတယ်။ နောက်တစ်ခါ library အရွယ်အစား တွေ ယှဉ်ရင်လည်း ဥပမာ အနေနဲ့ Qt library ကို တပ်ဆင်ရင် ≈ 200 MB လောက် အရွယ် ရှိပေမယ့် wxWidgets library က ≈ 30 MB လောက်ပဲ ယူပါတယ်။

wxWidgets က C++ အတွက် သာ မကပဲ python, perl, php, java, lua, lisp, erlang, eiffel, C# (.NET), BASIC, ruby နဲ့ javascript အတွက် တောင်မှ bindings [[wxW15a](#)] တွေ ရှိပါတယ်။ wxWidgets က တော်တော် ပြည့်စုံ ရင့်ကျက် တဲ့ GUI toolkits ဖြစ်ပြီး၊ utility classes လည်း အများကြီး

ရှိတာမို့ ကောင်းမွန် သင့်တော် တဲ့ GUI toolkits အနေနဲ့ ညွှန်းဆို ချင်ပါတယ်။

wxWidgets ကို အသုံးပြုတဲ့ သူတွေ၊ အဖွဲ့အစည်းတွေ အများကြီး ရှိပြီး အဲဒီ အထဲမှာ လူသိများ တာတွေက NASA, AMD, Xerox, နဲ့ Open Source Applications Foundation (OSAF) တို့ ဖြစ်ပါတယ်။ ထင်ရှားတဲ့ wxWidgets applications တွေက AVG AntiVirus, Audacity, Filezilla, Code::Blocks, CodeLite တို့ ဖြစ်ပါတယ်။

၇.၁ Linux တွင်တပ်ဆင်ခြင်း

wxWidgets ကို Linux မှာ ထည့်ဖို့အတွက် အောက်က စာရင်း ၇.၁ အတိုင်း terminal မှာ command တွေရိုက်ပြီး တပ်ဆင်ထည့်သွင်းနိုင် ပါတယ် [wxW14]။

```
1 sudo apt-get update
2 sudo apt-get install build-essential
3 sudo apt-get install libwxgtk3.0-dev
```

စာရင်း ၇.၁: Linux တွင် wxWidgets ကို တပ်ဆင်ခြင်း။

နမူနာ အနေနဲ့ ရိုးရှင်း တဲ့ wxsimple.cpp ([online](#)) ဆိုတဲ့ ပရိုဂရမ် လေးကို စာရင်း ၇.၂ အတိုင်း ဖန်တီး လိုက် ပါမယ် [Zet13]။

```
1 #include <wx/wx.h>
2 class Simple : public wxFrame
3 {
4 public:
5     Simple(const wxString& title);
6
7 };
8 Simple::Simple(const wxString& title)
9     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(250, 150))
10 {
11     Centre();
12 }
13
14 class MyApp : public wxApp
15 {
```

```

16 public:
17     virtual bool OnInit();
18 };
19
20 IMPLEMENT_APP(MyApp)
21 bool MyApp::OnInit()
22 {
23     Simple *simple = new Simple(wxT("Simple"));
24     simple->Show(true);
25
26     return true;
27 }

```

စာရင်း ၇.၂: wxsimple.cpp

ပြီးတဲ့ အခါ အောက်က command ကို သုံးပြီး compile လုပ်နိုင် ပါတယ်။

```
g++ wxsimple.cpp `wx-config --cxxflags --libs` -o wxsimple
```

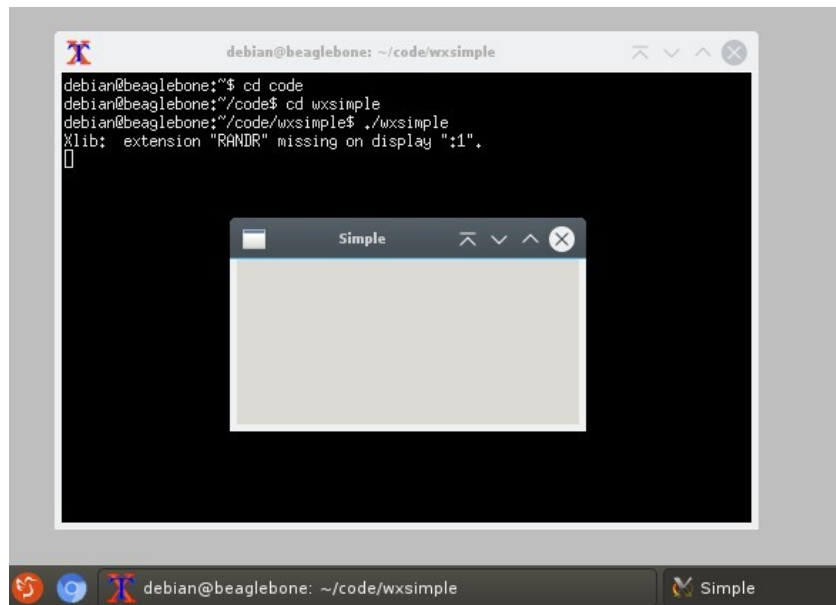
`wx-config -cxxflags` က compile လုပ်ဖို့ အတွက် လိုအပ်တဲ့ flags တွေကို ထုတ်ပေး ပြီး `wx-config -libs` က link လုပ်ဖို့ အတွက် လိုအပ်တဲ့ flags တွေကို ထုတ်ပေး ပါတယ်။

ရှိတဲ့ Terminal က အဆင် မပြေ ရင် xterm ကို အောက်ပါ အတိုင်း တပ်ဆင် နိုင် ပါတယ်။

```
sudo apt-get install xterm
```

ပြီးရင် ခုနက ရလာတဲ့ binary ကို အောက်က အတိုင်း run လိုက်တဲ့ အခါ ပုံ ၇.၁ အတိုင်း တွေ့နိုင် ပါတယ်။

```
./wxsimple
```



ပုံ ၇.၁: wxWidgets အတွက် wxsimple.cpp နမူနာ ကို BeagleBone Black, Debian Wheezy တွင် run ခြင်း။

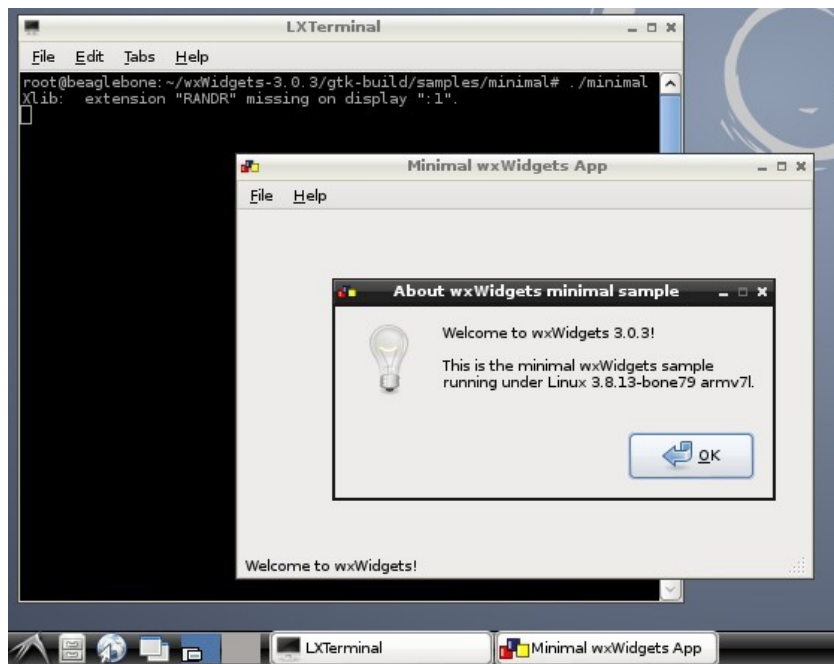
၇.၁.၁ Source မှ build လုပ်ခြင်း

wxWidgets ကို source ကနေ စိတ်ကြိုက် build လုပ်ချင် ရင် အောက်က အတိုင်း build လုပ်ပြီး install လုပ်နိုင် ပါတယ် [wxW14]။

```
sudo apt-get install build-essential
sudo apt-get install libgtk-3-dev
wget https://github.com/wxWidgets/wxWidgets/releases/download/v3.0.3/
    wxWidgets-3.0.3.tar.bz2
tar xjvf wxWidgets-3.0.3.tar.bz2
cd wxWidgets-3.0.3
mkdir gtk-build
cd gtk-build
../configure --enable-unicode --disable-shared --with-gtk=3
make
sudo make install
wx-config --version
```

Build လုပ်ပြီးတဲ့ wxWidgets ကို သုံးပြီး samples အခန်း ထဲက minimal ဆိုတဲ့ နမူနာကို run ကြည့် လိုက်တဲ့ အခါ ပုံ ၇.၂ အတိုင်း တွေ့ရ ပါတယ်။

```
cd gtk-build/samples/minimal
make
./minimal
```



ပုံ ၇.၂: wxWidgets ဖြင့် minimal နမူနာကို run ခြင်း။

၇.၂ OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း

OpenCV နှင့် wxWidgets ကို တွဲသုံး ထားတဲ့ ပရိုဂရမ် တွေကို Linux နဲ့ terminal ပေါ်မှာ command ရိုက်ထည့် build လုပ်တာ လွယ်ကူ ရိုးရှင်း ပါတယ်။ နမူနာ အနေနဲ့ wxcvsimple.cpp ([online](#)) ဆိုတဲ့ ရိုးရှင်း တဲ့ ပရိုဂရမ် လေး တစ်ခု ရေးကြည့် ပါမယ်။ ပရိုဂရမ် အစမှာ Application ရဲ့ OnInit() ဆိုတဲ့ method ထဲမှာ

```
wxInitAllImageHandlers();
```


ဆိုတာကို ထည့်ပါမယ်။ အဲဒီနောက် MyFrame ဆိုတဲ့ wxFrame ရဲ့ derived class ထဲမှာ ပုံရိပ် တွေကို ဖော်ပြဖို့ wxStaticBitmap variable တွေကို ကြေငြာ လိုက် ပါမယ်။ MyFrame ရဲ့ constructor မှာ wxStaticBitmap တွေကို ဖန်တီးဖို့ အောက်က ကုဒ် ကို သုံးနိုင် ပါတယ်။

```
lena = new wxStaticBitmap(this,wxID_ANY,wxBitmap(wxT("lena.jpg"),
wxBITMAP_TYPE_JPEG),wxPoint(0,0),wxSize(256, 256));
```

Image တွေ အတွက် OpenCV မှာသုံးတဲ့ Mat နဲ့ wxWidgets ရဲ့ wxImage ကို အပြန် အလှန် ပြောင်း ပေးတဲ့ mat_from_wx၊ wx_from_mat အစ ရှိတဲ့ ဖန်ရှင် တွေကို စာရင်း ၇.၃ က util.h ([online](#)) ထဲမှာ မှာ ဖော်ပြ ထား ပါတယ် [Dad10; pkl15]။

```
1 //File: util.h
2 //Description: Functions to convert wxImage and OpenCV Mat
3 //Author: Yan Naing Aye
4 //MIT License - Copyright (c) 2017 Yan Naing Aye
5
6 #include <wx/wx.h>
7 #include <opencv2/opencv.hpp>
8 using namespace cv;
9
10 wxImage wx_from_mat(Mat &img) {
11     Mat im2;
12     if(img.channels()==1){cvtColor(img,im2,CV_GRAY2RGB);}
13     else if (img.channels() == 4) { cvtColor(img, im2, CV_BGRA2RGB);}
14     else {cvtColor(img,im2,CV_BGR2RGB);}
15     long imsize = im2.rows*im2.cols*im2.channels();
16     wxImage wx(im2.cols, im2.rows,(unsigned char*)malloc(imsize), false);
17     unsigned char* s=im2.data;
18     unsigned char* d=wx.GetData();
19     for (long i = 0; i < imsize; i++) { d[i] = s[i];}
20     return wx;
21 }
22
23 Mat mat_from_wx(wxImage &wx) {
24     Mat im2(Size(wx.GetWidth(),wx.GetHeight()),CV_8UC3,wx.GetData());
```

```

25     cvtColor(im2,im2,CV_RGB2BGR);
26     return im2;
27 }

```

စာရင်း ၇.၃: util.h

ဆက်ပြီး MyFrame ရဲ့ constructor ထဲမှာ သီရိ ရဲ့ ပုံကို OpenCV သုံးပြီး imread နဲ့ ဖတ်လိုက်ပါတယ်။ ဖတ်လို့ရတဲ့ Mat အမျိုးအစား ပုံကို wx_from_mat ဖန်ရှင် ကို သုံးပြီး wxImage အမျိုးအစား ပြောင်းလိုက်ပါတယ်။ ပြီးတော့ wxStaticBitmap ရဲ့ SetBitmap method သုံးပြီး Frame ပေါ်မှာ ဖော်ပြလိုက်ပါတယ်။

နောက်တစ်ခါ wxImage အမျိုးအစား variable တစ်ခု ကြေငြာ ပြီး LoadFile method နဲ့ လီနာ ရဲ့ ပုံကို wxWidget သုံးပြီး ဖတ်လိုက်ပါတယ်။ ရလာတဲ့ wxImage အမျိုးအစား ပုံကို mat_from_wx ဆိုတဲ့ ဖန်ရှင် သုံးပြီး OpenCV အတွက် Mat အမျိုးအစား ပြောင်းလိုက်ပါတယ်။ ရလာတဲ့ ပုံရိပ်ကို OpenCV ရဲ့ imshow နဲ့ ဖော်ပြလိုက်ပါတယ်။ ပရိုဂရမ် wxcvsimple.cpp ကို စာရင်း ၇.၄ မှာ တွေ့နိုင်ပါတယ်။

```

1 //File: wxcvsimple.cpp
2 //Description: A simple example to use OpenCV with wxWidgets
3 //Author: Yan Naing Aye
4 //Date: 2017 November 01
5 //MIT License - Copyright (c) 2017 Yan Naing Aye
6
7 #include <wx/wx.h>
8 #include <opencv2/opencv.hpp>
9 #include "util.h"
10 using namespace cv;
11
12 class MyFrame : public wxFrame
13 {
14     wxStaticBitmap *lena;
15     wxStaticBitmap *thiri;
16 public:
17     MyFrame(const wxString& title);
18

```

```

19 };
20 MyFrame::MyFrame(const wxString& title)
21     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(512, 256))
22 {
23     Centre();
24     lena = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("lena.jpg"),
25     wxBITMAP_TYPE_JPEG), wxPoint(0,0), wxSize(256, 256));
26
27     thiri = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("lena.jpg"),
28     wxBITMAP_TYPE_JPEG), wxPoint(256,0), wxSize(256, 256));
29
30     //From opencv to wx
31     Mat imcv1=imread("thiri.jpg", CV_LOAD_IMAGE_COLOR);
32     wxBitmap imwx1=wx_from_mat(imcv1);
33     thiri->SetBitmap(imwx1);
34
35     //From wx to opencv
36     wxImage imwx2;
37     imwx2.LoadFile(wxT("lena.jpg"), wxBITMAP_TYPE_JPEG);
38     Mat imcv2=mat_from_wx(imwx2);
39     namedWindow("Img", CV_WINDOW_AUTOSIZE);
40     imshow("Img", imcv2);
41 }
42
43 class MyApp : public wxApp
44 {
45     public:
46     virtual bool OnInit();
47 };
48
49 IMPLEMENT_APP(MyApp)
50 bool MyApp::OnInit()
51 {
52     if ( !wxApp::OnInit() )
53         return false;
54     wxInitAllImageHandlers();
55     MyFrame *frame = new MyFrame(wxT("Simple wxWidgets + OpenCV"));

```

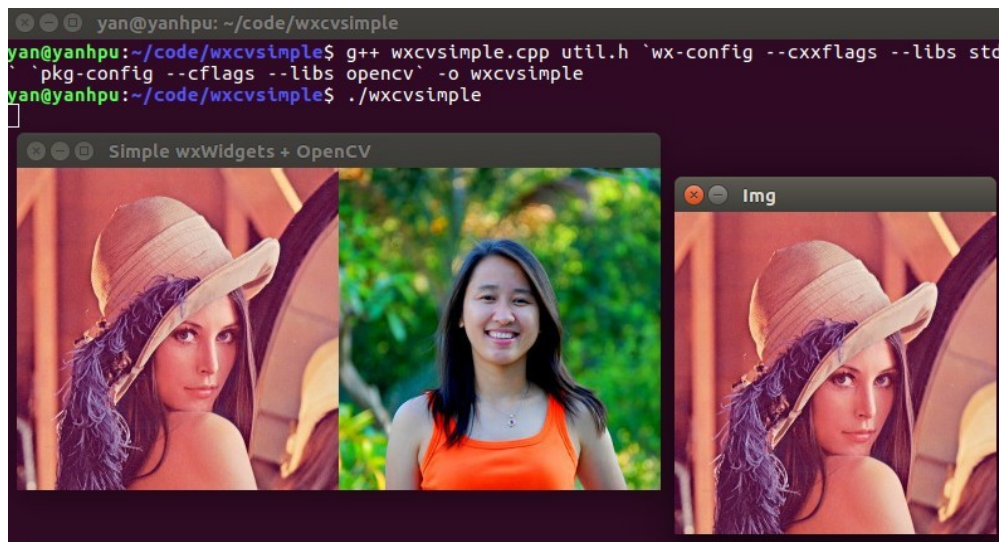
```
53     frame->Show(true);
54
55     return true;
56 }
```

စာရင်း ၇.၄: wxcvsimple.cpp

ပရိုဂရမ်ကို build လုပ်ဖို့ အတွက် terminal မှာ အောက်က command ကို သုံးနိုင်ပါတယ်။

```
g++ wxcvsimple.cpp util.h `wx-config --cxxflags --libs std` `pkg-config --cflags --libs opencv` -o wxcvsimple
```

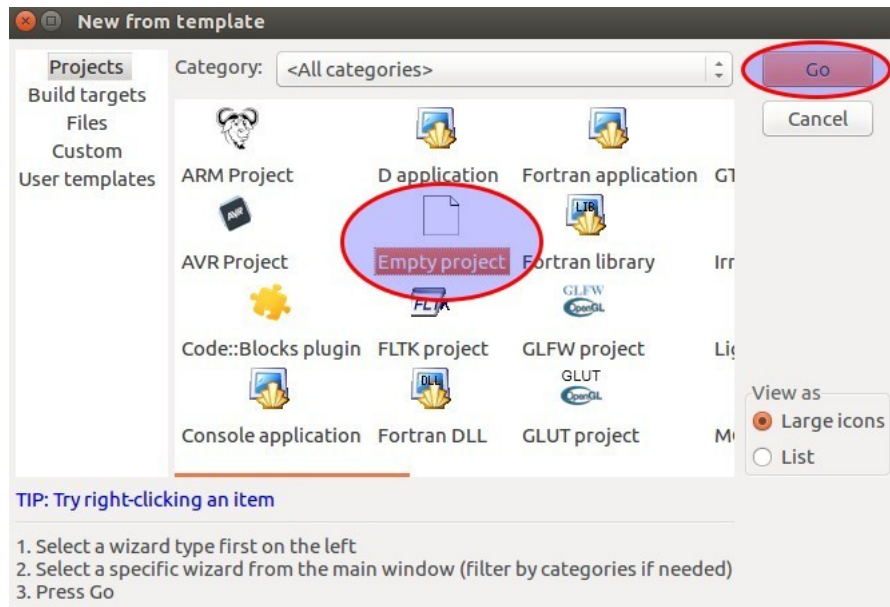
./wxcvsimple ကို ရိုက်ပြီး run လိုက်တဲ့ အခါ ရလာတဲ့ ရလဒ်ကို ပုံ ၇.၃ မှာ တွေ့နိုင်ပါတယ်။



ပုံ ၇.၃: wxWidgets နှင့် OpenCV ကို တွဲစပ် အသုံးပြု ထားသည့် ရိုးရှင်းသော နမူနာ။

၇.၃ Code::Blocks တွင်တွဲသုံးခြင်း

Code::Blocks မှာ wxWidgets ကို OpenCV နဲ့ တွဲသုံး ဖို့အတွက် wxWidgets project အသစ် တစ်ခုကို ပုံ ၇.၄ မှာ ဖော်ပြ ထားတဲ့ အတိုင်း empty project တစ်ခု ဖန်တီးပြီး wxcv ([online](#)) ဆိုတဲ့ project နာမည် ပေးလိုက် ပါမယ်။ ပြီးတဲ့ အခါ GNU GCC Compiler ကို ရွေး ပါမယ်။



ပုံ ၇.၄: Code::Blocks တွင် Empty project တစ်ခု ဖန်တီးခြင်း။

File Menu → New → Empty File ကို နှိပ်ပြီး စာရင်း ၇.၅ မှာ ပြထားတဲ့ wxcv.cpp မိုင်ကို ဖန်တီးလိုက် ပါမယ်။

```

1
2 #include <wx/wx.h>
3 #include <opencv2/opencv.hpp>
4 using namespace cv;
5
6 class Simple : public wxFrame
7 {
8 public:
9     Simple(const wxString& title);
10
11 };
12 Simple::Simple(const wxString& title)
13     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(250, 150))
14 {
15     Centre();
16     Mat img=imread("grove.jpg",CV_LOAD_IMAGE_COLOR);

```

```

17     namedWindow("Img", CV_WINDOW_AUTOSIZE);
18     imshow("Img", img);
19 }
20
21 class MyApp : public wxApp
22 {
23     public:
24         virtual bool OnInit();
25 };
26
27 IMPLEMENT_APP(MyApp)
28 bool MyApp::OnInit()
29 {
30     Simple *simple = new Simple(wxT("Simple"));
31     simple->Show(true);
32
33     return true;
34 }

```

စာရင်း ၇.၅: wxcv.cpp

ပြီးရင် Project Menu → Build Options... ကို နှိပ်ပြီး Compiler settings tab → Other compiler options မှာ ပုံ ၇.၅ အတိုင်း

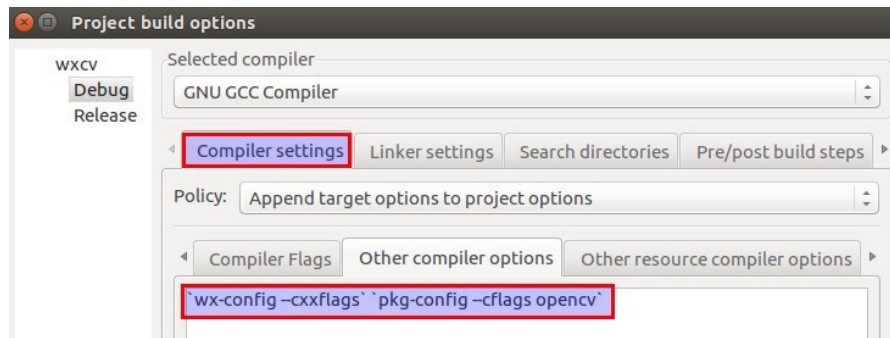
```
`wx-config --cxxflags` `pkg-config --cflags opencv`
```

ကို Debug အတွက်ရော၊ Release အတွက်ပါ သတ်မှတ် နိုင် ပါတယ်။ နောက်တစ်ခါ Linker settings tab → Other linker options မှာ လည်း ပုံ ၇.၆ အတိုင်း

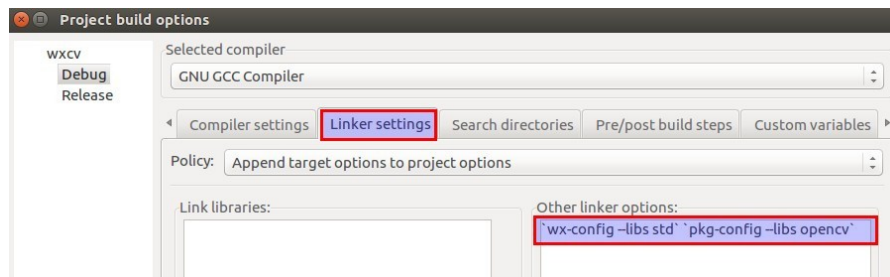
```
`wx-config --libs std` `pkg-config --libs opencv`
```

ကို ထပ် သတ်မှတ်ပါမယ်။

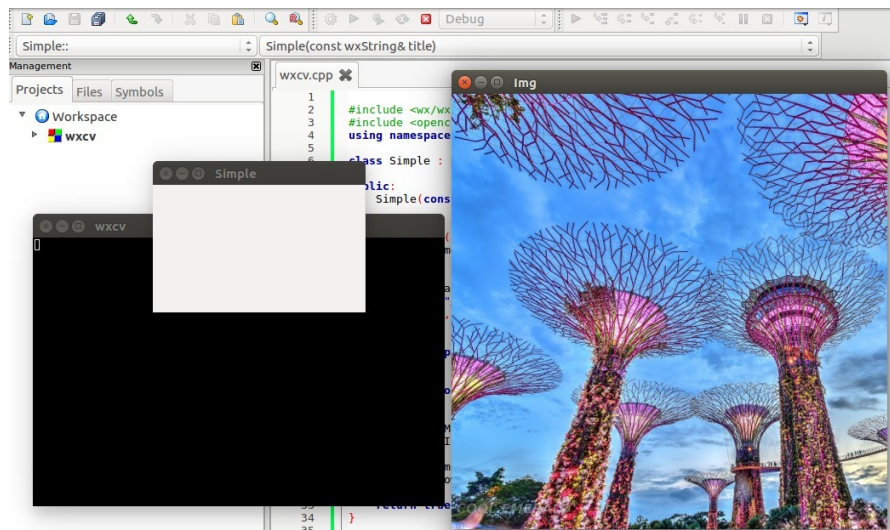
နောက်ဆုံးမှာ F9 ခလုတ်ကို နှိပ်ပြီး Build and run လုပ်လိုက် တဲ့အခါ ပုံ ၇.၇ မှာ ပြထား သလို ပရိုဂရမ် ရဲ့ output ကို တွေ့နိုင် ပါတယ်။



ပုံ ၇.၅: Compiler options များသတ်မှတ်ခြင်း။

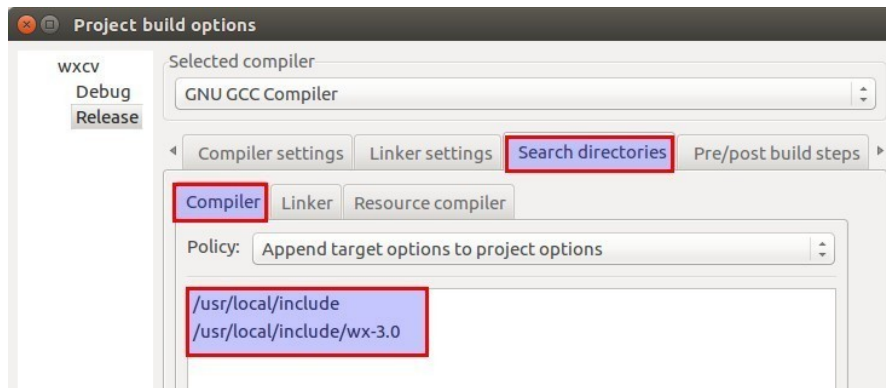


ပုံ ၇.၆: Linker options များသတ်မှတ်ခြင်း။



ပုံ ၇.၇: OpenCV နမူနာ ပရိုဂရမ်ကို Code::Blocks တွင် run ခြင်း။

ပရောဂျက် နဲ့ ပတ်သက် တဲ့ အချက်အလက် တွေကို IDE ကို ပိုပြီး သိစေချင်ရင် optional အဆင့်တွေ အနေနဲ့ Project Menu → Build Options... ကို နှိပ်ပြီး Search directories tab → Compiler tab မှာ ပုံ ၇.၈ အတိုင်း /usr/local/include နဲ့ /usr/local/include/wx-3.0 ကို Debug အတွက်ရော၊ Release အတွက်ပါ သတ်မှတ် နိုင် ပါတယ်။



ပုံ ၇.၈: Compiler Search directories များသတ်မှတ်ခြင်း။

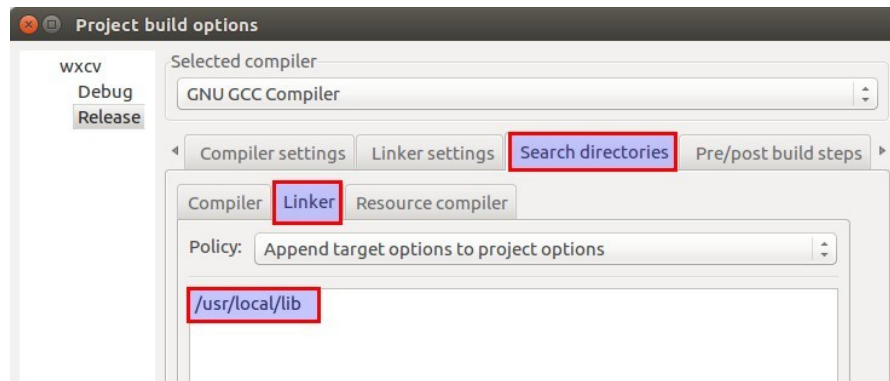
နောက်တစ်ခါ ဘေးဘက် က Linker tab မှာ ပုံ ၇.၉ အတိုင်း lib တွေရဲ့လမ်းကြောင်း /usr/local/lib ကို သတ်မှတ် ပါမယ်။ Linker settings tab မှာလည်း ပုံ ၇.၁၀ အတိုင်း Link libraries တွေ ဖြစ်တဲ့

```
libopencv_core.so
libopencv_highgui.so
libopencv_imgcodecs.so
libopencv_imgproc.so
```

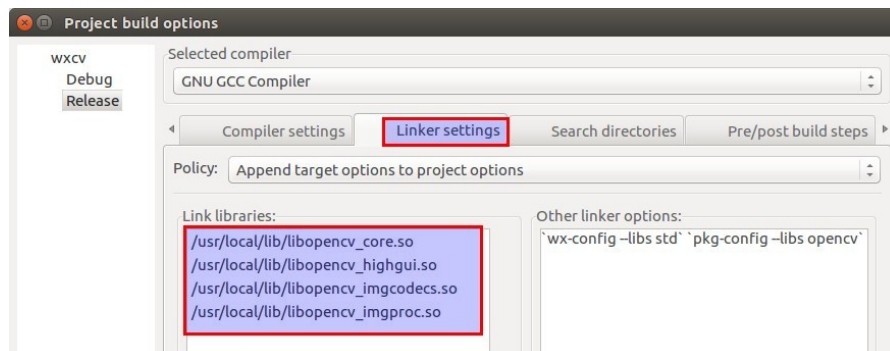
အစ ရှိတဲ့ library တွေကို လိုသလို သတ်မှတ် နိုင် ပါတယ်။

Search directories တွေနဲ့ ပတ်သက် တဲ့ အချက်အလက် တွေကို အောက်က command တွေရိုက် ကြည့်ပြီး လည်း စစ်ဆေး သိရှိ နိုင် ပါတယ်။

```
wx-config --cxxflags
pkg-config --cflags opencv
wx-config --libs std
pkg-config --libs opencv
```

ပုံ ၇.၉: Library ဖိုင်များအတွက် Search directories များသတ်မှတ်ခြင်း။



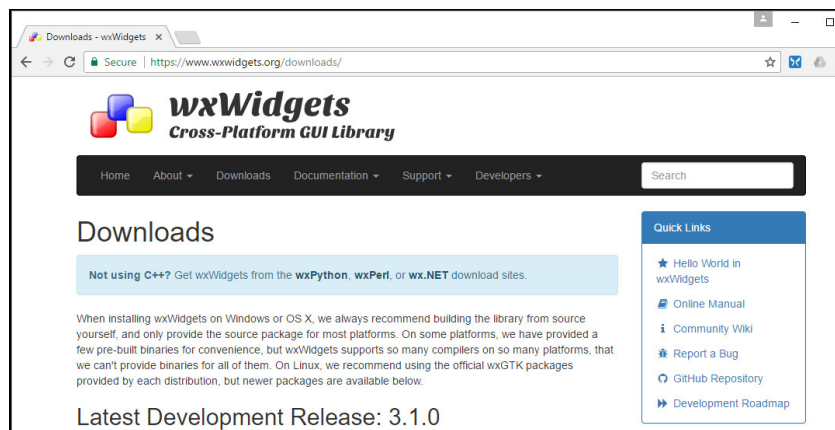
ပုံ ၇.၁၀: Link libraries များသတ်မှတ်ခြင်း။

၇.၄ Windows တွင်တပ်ဆင်ခြင်း

wxWidgets ကို Windows မှာ အသုံးပြု တဲ့အခါ IDE အမျိုးမျိုး နဲ့ တွဲသုံးနိုင် ပါတယ်။ Microsoft Visual Studio IDE ကို သုံးလို့ ရသလို၊ wxDev-C++ ၊ CodeLite [Cod17; Cod13] ၊ Code::Blocks စတဲ့ IDE တွေ ကိုလည်း ရွေးချယ် အသုံး ပြုနိုင် ပါတယ်။

၇.၄.၁ Visual Studio ဖြင့်အသုံးပြုခြင်း

wxWidgets ကို <https://www.wxwidgets.org/downloads/> ကနေ download လုပ်လိုက် ပါမယ် (ပုံ ၇.၁၁)။ ဒီ နမူနာ မှာတော့ latest stable release ဖြစ်တဲ့ 3.0.3 Windows installer ကို ရွေးလိုက် ပြီး C:/wxWidgets303/ ထဲမှာ install လုပ်လိုက် ပါမယ် (ပုံ ၇.၁၂)။



ပုံ ၇.၁၁: wxWidgets ကို free download လုပ်၍ ရယူ ခြင်း။

အဲဒီနောက် C:/wxWidgets303/docs/msw ထဲမှာ ဒါမှမဟုတ် [online](#) မှာလည်း တွေ့နိုင် တဲ့ install.txt ဆိုတဲ့ guide ထဲက Microsoft Visual C++ compilation ဆိုတဲ့ အပိုင်း ကို ဖတ်နိုင် ပါတယ် [wxW15b]။ စစ်ခြင်း WXWIN ဆိုတဲ့ environment variable ကို သတ်မှတ်ဖို့ လိုပါတယ်။ အဲ့ဒါ အတွက် command window ကို administrator အနေနဲ့ run ဖို့ start menu ကို right click နှိပ်ပြီး command prompt (admin) ကို ရွေးလိုက် ပါမယ်။ အဲ့ဒီမှာ အောက်က command သုံးပြီး WXWIN ကို ပုံ ၇.၁၃ ကလို သတ်မှတ် လိုက် ပါမယ်။

```
setx -m WXWIN C:\wxWidgets303
```

Latest Stable Release: 3.0.3

Source Code	Documentation
Windows ZIP (31 MB) Windows 7Z (16 MB) Windows Installer (47 MB) Source for Linux, OS X, etc (20 MB)	Readme Changes Online Manual Manual (HTML) ZIP (33 MB) Manual (HTML) BZIP (23 MB) Manual (CHM) (32 MB)
Binaries wxMSW DLLs for the selected compilers: <ul style="list-style-type: none"> Visual C++ 2008-2017 (more details here) TDM-GCC 4.9 and 5.1 Ubuntu / Debian Packages Fedora / openSUSE Packages	

• Released: May 2nd, 2017
 • API Stable Since: November 11th, 2013

ပုံ ၇.၁၂: wxWidgets latest stable release ကို Windows အတွက် installer ရယူခြင်း။

```

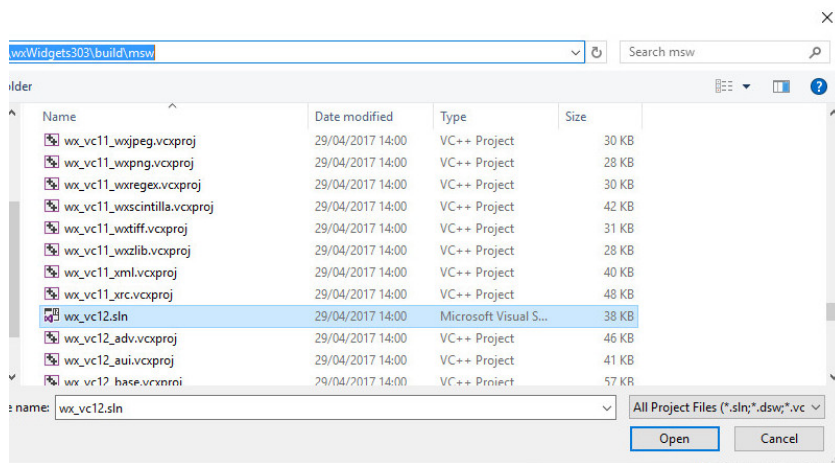
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\WINDOWS\system32> setx -m WXWIN C:\wxWidgets303

SUCCESS: Specified value was saved.
PS C:\WINDOWS\system32>
  
```

ပုံ ၇.၁၃: WXWIN environment variable သတ်မှတ်ခြင်း။

အဲဒီနောက်မှာ wxWidgets ကို build လုပ်ဖို့အတွက် အသင့်ပါရှိတဲ့ C:/wxWidgets303/build/msw ထဲက wx_vc12.sln ဆိုတဲ့ solution (ပုံ ၇.၁၄) ကို Visual Studio 2017 နဲ့ ဖွင့်လိုက်ပါမယ်။ Retarget Projects ဆိုပြီး upgrade လုပ်ဖို့ မေးလာတဲ့အခါ OK ကိုနှိပ်လိုက်ပါမယ် (ပုံ ၇.၁၅)။ ပြီးတဲ့အခါ ပုံ ၇.၁၆ မှာ ပြထားသလို configuration အမျိုးမျိုးကို ရွေးပြီး build အကြိမ်ကြိမ် လုပ်နိုင်ပါတယ်။



ပုံ ၇.၁၄: wxWidgets ကို build လုပ်ခြင်း။

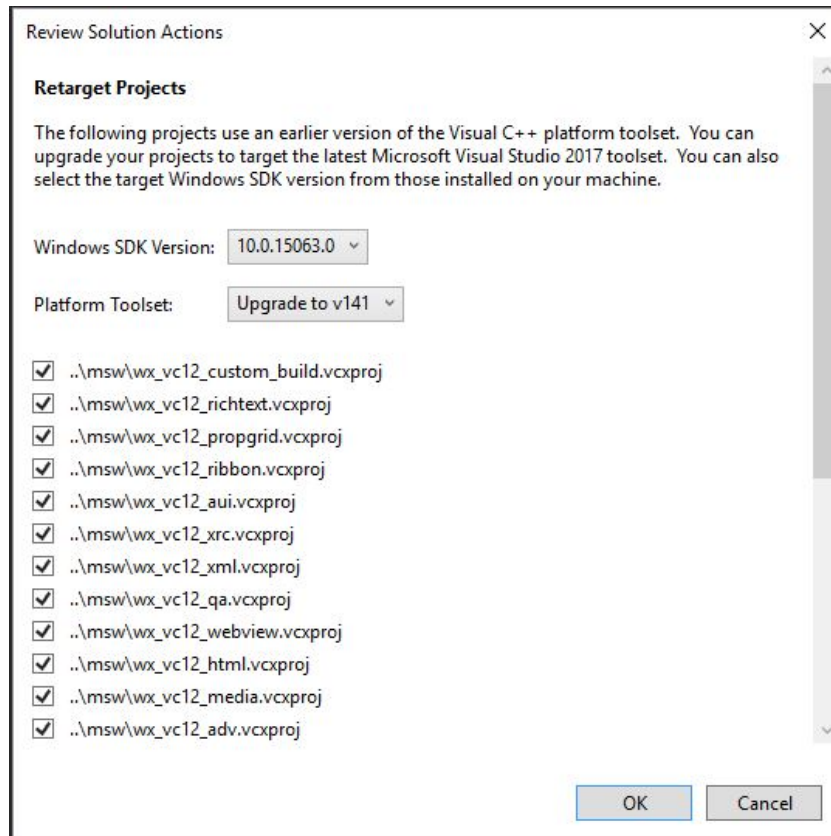
အဲဒီလို build လုပ်ပြီးတဲ့အခါ နမူနာ wxWidgets application တစ်ခုကို compile လုပ်ပြီး၊ run ကြည့်ပါမယ်။ ဒါကြောင့် C:/wxWidgets303/samples/ ထဲက minimal ဆိုတဲ့ folder ကို minimal2 ဆိုတဲ့ နာမည်နဲ့ duplicate လုပ်လိုက်ပါမယ်။

ပြီးတဲ့အခါ minimal_vc9.vcxproj ကို ဖွင့်လိုက်ပါမယ် (ပုံ ၇.၁၇)။ အဲဒါကို build လုပ်ပြီး၊ run လိုက်တဲ့အခါ ပုံ ၇.၁၈ က အတိုင်း တွေ့ရမှာဖြစ်ပါတယ်။ အဲဒီမှာ minimal.cpp ဆိုတဲ့ source ကို အမျိုးမျိုး ပြုပြင် စမ်းသပ် ကြည့်နိုင်ပါတယ်။

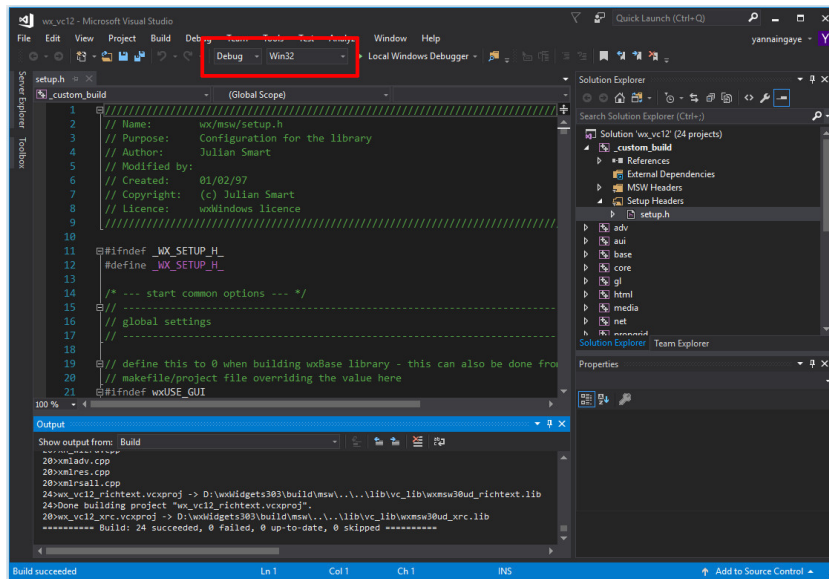
ပရိုဂရမ်ရဲ့ icon ကို ပြင်ချင်ရင် C:/wxWidgets303/samples/sample.ico ကို ကိုယ်လိုချင်တဲ့ icon နဲ့ အစားထိုးလို့ရပါတယ်။

၇.၄.၂ VS တွင် OpenCV နှင့် wxWidgets ကိုတွဲသုံးခြင်း

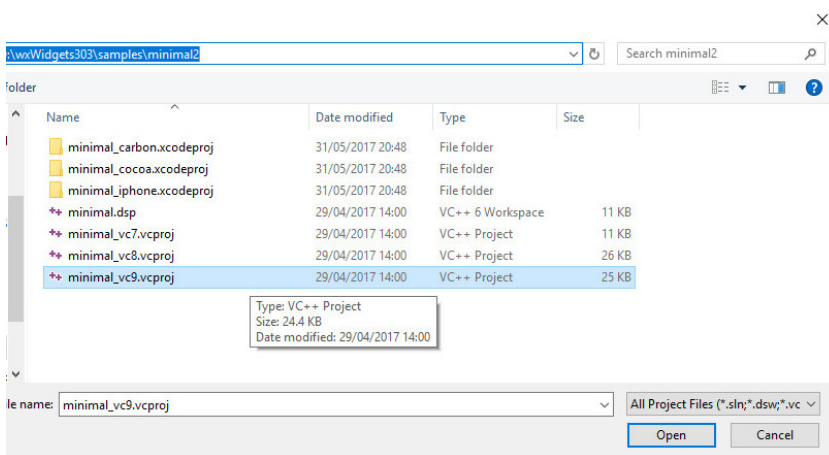
Windows ပေါ်မှာ wxWidgets နဲ့ OpenCV ကို တွဲသုံးဖို့အတွက် C:/wxWidgets303/samples/ ထဲက minimal ဆိုတဲ့ folder ကို minimalcv ဆိုတဲ့ နာမည်နဲ့ duplicate လုပ်လိုက်ပါမယ် [Eme17]။ ပြီးတဲ့အခါ minimal_vc9.vcxproj ကို ဖွင့်လိုက်ပါမယ်။ အောက်ပါ link မှာရှိတဲ့ OpenCV320.props ကို ယူပြီး



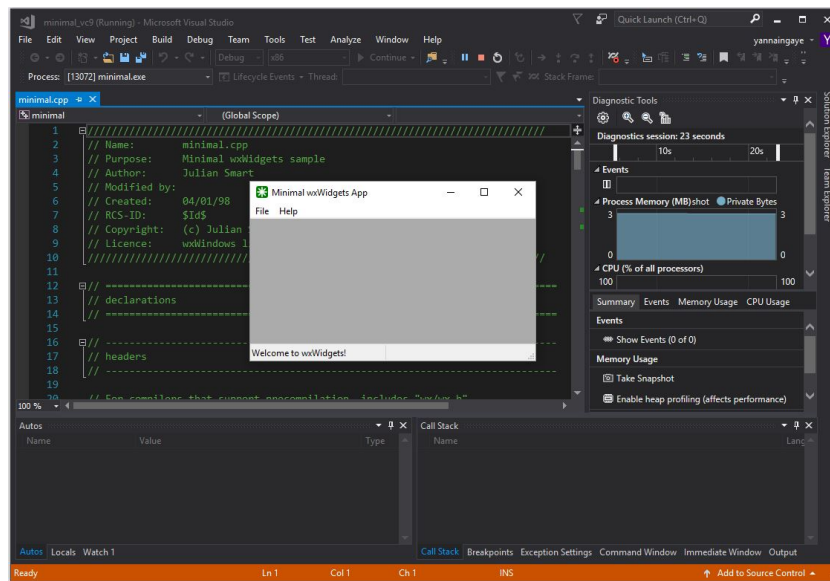
ပုံ ၇.၁၅: Project ကို upgrade လုပ်ခြင်း။



ပုံ ၇.၁၆: Configuration အမျိုးမျိုးဖြင့် build လုပ်ခြင်း။



ပုံ ၇.၁၇: နမူနာ wxWidgets project တစ်ခုကို စမ်းသပ်ခြင်း။



ပုံ ၇.၁၈: wxWidgets minimal project ကို run ခြင်း။

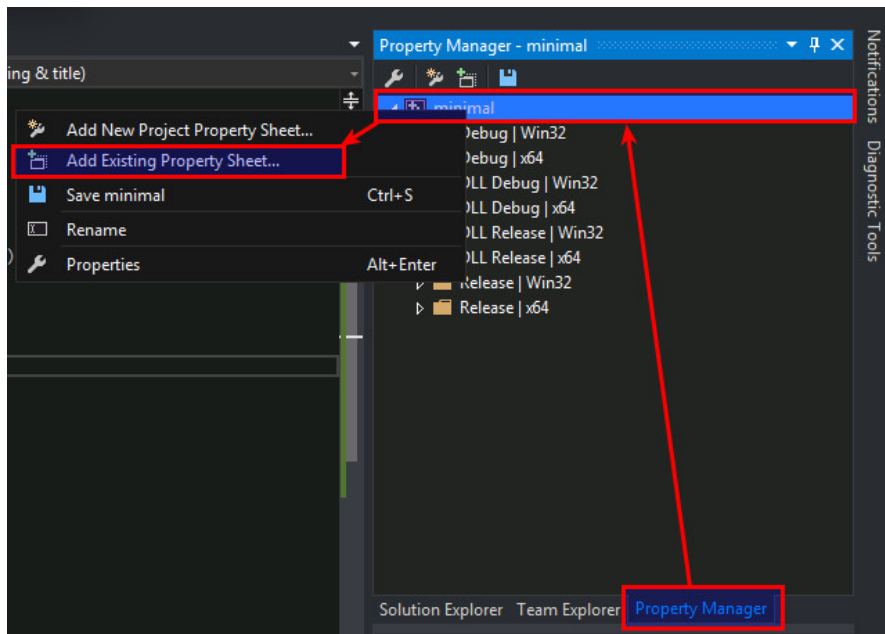
ပရောဂျက် အခန်း ထဲမှာ ကူးထည့် လိုက် ပါမယ်။

https://github.com/yan9a/OpenCV_Projects/blob/master/img_wins_vs2015_x64/img_wins/OpenCV320.props

အပေါ် menu bar မှာရှိတဲ့ View > Other Windows > Property Manager ကိုသွားဖွင့် ပြီးတဲ့ အခါ ပုံ ၇.၁၉ မှာ ပြထား သလို ညာဘက် Property window ထဲက minimal ပေါ်မှာ ညာဘက် ကလစ် နှိပ်ပြီး Add existing property sheet ကို ရွေး၊ OpenCV320.props ကို ရွေးထည့် လိုက် ပါမယ်။

အပေါ်က Solution Platforms ဆိုတဲ့ drop down list မှာ x64 ကို ရွေးထား လိုက် ပါမယ်။ Solution Explorer ထဲက Source Files ပေါ်မှာ ညာဘက် ကလစ် နှိပ်ပြီး Add Existing Item ... ကို ရွေးပြီး စာရင်း ၇.၃ မှာ ဖော်ပြ ထားတဲ့ util.h ကိုလည်း ထည့်ထား လိုက် ပါမယ်။ ပြီးတဲ့ အခါ minimal.cpp (online) ကို စာရင်း ၇.၆ မှာ ပြထား သလို ပြင်လိုက် ပါမယ်။ အဲဒီနောက် ပရိုဂရမ် ကို run လိုက်ရင် ပုံ ၇.၂၀ မှာ ပြထား သလို OpenCV နဲ့ process လုပ်ထား တဲ့ ပုံရိပ်ကို wxWidgets ရဲ့ frame ပေါ်မှာ ဖော်ပြ ထားတဲ့ output ကို တွေ့နိုင် ပါတယ်။

```
1 //File: minimalcv.cpp
2 //Description: A simple example to use OpenCV with wxWidgets
3 //Author: Yan Naing Aye
4 //Date: 2017 November 07
5 //MIT License - Copyright (c) 2017 Yan Naing Aye
```



ပုံ ၇.၁၉: OpenCV Property Sheet ကို ထည့်ခြင်း။

```

6
7 #include <wx/wx.h>
8 #include <opencv2/opencv.hpp>
9 #include "util.h"
10 #include <string>
11 using namespace std;
12 using namespace cv;
13
14 class MyFrame : public wxFrame
15 {
16     wxStaticBitmap *thiri;
17 public:
18     MyFrame(const wxString& title);
19
20 };
21 MyFrame::MyFrame(const wxString& title)
22     : wxFrame(NULL, wxID_ANY, title, wxDefaultPosition, wxSize(512, 600))
23 {

```

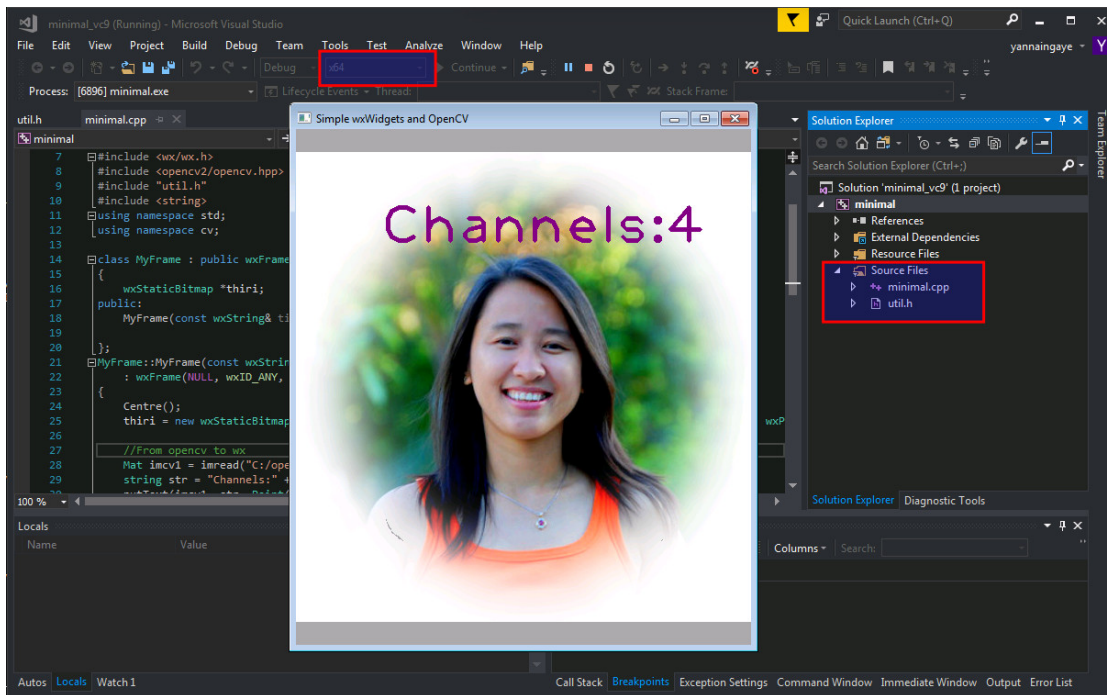


```

24     Centre();
25     thiri = new wxStaticBitmap(this, wxID_ANY, wxBitmap(wxT("C:/opencv/thiri.
        png"), wxBITMAP_TYPE_PNG), wxPoint(256, 0), wxSize(512,512));
26
27     //From opencv to wx
28     Mat imcv1 = imread("C:/opencv/thiri.png", IMREAD_UNCHANGED);
29     string str = "Channels:" + to_string(imcv1.channels());
30     putText(imcv1, str, Point(100, 100), FONT_HERSHEY_PLAIN, 4.0, CV_RGB(128,
        0, 128), 4.0);
31     wxBitmap imwx1 = wx_from_mat(imcv1);
32     thiri->SetBitmap(imwx1);
33 }
34
35 class MyApp : public wxApp
36 {
37 public:
38     virtual bool OnInit();
39 };
40
41 IMPLEMENT_APP(MyApp)
42 bool MyApp::OnInit()
43 {
44     if (!wxApp::OnInit())
45         return false;
46     wxInitAllImageHandlers();
47     MyFrame *frame = new MyFrame(wxT("Simple wxWidgets and OpenCV"));
48     frame->Show(true);
49
50     return true;
51 }

```

စာရင်း ၇.၆: minimal.cpp



ပုံ ၇.၂၀: wxWidgets နှင့် OpenCV ကို Windows ပေါ်တွင် Visual Studio ဖြင့် အသုံးပြုခြင်း။

၇.၅ Source ကုဒ်များ

ဒီ အခန်း က ပရိုဂရမ် တွေရဲ့ source ကုဒ် တွေကို အောက်က လင့်ခ် မှာ တွေ့နိုင် ပါတယ်။

- wxWidgets - <https://github.com/yan9a/wxwidgets>

အကိုးအကားများ

- [Dad10] Jive Dadson. OpenCV Image and wxImage Conversion. 2010. url: <https://stackoverflow.com/a/2241517>.
- [Eme17] Cool Emerald. Cross-platform C++ programming with wxWidgets. 2017. url: <http://coolemerald.blogspot.sg/2017/10/cross-platform-c-programming-with.html#wxvs>.

- [pkl15] pklab. Alpha from cv::mat for wxWidgets::wxImage. 2015. url: <http://answers.opencv.org/question/76540/how-to-obtain-alpha-from-cvmat-for-wxwidgetswximage/>.
- [Cod13] CodeLite. Download CodeLite bundled with MinGW and wxWidgets. 2013. url: <https://sourceforge.net/projects/codelite/files/Releases/codelite-5.0/codelite-5.0.6213-mingw4.7.1-wx2.9.4.exe/download>.
- [Cod17] CodeLite Wiki. Compiling wxWidgets with MinGW. 2017. url: <http://codelite.org/Developers/BuildingWxWidgetsWin>.
- [Qt17] Qt. Licensing - Before you begin, make the right license choice. 2017. url: <https://www1.qt.io/licensing/>.
- [Zet13] ZetCode. wxWidgets tutorial. 2013. url: <http://zetcode.com/gui/wxwidgets/>.
- [wxW12] wxWiki. WxWidgets Compared To Other Toolkits. 2012. url: https://wiki.wxwidgets.org/WxWidgets_Compared_To_Other_Toolkits.
- [wxW14] wxWiki. Compiling and getting started. 2014. url: https://wiki.wxwidgets.org/Compiling_and_getting_started.
- [wxW15a] wxWiki. Bindings. 2015. url: <https://wiki.wxwidgets.org/Bindings>.
- [wxW15b] wxWiki. Microsoft Visual C++ Guide. 2015. url: https://wiki.wxwidgets.org/Microsoft_Visual_C%2B%2B_Guide.
- [wxW98] wxWidgets. wxWindows Library Licence. 1998. url: <https://www.wxwidgets.org/about/licence/>.

Appendix A

နောက်ဆက်တွဲ ရှင်းလင်းချက်

၁.၁ စကားလုံးဖွင့်ဆိုချက်များ

OpenCV Open source computer vision

vs15 Visual studio 2017

x86 A family of backward compatible 32 bit instruction set
architectures based on the Intel 80386 CPU

x64 The 64-bit version of the x86 instruction set

DFT Discrete Fourier Transform

ကြိမ်နှုန်းနယ် Frequency domain

တိုင် Column

တည်ရာနယ် Spatial domain

တန်း Row

ပုံရိပ်ပြုစင်ခြင်း..... Image processing

ပုံလုံးဖော်လုပ်ဆောင်ချက်များ... Morphological operation

ပြကွက်..... Frame

ပြနှုန်း..... Frame rate

ဖယ်တာ..... Filter

အရောင်နယ်..... Color space

ဖြည့်စွက်အာရုံရိပ်..... Augmented reality

၁.၂ ThresholdTypes

ThresholdTypes တစ်ခုစီ အတွက် သင်္ချာညီမျှခြင်း ဖြင့် ဖော်ပြချက် များကို ဇယား ၁.၁ တွင် တွေ့နိုင်သည်။

ဇယား ၁.၁: Threshold အမျိုးအစားများ၏ ညီမျှခြင်းများ။

THRESH_BINARY	$\text{dst}(x, y) = \begin{cases} \text{maxval}, & \text{if } \text{src}(x, y) > \text{thresh} \\ 0, & \text{otherwise} \end{cases}$
THRESH_BINARY_INV	$\text{dst}(x, y) = \begin{cases} 0, & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{maxval}, & \text{otherwise} \end{cases}$
THRESH_TRUNC	$\text{dst}(x, y) = \begin{cases} \text{thresh}, & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{src}(x, y), & \text{otherwise} \end{cases}$
THRESH_TOZERO	$\text{dst}(x, y) = \begin{cases} \text{src}(x, y), & \text{if } \text{src}(x, y) > \text{thresh} \\ 0, & \text{otherwise} \end{cases}$
THRESH_TOZERO_INV	$\text{dst}(x, y) = \begin{cases} 0, & \text{if } \text{src}(x, y) > \text{thresh} \\ \text{src}(x, y), & \text{otherwise} \end{cases}$
THRESH_MASK	
THRESH_OTSU	flag ဖြစ်ပါတယ်။ Otsu algorithm ကိုသုံးပြီး optimal threshold တန်ဖိုးကို ရွေးဖို့ပါ။
THRESH_TRIANGLE	flag ဖြစ်ပါတယ်။ Triangle algorithm ကိုသုံးပြီး optimal threshold တန်ဖိုးကို ရွေးဖို့ပါ။