**1. Qroilib 시작**

qroilib는 처음 ViewMainPage instance를 만드는것으로 시작합니다.

ViewMainPage\* mViewMainPage;

mViewMainPage = new ViewMainPage(mViewStackedWidget);

ViewMainPage를 만든다음에 ROI Editor기능을 이용하고자한다면.아래와 같이

paramTable을 만들고 setParamTable()로 등록을 해야합니다.

extern Qroilib::ParamTable ROIDSHARED\_EXPORT paramTable[];

for (int i=0; i<2; i++)

{

DocumentView\* view = d->mViewMainPage->view(i);

if (view)

{

view->setParamTable(paramTable);

view->bMultiView = true;

connect(view, &Qroilib::DocumentView::finishNewRoiObject,

this, &MainWindow::finishNewRoiObject);

}

}

Qroilib::ParamTable paramTable[] = {

\_Inspect\_Patt\_Start, CParam(\_ProcessValue1, (""), \_IntValue, ("0")), // do not delete.

\_Patt1, CParam(\_ProcessValue1, ("Pattern matching rate"), \_DoubleValue, ("60")),

\_Patt1, CParam(\_ProcessValue1, ("Low Threshold"), \_IntValue, ("120")),

\_Patt1, CParam(\_ProcessValue1, ("High Threshold"), \_IntValue, ("255")),

\_Patt1, CParam(\_ProcessValue1, ("Noise out 1"), \_IntValue, ("-2")), // -1 : morphology Open

\_Patt1, CParam(\_ProcessValue1, ("Noise out 2"), \_IntValue, ("2")), // 1 : morphology Close

\_Patt1, CParam(\_ProcessValue1, ("Shape matching rate"), \_DoubleValue, ("70")),

\_Inspect\_Roi\_Start, CParam(\_ProcessValue1, (""), \_IntValue, ("0")), // do not delete.

\_Roi1, CParam(\_ProcessValue1, ("Direction"), \_ComboValue, ("0"), ("L2R,R2L,T2B,B2T,Center")),

\_Roi1, CParam(\_ProcessValue1, ("Detect method"), \_ComboValue, ("0"), ("Average, First")),

\_Roi1, CParam(\_ProcessValue1, ("Polarity"), \_ComboValue, ("0"), ("W2B, B2W")),

\_Roi1, CParam(\_ProcessValue1, ("Low Threshold"), \_IntValue, ("200")),

\_Roi1, CParam(\_ProcessValue1, ("High Threshold"), \_IntValue, ("255")),

\_Roi1, CParam(\_ProcessValue1, ("Noise out 1"), \_IntValue, ("-2")),

\_Roi1, CParam(\_ProcessValue1, ("Noise out 2"), \_IntValue, ("2")),

\_Roi1, CParam(\_ProcessValue1, ("Ramp width"), \_IntValue, ("6")),

\_Roi1, CParam(\_ProcessValue1, ("Detect method"), \_ComboValue, ("0"), ("Average, First")),

\_Inspect\_Point\_Start, CParam(\_ProcessValue1, (""), \_IntValue, ("0")), // do not delete.

\_Point1, CParam(\_ProcessValue1, ("Mark color"), \_IntValue, ("128")), // not yet implement.

\_Inspect\_Type\_End, CParam(\_FilterValue, (""), \_IntValue, ("")), // do not delete.

};

finishNewRoiObject()는 ROI를 생성한후 호출된다.

bMultiView를 true로 두면 초기 화면에 ROI 정보를 보여주지 않습니다.

**2. ViewMainPage()**

ViewMainPage를 생성할때 아래와 같이 Single 또는 Multi document를 생성합니다.

Single일때에는 bMultiView를 false, Multi일때에는 true로 합니다.

const QUrl ch1 = QUrl("file:channel1");

const QUrl ch2 = QUrl("file:channel2");

QList<QUrl> urls;

urls.append(ch1);

if (gCfg.m\_nCamNumber > 1)

urls.append(ch2);

openUrls(urls, ch1);

"[file:channel1](file:///channel1)” 과 같이 Qurl name인 중복되지 않는 이름을 부여합니다.

**4. ROI Tools**

v->selectTool(v->actSelectTool);

v->selectTool(v->actCreatePoint);

v->selectTool(v->actCreateRectangle);

v->selectTool(v->actCreatePattern);

ROI 선택 도구, Point ROI 생성도구, Rectangle ROI 생성도구, Pattern ROI 생성도구를 선택합니다.

**5. Read Image**

DocumentView\* v = currentView();

QImage img;

v->mRoi->setWidth(img.width());

v->mRoi->setHeight(img.height());

v->document()->setImageInternal(img);

v->imageView()->updateBuffer();

이미지 파일을 로드해서 Display합니다.

**6. Save Image**

DocumentView\* v = currentView();

const QImage \*pimg = v->image();

pimg->save(fileName);

현재 보여지는 이미지를 파일로 Write합니다.

아래 사이트에 qroilib를 이용한 이미지 viewer program이 있습니다.

<https://github.com/JonglockYoon/qroilib/tree/master/qroiviewer>

**7. Capture Image**

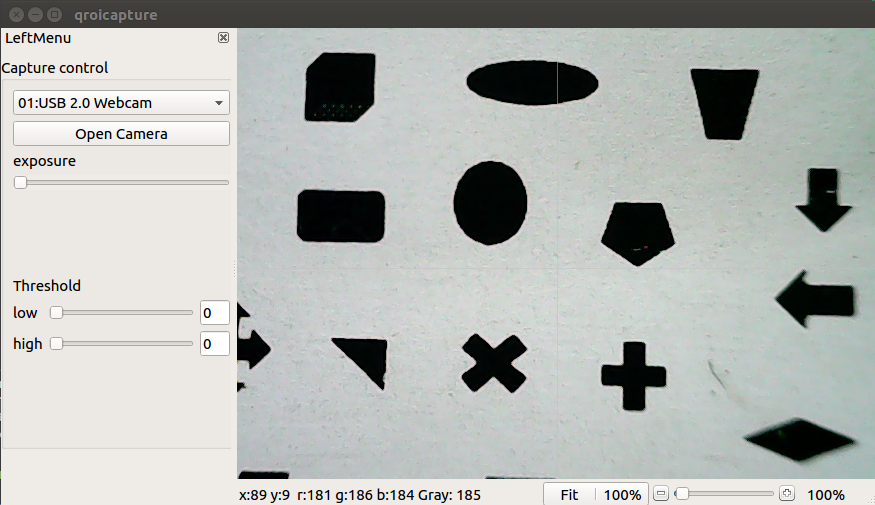
LeftDock \*mLeftDock;

mLeftDock = new LeftDock(tr("Menu"), this);

mLeftDock->setFeatures(QDockWidget::DockWidgetClosable);

addDockWidget(Qt::LeftDockWidgetArea, mLeftDock);

d->viewMenu->addAction(mLeftDock->toggleViewAction());

LeftDock 메뉴가 생성되면서 ComboBox에 현재 연결되어 있는 video device를 선택할수 있게 됩니다.

ViewMainPage\* pView = theMainWindow->viewMainPage();

str = ui->comboBoxCam1Port->currentText();

int ncam1 = str.mid(0,2).toInt();

pView->OpenCam(0, ncam1-1);

“Open Camera” 버턴을 누르면 현재 선택된 Video Device를 Open합니다.

void ViewMainPage::OpenCam(int seq, int nCam)

{

myCamCapture = new CamCapture(d->mDocumentViews[seq], seq, nCam);

QObject::connect(myCamCapture, SIGNAL(processedImage(const QImage&, int)),

this, SLOT(updatePlayerUI(const QImage&, int)));

if (myCamCapture->OpenCam() == true)

myCamCapture->Play();

}

OpenCam()함수에서 cv::VideoCapture 기능을 이용하여 카메라 Device를 Open하게 되고

capture된 이미지들은 updatePlayerUI()함수를 통하여 화면에 Display됩니다.

아래 사이트에 qroilib를 이용한 카메라 image capture program이 있습니다.

<https://github.com/JonglockYoon/qroilib/tree/master/qroicapture>

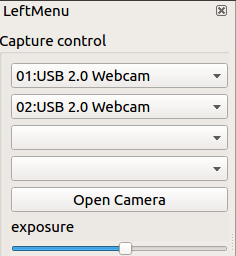
**8. Multi Capture Image**

앞에서 Single Camera Capture Program을 보았으며,

여기서는 4개의 Camera를 동시에 Capture할수 있는 Program을 보고자 합니다.

Qroilib library는 8개의 multi view를 지원하므로 화면을 최대 8분할하여 사용하수 있습니다.

여기서는 화면을 4분할 해서 4개의 카메라를 실시간 capture해서 preview할수 있는 program흐름을 보고자 합니다.



Left Dock화면을보면 카메라 선택 Combo Box가 4개로 늘어 났습니다.

연결된 카메라들을 선택후 “Open Camera”버턴을 클릭하면 선택한 카메라 대상으로

pView->OpenCam(...) 함수가 호출됩니다.

void ViewMainPage::**OpenCam**(int seq, int nCam)

{

myCamCapture[seq] = new CamCapture(d->mDocumentViews[seq], seq, nCam);

QObject::connect(myCamCapture[seq], SIGNAL(processedImage(const QImage&, int)),

this, SLOT(updatePlayerUI(const QImage&, int)));

if (myCamCapture[seq]->OpenCam() == true)

myCamCapture[seq]->Play();

}

OpenCam()함수는 카메라 한 개당 CamCapture Instance를 생성시키고 생성된 CamCapture는

각 연결된 카메라에서 이미지를 획득하고 processedImage Signal을 보내게됩니다.

**updatePlayerUI**(const QImage& img, int seq)의 seq는 view의 순서번호가 들어 있습니다.

이것으로 몇번째 view에 Display할것인지를 결정합니다.

void ViewMainPage::updatePlayerUI(const QImage& img, int seq)

{

Qroilib::DocumentView\* v = view(seq);

v->document()->setImageInternal(img);

v->imageView()->updateBuffer();

}

Tip) USB Camera들은 USB대역폭을 많이 사용하므로 USB HUB를 통하여 여러대의 카메라를 연결하면 대역폭이 모자라 Capture가 되지 않습니다.

PC의 USB Port에 직접 연결해서 사용해야합니다.

Raspberry Pi 3경우는 4개의 USB Port가 있지만 CPU Processing성능이 낮으므로 30만화소 Webcam Full Speed로 2개를 사용할수 없습니다.

2개이상의 Camera를 사용할려고 한다면 대역폭 조정이 되는 Camera를 선택 하셔야합니다.

대역폭조정)

CV\_CAP\_PROP\_FPS set을 이용해서 FPS(Frame Per Sec)설정이 되었는지 확인합니다.

대역폭 조정이 되지 않는 Camera도 있으니 get으로 확인절차가 필요합니다.

capture.set(CV\_CAP\_PROP\_FPS, val) 함수는 시스템 또는 Camera에 따라 지원되지 않는경우가 있습니다.

qroicapture program으로 확인할수 있습니다.