China Daheng Group, Inc. Beijing Image Vision Technology Branch

# How to use Mercury Family Cameras with OpenCV on Windows OS

Version: V1.0.31

Date: 2018-03-19

DAHENG IMAGING 大恒图像

Web: http://www.daheng-imaging.com

Sales Email: isales@daheng-imaging.com

Sales Tel: +86 10 8282 8878

Support Email: isupport@daheng-imaging.com

# Contents

# 1. Introduction

OpenCV(Open Source Computer Vision) is a library of programming functions manly aimed at real-time computer vision. OpenCV is written in C++ and its primary interface is in C++ , but it sill remains a less comprehensive though extensive older C interface.There are bindings in PYThon , Java and MATLAB/OCTAVE.

This application note provides how to use OpenCV in combination with Daheng Mercury Family Cameras in Microsoft Visual Studio on Windows OS.

This application note assumes you are using the latest version of mercury cameras SDK.

You can download the mercury cameras SDK from the website : www.daheng-imaging.com.

# 2. Steps to run Mercury cameras with OpenCV

The following steps shows how to complete the function

1. Acquire images with Mercury SDK sample
2. Initialize the OpenCV Image( Mat class object).
3. Copy the image data from Mercury camera buffer to OpenCV Image(Mat.data);
4. Save Image with OpenCV function

# 3. Sample Code to Run Mercury Mono Camera with OpenCV

## 3.1 Acquire images with Mercury API functions.

We recommended use the sample named “GxSingleCamMono” for mono cameras. You can find this project under the “Samples” folder . You can find the “samples” folder as figure 1 .The following code is base on “GxSingleCamMono “ sample.

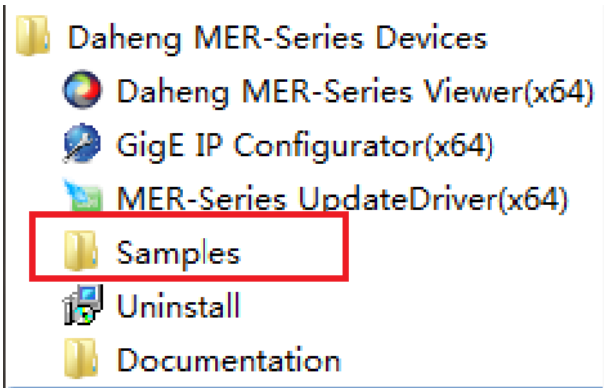


Figure 1 ： The location of examples

## 3.2 Initialize the Mat class object.

a) Define Mat class object in GxSingleCamMonoDLG.h file. The code is as follows;

Code ： Mat m_pImage

b) Initialize the Mat class object in the GetDeviceParam() function from the GxSingleCamMonoDLG.cpp file.The code is as follows;

m_pImage.create(m_nImageHeight,m_nImageWidth, CV_8UC1);

You can see the code as figure 2.

```
//----------------------------------------------------------------------------------
/**
\brief  获取设备的宽高等属性信息

\return GX_STATUS_SUCCESS:全部获取成功，其它状态码:未成功获取全部
*/
//----------------------------------------------------------------------------------
GX_STATUS CGxSingleCamMonoDlg::GetDeviceParam()
{
    GX_STATUS emStatus       = GX_STATUS_ERROR;
    bool      bIsImplemented = false;

    // 获取图像大小
    emStatus = GXGetInt(m_hDevice, GX_INT_PAYLOAD_SIZE, &m_nPayLoadSize);
    VERIFY_STATUS_RET(emStatus);

    // 获取宽度
    emStatus = GXGetInt(m_hDevice, GX_INT_WIDTH, &m_nImageWidth);
    VERIFY_STATUS_RET(emStatus);

    // 获取高度
    emStatus = GXGetInt(m_hDevice, GX_INT_HEIGHT, &m_nImageHeight);

        //
    m_pImage.create(m_nImageHeight,m_nImageWidth, CV_8UC1);
    return emStatus;
}
```

Figure 2 Mat initialization of mono images

## 3.3 Convert camera Image to OpenCV Image

In the callback function named “OnFrameCallbackFun”, copy the image data to Mat class object through memcp() function. The specific code is as follows:

memcpy(m_pImage.data, pDlg->m_pImageBuffer, nImageHeight * nImageWidth);

```
//----------------------------------------------------------------------------------
/**
\brief   采集图像回调函数
\param   pFrame      回调参数

\return  无
*/
//----------------------------------------------------------------------------------
void __stdcall CGxSingleCamMonoDlg::OnFrameCallbackFun(GX_FRAME_CALLBACK_PARAM* pFrame)
{
    CGxSingleCamMonoDlg *pDlg = (CGxSingleCamMonoDlg*)(pFrame->pUserParam);
    int nImageHeight = (int)pDlg->m_nImageHeight;
    int nImageWidth  = (int)pDlg->m_nImageWidth;

    if (pFrame->status == 0)
    {
        memcpy(pDlg->m_pBufferRaw,pFrame->pImgBuf,pFrame->nImgSize);

        // 黑白相机需要翻转数据后显示
        for(int i =0;i <nImageHeight;i++)
        {
            memcpy(pDlg->m_pImageBuffer+i*nImageWidth,
                pDlg->m_pBufferRaw+(nImageHeight-i-1)*nImageWidth,(size_t)nImageWidth);
        }

        pDlg->DrawImg();
        memcpy(m_pImage.data, pDlg->m_pImageBuffer, nImageHeight * nImageWidth);
        // 图像保存处理
        if (pDlg->m_bIsSaveImg)
        {
            pDlg->SaveImage();
        }
    }
}
```

Figure 3 Mono image data copy

## 3.4 Save the OpenCV Image to the disk

Finally we get the OpenCV Image ,which is stored in the mat.data . Then you can use this image for display ,saving ,and recording use OpenCV function . We take the image saving for example .

```
//----------------------------------------------------------------------------------
void __stdcall CGxSingleCamMonoDlg::OnFrameCallbackFun(GX_FRAME_CALLBACK_PARAM* pFrame)
{
    CGxSingleCamMonoDlg *pDlg = (CGxSingleCamMonoDlg*)(pFrame->pUserParam);
    int nImageHeight = (int)pDlg->m_nImageHeight;
    int nImageWidth  = (int)pDlg->m_nImageWidth;
    if (pFrame->status == 0)
    {
        memcpy(pDlg->m_pBufferRaw,pFrame->pImgBuf,pFrame->nImgSize);

        // 黑白相机需要翻转数据后显示
        for(int i =0;i <nImageHeight;i++)
        {
            memcpy(pDlg->m_pImageBuffer+i*nImageWidth,
                pDlg->m_pBufferRaw+(nImageHeight-i-1)*nImageWidth,(size_t)nImageWidth);
        }

        pDlg->DrawImg();

        memccpy(m_pImage.data,pDlg->m_pImageBuffer,nImageHeight*nImageWidth);

        imwrite("./test1.bmp", m_pImage);
```

Figure 4 Mono Image save to the disk

# 4. Sample Code to Run Mercury Color Camera with OpenCV

## 4.1 Acquire images with MER API functions.

We recommended use the sample “GxSingleCamColor” for mono cameras. You can find this project under the “Samples” folder . You can find the “samples” folder as figure 1 .The following code is base on “GxSingleCamColor “ sample.

## 4.2 Initialize the Mat class object.

c) Define Mat class object(Mat m_pImage) in GxSingleCamColorDLG.h file;

d) Initialize the Mat class object in the GetDeviceParam() function.The specific code is as follows;

m_pImage.create(m_nImageHeight,m_nImageWidth, CV_8UC3);

```
VERIFY_STATUS_RET(emStatus);

// 获取宽度
emStatus = GXGetInt(m_hDevice, GX_INT_WIDTH, &m_nImageWidth);
VERIFY_STATUS_RET(emStatus);

//判断相机是否支持bayer格式
emStatus = GXIsImplemented(m_hDevice, GX_ENUM_PIXEL_COLOR_FILTER, &m_bColorFilter);
VERIFY_STATUS_RET(emStatus);
if (m_bColorFilter)
{
    //获取自动白平衡枚举值
    emStatus = GXGetEnum(m_hDevice, GX_ENUM_BALANCE_WHITE_AUTO, &m_nAutoWhiteBalance);
    VERIFY_STATUS_RET(emStatus);

    //获取图像Bayer格式
    emStatus = GXGetEnum(m_hDevice, GX_ENUM_PIXEL_COLOR_FILTER, &m_nPixelColorFilter);
    VERIFY_STATUS_RET(emStatus);
}

//判断相机是否支持白平衡通道
emStatus = GXIsImplemented(m_hDevice, GX_ENUM_BALANCE_RATIO_SELECTOR, &m_bBalanceRatioSelector);
VERIFY_STATUS_RET(emStatus);

// 获取高度
emStatus = GXGetInt(m_hDevice, GX_INT_HEIGHT, &m_nImageHeight);

m_pImage.create(m_nImageHeight,m_nImageWidth, CV_8UC3);
return emStatus;
```

Figure 5 Color image Mat initialization

## 4.3 Copy the image data from acquisition buffer to Mat.data

In the callback function of the camera, copy the image data to Mat class object through memcp() function. The specific code is as follows:

memcpy(m_pImage.data, pDlg->m_pImageBuffer, nImageHeight * nImageWidth*3);

```
//------------------------------------------------------------------------------
/**
\brief   回调函数
\param   pFrame         回调参数

\return  无
*/
//------------------------------------------------------------------------------
void __stdcall CGxSingleCamColorDlg::OnFrameCallbackFun(GX_FRAME_CALLBACK_PARAM* pFrame)
{
    CGxSingleCamColorDlg *pDlg = (CGxSingleCamColorDlg*)(pFrame->pUserParam);

    if (pFrame->status == 0)
    {
        memcpy(pDlg->m_pBufferRaw, pFrame->pImgBuf, pFrame->nImgSize);

        // RGB转换
        DxRaw8toRGB24( pDlg->m_pBufferRaw
                     , pDlg->m_pBufferRGB
                     , (VxUint32)(pDlg->m_nImageWidth)
                     , (VxUint32)(pDlg->m_nImageHeight)
                     , RAW2RGB_NEIGHBOUR
                     , DX_PIXEL_COLOR_FILTER(pDlg->m_nPixelColorFilter)
                     , TRUE);
        memcpy(m_pImage.data, pDlg->m_pBufferRGB, nImageHeight * nImageWidth*3);

        // 图像质量提升
```

Figure 6 Copy of color image data

## 4.4 Save the OpenCV Image to the disk

Finally we get the OpenCV Image ,which is stored in the mat.data . Then you can use this image for display ,saving ,and recording use OpenCV function . We take the image saving for example .

```
//------------------------------------------------------------------------------
void __stdcall CGxSingleCamColorDlg::OnFrameCallbackFun(GX_FRAME_CALLBACK_PARAM* pFrame)
{
    CGxSingleCamColorDlg *pDlg = (CGxSingleCamColorDlg*)(pFrame->pUserParam);
    if (pFrame->status == 0)
    {
        memcpy(pDlg->m_pBufferRaw, pFrame->pImgBuf, pFrame->nImgSize);

        // RGB转换
        DxRaw8toRGB24( pDlg->m_pBufferRaw
                     , pDlg->m_pBufferRGB
                     , (VxUint32)(pDlg->m_nImageWidth)
                     , (VxUint32)(pDlg->m_nImageHeight)
                     , RAW2RGB_NEIGHBOUR
                     , DX_PIXEL_COLOR_FILTER(pDlg->m_nPixelColorFilter)
                     , TRUE);
        memcpy(m_pImage.data, pDlg->m_pBufferRGB, nImageHeight * nImageWidth*3);
        // 图像质量提升
        if (pDlg->m_bIsImproveImg)
        {
            //提升图像质量
            DxImageImprovment(pDlg->m_pBufferRGB
                            , pDlg->m_pBufferRGB
                            , (VxUint32)(pDlg->m_nImageWidth)
                            , (VxUint32)(pDlg->m_nImageHeight)
                            , pDlg->m_nColorCorrection
                            , pDlg->m_pContrastLut
                            , pDlg->m_pGammaLut);
        }
        // 显示图像
        pDlg->DrawImg();
        memccpy(m_pImage.data,pDlg->m_pImageBuffer,nImageHeight*nImageWidth*3);

        imwrite("./test1.bmp", m_pImage);
```

Figure 7 Color Image save to the disk

# 5. Revision History

| No. | Version | Changes | Data |
|---|---|---|---|
| 1 | V1.0.0 | Initial release | 2018-0413 |