DUO OpenCV Samples

Overview

This article is a review of the `C++` OpenCV samples that ship with the DUO SDK. This will be updated as new features are added or specifications updated to the DUO API. In the `Developers/Samples` folder within the SDK download you can find the latest version of these samples with a `cmake` project to generate specific IDE projects (Visual Studio/Qt Creator).

Prerequisites

Before reading this guide it is recommended to review our API and SDK Docs to get an understanding of the design, common practices and usage. With DUO devices you will always receive a "frame" which contains all relevant sensor data. The following examples showcase the usage of the DUOFrame structure while using different device functions. Also make sure you have the latest SDK download and OpenCV.

- DUOLib.lib & DUOLib dynamic library that match your compiler and architecture (x86/x64)
- OpenCV v2.4.7+
- CMake 2.8+

Methods/Structures

Here are some of the common methods used through-out the examples:

- `EnumerateResolutions` - Lists available resolutions.
    - `DUOResolutionInfo` - Structure used for resolution information.
- `DUOFrameCallback` - Callback on frame update.
- `OpenDUO` - Opens a new connection to the device.
- `StartDUO` - Initializes capture on the device.
- `StopDUO` - Un-Initializes capture on the device.
- `CloseDUO` - Closes the connection to the device.

And variables/structures:

- `PDUOFrame` - Structure containing device sensor data.
    - `pFrameData->timeStamp` - Frame time stamp in 100µs increments.

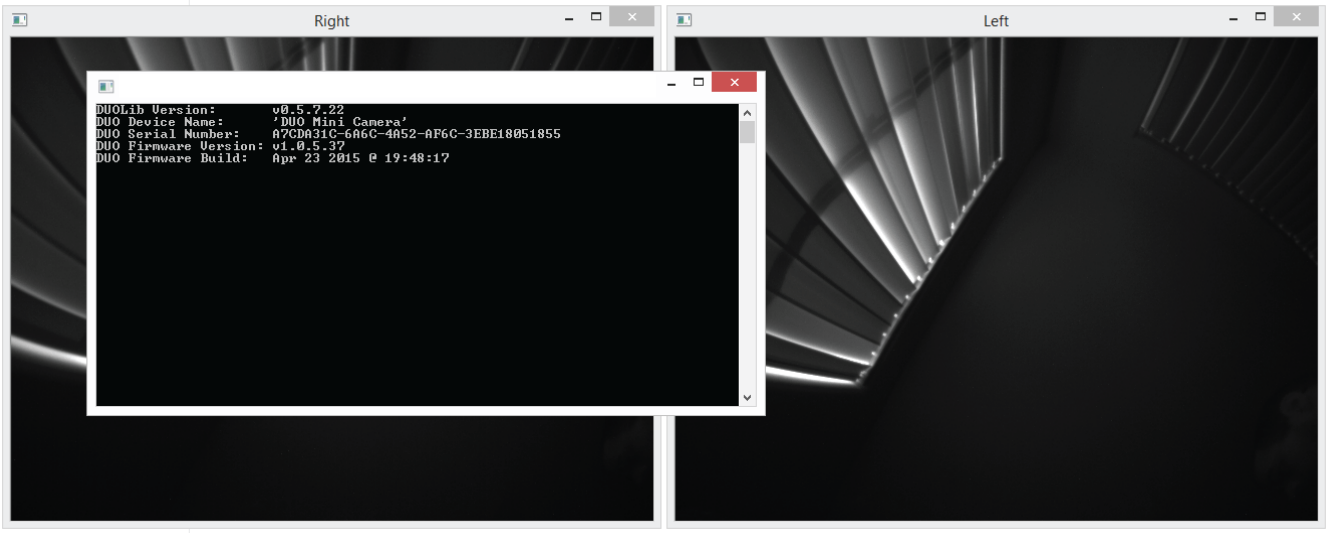
Include/Linkage

The including the DUOLib and OpenCV libraries (which may vary dependant on compiler).

```
#include "DUOLib.h"
#include "opencv2/opencv.hpp"
using namespace cv;
```

Sample 01

Display frames using polling mechanism (OpenCV)



Demonstrates polling mechanism for capturing and display of frames using the OpenCV library (opencv.org).

Utilizing all highlighted methods/structures from Samples 1-4 and the polling technique from Sample 5.

We use these specific OpenCV functions/structures:

- `IplImage` - Contains the image data we recieve from the cameras.
- `cvNamedWindow` - Creates a named window with highgui.
- `cvReleaseImageHeader` - Releases image upon closing of window.
- `cvShowImage` - Displays image on prior created window.

Sample.h

```
#ifndef SAMPLE_H
#define SAMPLE_H

// Platform Specific
...


// Include DUO API header file
#include "DUOLib.h"

#include "opencv2/opencv.hpp"
using namespace cv;

// Some global variables

static DUOInstance _duo = NULL;
static PDUOFrame _pFrameData = NULL;

// Platform Specific
...


// One and only duo callback function
// It sets the current frame data and signals that the new frame data is ready
static void CALLBACK DUOCallback(const PDUOFrame pFrameData, void *pUserData)
{
    _pFrameData = pFrameData;
    SetEvent(_evFrame);
}

// Opens, sets current image format and fps and starts capturing
static bool OpenDUOCamera(int width, int height, float fps)
{
    if(_duo != NULL)
    {
        StopDUO(_duo);
        CloseDUO(_duo);
        _duo = NULL;
    }

    // Find optimal binning parameters for given (width, height)
    // This maximizes sensor imaging area for given resolution
    int binning = DUO_BIN_NONE;
    if(width <= 752/2)
        binning += DUO_BIN_HORIZONTAL2;
    if(height <= 480/4)
        binning += DUO_BIN_VERTICAL4;
    else if(height <= 480/2)
        binning += DUO_BIN_VERTICAL2;

    // Check if we support given resolution (width, height, binning, fps)
```

```
    DUOResolutionInfo ri;
    if(!EnumerateResolutions(&ri, 1, width, height, binning, fps))
        return 0;

    if(!OpenDUO(&_duo))
        return 0;

    char tmp[260];
    // Get and print some DUO parameter values

    GetDUODeviceName(_duo,tmp);
    printf("DUO Device Name:      '%s'\n", tmp);

    GetDUOSerialNumber(_duo, tmp);
    printf("DUO Serial Number:    %s\n", tmp);

    GetDUOFirmwareVersion(_duo, tmp);
    printf("DUO Firmware Version: v%s\n", tmp);

    GetDUOFirmwareBuild(_duo, tmp);
    printf("DUO Firmware Build:   %s\n", tmp);

    // Set selected resolution
    SetDUOResolutionInfo(_duo, ri);

    // Start capture
    if(!StartDUO(_duo, DUOCallback, NULL))
        return 0;
    return true;
}

// Waits until the new DUO frame is ready and returns it
static PDUOFrame GetDUOFrame()
{
    if(_duo == NULL)
        return 0;
    if(WaitForSingleObject(_evFrame, 1000) == WAIT_OBJECT_0)
        return _pFrameData;
    else
        return NULL;
}

// Stops capture and closes the camera
static void CloseDUOCamera()
{
    if(_duo == NULL)
        return;
    // Stop capture
    StopDUO(_duo);
    // Close DUO
    CloseDUO(_duo);
    _duo = NULL;
}
```

```
static void SetExposure(float value)
{
    if(_duo == NULL)
        return;
    SetDUOExposure(_duo, value);
}

static void SetGain(float value)
{
    if(_duo == NULL)
        return;
    SetDUOGain(_duo, value);
}

static void SetLed(float value)
{
    if(_duo == NULL)
        return;
    SetDUOLedPWM(_duo, value);
}

#endif // SAMPLE_H
```

Sample.cpp

As for the implementation we go through the following steps:

- **Step 1** - Create and Open DUO Instance
- **Step 2** - Create OpenCV Windows and IplImage variables
- **Step 3** - Loop through DUO frames and display using cvShowImage
- **Step 4** - Release and cleanup images and devices

```
#include "Sample.h"

#define WIDTH    640
#define HEIGHT    480
#define FPS        30

int main(int argc, char* argv[])
{
    printf("DUOLib Version:       v%s\n", GetLibVersion());

    // Open DUO camera and start capturing
    if(!OpenDUOCamera(WIDTH, HEIGHT, FPS))
    {
        printf("Could not open DUO camera\n");
        return 0;
```

```
    }
    // Create OpenCV windows
    cvNamedWindow("Left");
    cvNamedWindow("Right");

    // Set exposure and LED brightness
    SetExposure(50);
    SetLed(25);

    // Create image headers for left & right frames
    IplImage *left = cvCreateImageHeader(cvSize(WIDTH, HEIGHT), IPL_DEPTH_8U, 1);
    IplImage *right = cvCreateImageHeader(cvSize(WIDTH, HEIGHT), IPL_DEPTH_8U, 1);

    // Run capture loop until  key is pressed
    while((cvWaitKey(1) & 0xff) != 27)
    {
        // Capture DUO frame
        PDUOFrame pFrameData = GetDUOFrame();
        if(pFrameData == NULL) continue;

        // Set the image data
        left->imageData = (char*)pFrameData->leftData;
        right->imageData = (char*)pFrameData->rightData;

        // Process images here (optional)

        // Display images
        cvShowImage("Left", left);
        cvShowImage("Right", right);
    }

    // Release image headers
    cvReleaseImageHeader(&left);
    cvReleaseImageHeader(&right);

    // Close DUO camera
    CloseDUOCamera();
    return 0;
}
```

Resources

Related

- DUO API
- DUO SDK
- DUO Developers
- DUO Devices
- DUO Downloads