WiDy SWIR

User Manual

Table of contents

1. Kit description

We thank you for ordering a WiDy SWIR camera kit.

The kit comprises (See figure 1):
- A WiDy SWIR camera [1]
- A USB cable [2]
- A Trigger cable [3]
- A USB Key with WiDy View Software and User Manual [4]

Figure 1- WiDy SWIR kit

The WiDy camera is equipped with a CS/C-Mount adaptor. It has a unique serial number labeled on a sticker that allows traceability of sensor and software parameters.

When use of camera is finished or for transportation, we recommend putting back each item in their location in protecting foam inside the case to allow protection and avoid losing pieces.

2. Operation

a. Software installation

The USB Key [*item 4*] comprises the following files:

- WiDy View software
- User Manual

It is recommended:
- ▪ to save the complete software package as it contains the factory settings (Bad pixel replacement and Non uniformity corrections)
- ▪ to execute Software from local hard drive AND NOT FROM USB KEY

b. Camera connexion

The USB 2.0 cable [*item 2*] allows to power the WiDy camera and to output the data in 14 bits.

Plug the USB cable (mini USB male connector end) into the female mini USB connector on the back of the camera as shown beneath.
Plug the USB male connector into one of the USB slot on the computer.
The camera will then be recognized by Windows as a standard USB device

Mini USB female connector
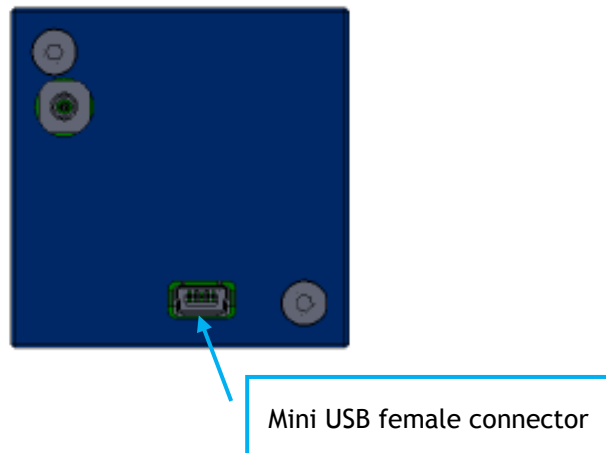
c. Triggering of camera

The WiDy SWIR camera can be triggered (i.e. synchronized with external event by an external pulse) using a BNC-SMC cable [*item 3*]

Plug the trigger cable (SMC female connector end) into the male SMC connector on the back of the camera as shown beneath.

Plug the BNC connector to a pulse generator. The pulse signal must have square shape and must be comprised between 0 and 3.3 volts.
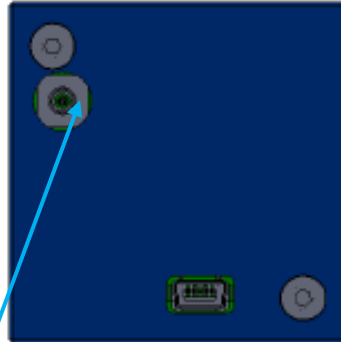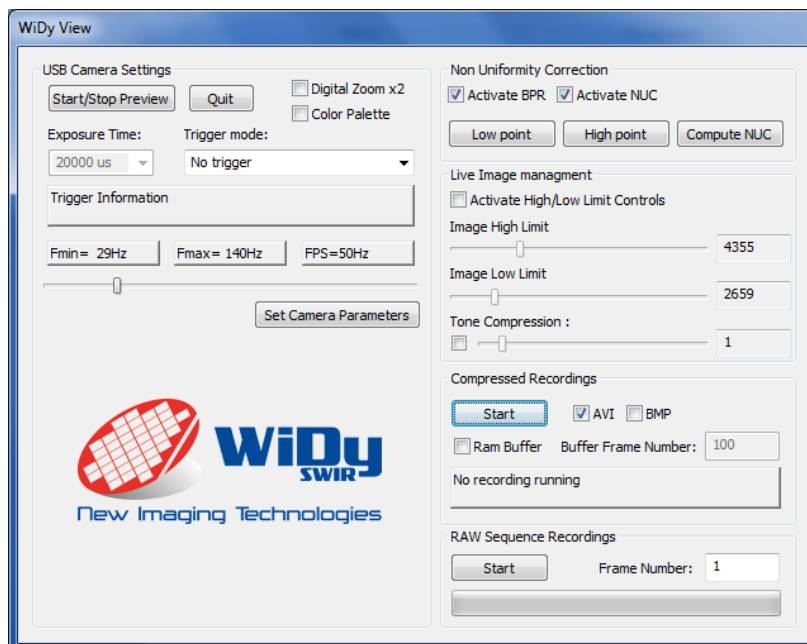
SMC male connector

3. Accessible functions

All of the following set-up functions are selectable via the WiDy View software interface.

The WiDy View interface is shown beneath

a. Frame rate

The frame rate can be selected between the minimum value (F_{min}) and maximum (F_{max}) values displayed on the left and center rectangles by moving a cursor. To activate the selection you have to click on Set camera parameters button. The selected frame rate is displayed on the right rectangle.

b. Digital Zoom

The native format of sensor of the WiDy SWIR camera is 320 x256 pixels. By default the WiDy SWIR camera is preset with a 2 X zoom for displaying the image. If you want to display image in native format, you have to unclick the Digital Zoom x2 case

c. Non Uniformity Correction

Two points correction are available on the Widy SWIR camera.

1- A two point factory calibration is done by NIT and the camera can operate without new calibration in majority of cases. This calibration is stored in the two files CNUH.yml and CNUL.yml in the player directory. *It is recommended to save those two files in a separate directory for saving them as back-up*.

2- If user wants to make a new calibration [*any new calibration will replace the previous one*], the following procedure must be followed carefully:

i. Acquire "Low point" (*low temperature or dark illumination*). The camera must be pointed to **a uniform dark** source. To launch the acquisition, press the "Low point" button. During the acquisition the image will freeze. Once image is live again; acquisition of low point is finished.

ii. Acquire "High point" (High temperature or illumination). The camera must be pointed to **a uniform light** source [*hint; use a light diffuser in front of the camera*]. To launch the acquisition, press the "High point" button. During the acquisition the image will freeze. Once image is live again; acquisition of high point is finished.

iii. At this point, the NUC table has not been updated. To compute the new table, press the "Compute NUC" button. Your new calibration is now applied.

d. Image management

    i. **Activate High/Low limits Control**. By default when this function is deactivated, the software performs automatic image adaptation to the screen range (Digital AGC). Once activated, the display range can be adapted manually through the High and Low limits sliders. The RAW recording is not affected by either the AGC or the High and Low limits.

    ii. **Color Palette**. The standard grey scale image can be converted to a false color mode by activating the "Color Palette" function.

e. Video recording

    i. **Compressed recordings**

    AVI

The AVI recording function stores the live stream displayed on screen. The file is encoded using mpeg4 codec. Two modes are available:

- **Ram Buffer acquisition**. Specify the number of frame to be acquired in "Buffer Frame Number" box. The software will then record the sequence in the RAM and then transfer it on hard drive.

- **Continuous recording**. Press "Start" button to start sequence acquisition. A dialog box will request file location and file name. The software will acquire image continuously until user press "Stop" button.

BMP

The BMP recording function stores sequence of .bmp files. Two modes are available:

- **Ram Buffer acquisition**. Specify the number of frame to be acquired in "Buffer Frame Number" box. The software will then record the sequence in the RAM and then transfer it on hard drive.

- **Continuous recording**. Press "Start" button to start sequence acquisition. A dialog box will request file location and file name. The software will acquire image continuously until user press "Stop" button.

    ii. **RAW sequence recordings**

RAW recording consists of recording a sequence of images in full dynamic range format (14 bits). No image processing is applied in this mode this. The resulting file contains as many frame as

specified in "Frame number" box prior to start recording. A dialog box will request file location and file name. Section 5 of this document describes the RAW image file format to be read by MATLAB or C language software routines.

4. Camera Characteristics

Specifications	Value
Photodiode array	InGaAs
Spectral response	900nm to 1750nm
Number of pixels	320x256
Pitch	25um x 25um
Array cooling	Uncooled
Dynamic Range	>140dB
Pixel operability	>99%
Frame rate	Up to 150 fps
Digital output	USB 2.0
Power supply	Through USB 2.0
Synchronization	Internal or External I0 (LVTTL)

External Dimensions	46x46x32 mm
Image Format	Rolling Shutter
Weight	120 g (lens not included)
Operating Temperature	0˚C to 50˚C
Optical interface	C or CS mount
WidyWiew Software	Operates on Win XP,7, and 8
Image storage format	Video AVI and full dynamic range RAW format (accessible from MATLAB)

5. RAW Image Format

The RAW database is a binary file where each pixel is stored under 16 bits format. The architecture of the database is the following:

The main database header describes proprietary information and is **3476** Bytes long.

There is short header prior to the location of each frame which is **1016** Bytes long.

Each frame starts after each frame header. Frames are stored under 16 bits per pixels. If the ADC is 14 bits then the two upper bits are equal to zero. The frame size is equal to number of pixels of the sensor, for NSC0806 and NSC0902 it is 768x576 pixels = **442 368** pixels = **884 736** Bytes.

The following C langage and Matlab routines allows reading one frame from a database.

MATLAB routine to read an image from the file.

```
clear all

close all

clc

% Open a Dialogbox to get the database file .ptw

[filename  pathname] = uigetfile('*.ptw*', 'Choose a picture ptw file');


% Initialization of the file pointer.

fid = fopen([pathname filename],'r');
```

```
% File Header length in Bytes.

LgthFileMainHeader = 3476;


% Image Header length in Bytes.

LgthImHeader = 1016;




% Recover the number of pixels in images:

fseek(fid, 23, 'bof');

NbPixelImage = fread(fid,1,'uint32');


% Recover the total number of images:

fseek(fid, 27, 'bof');

Nbimage = fread(fid,1,'uint32');


% Recover width of images:

fseek(fid, 377, 'bof');

NbColImage = fread(fid,1,'uint16');


% Recover height of images:

fseek(fid, 379, 'bof');


NbRowImage = fread(fid,1,'uint16');


% Set the pointer at the beginning  of the image header.

fseek(fid, LgthFileMainHeader, 'bof');


% Initialization of a viedo buffer.

A = zeros(NbRowImage,NbColImage,Nbimage,'uint16');

h = waitbar(0,[filename ' database importation : ' num2str(0) '/' num2str(Nbimage) ]);




% Main Loop

for i=1:Nbimage
```

waitbar(i/Nbimage,h,[filename ' database importation : ' num2str(i) '/' num2str(Nbimage)]);

  % The file pointer fid is incremented by LgthImHeader.

fread(fid,LgthImHeader);

% The image is extracted from the file.

% The image is stored in raw order in the binary file.

B = fread(fid , [NbColImage,NbRowImage] , 'uint16');

% The image is stored in the buffer.

A(:,:,i) = uint16(B');

end

close(h)

fclose('all');

C Langage routine to read and display an image from a RAW database. It requires OPENCV as an open source display library.

```cpp
#include <iostream>

#include <fstream>

#include "windows.h"

#include "Mmsystem.h"

#include "cv.h"

#include "highgui.h"


using namespace std;

using namespace cv;



// Simple Program for reading images in a .ptw files and

// and displaying it using OpenCV library.


// Initialization of the different buffers to be used.

unsigned short *bufferImage;

char *buffer;

ifstream DataBase;

int LgthFileMainHeader = 3476;

int LgthImHeader = 1016;

int NombrPixelImage=0;

int NombrImage=0;

unsigned short NombrColImage=0;

unsigned short NombrRowImage=0;

Mat CVImBuffer;

Mat CVIm;

double MaxVal;

double MinVal;



void main()

{
```

WDR InGaAs NIR Camera
QVGA resolution

```cpp
 //Open the file "Data.ptw" for reading

    DataBase.open("C:\\Documents and Settings\\cbouvier\\Bureau\\testvid.ptw",
ios::in|ios::binary);


    DataBase.seekg(0,ios::beg);

    //Recover the number of pixels in images:

     DataBase.seekg(23,ios::beg);

     DataBase.read((char *)&NombrPixelImage,4);



    //Recover the total number of images:

     DataBase.seekg(27,ios::beg);

     DataBase.read((char *)&NombrImage,4);



    //Recover width of images:

     DataBase.seekg(377,ios::beg);

     DataBase.read((char *)&NombrColImage,2);



    //Recover height of images:

     DataBase.seekg(379,ios::beg);

     DataBase.read((char *)&NombrRowImage,2);



    // We Initiate the image buffer size. Images are stored in the database

    // are stored under 16 bits per pixel format.

    buffer = (char *)malloc( NombrPixelImage*2*sizeof(char) );



    //Matrix allocation for displaying images with OpenCV.

    CVImBuffer.create(NombrRowImage,NombrColImage,CV_8U);

    CVIm.create(NombrRowImage,NombrColImage,CV_32F);



     DataBase.seekg(3476,ios::beg);



    if(DataBase.is_open())

    {

            for (int i=0; i<NombrImage; i++)
```

Page 14

```cpp
                {
                        // The file pointer is moved to the beginning of the image pixels data block.

                        DataBase.seekg(1016,ios::cur);


                        //We read the data block corresponding to the image pixels in row order.

                        DataBase.read(buffer,884736);


                        //A pointer is initiated to fill a OpenCV array and we fill the array with the
image pixels.

                        float* Mj = CVIm.ptr<float>(0);

                        for (int jjj = 0; jjj<(int)NombrRowImage * (int)NombrColImage; jjj++)

                        {


                                bufferImage = (unsigned short*)&buffer[2*jjj];

                                Mj[jjj] = saturate_cast<float>(*bufferImage);



                        }

                        // The image is rescaled to be displayed on a conventional display.

                        minMaxLoc(CVIm, &MinVal, &MaxVal);

                        CVIm = (CVIm - MinVal)/(MaxVal-MinVal);

                        CVIm = 255.0 * CVIm;

                        CVIm.convertTo(CVImBuffer,CV_8U);

                        //imwrite("Imtest.bmp",CVIm);


                        // We draw the image on the display.

                        imshow("Video Stream",CVImBuffer);

                        cout << "Image =" << i  << endl;

                        waitKey();

                }

                DataBase.close();

        }

        else

        {

        cout << "Unable to open file";

        }

        }
```

NIT
1 Impasse de la noisette, Bat D- 1er étage, BP426
91370 Verrières Le Buisson Cedex, France
Tel +33(0) 1 64 47 88 58
www.new-imaging-technologies.com
info@new-imaging-technologies.com

NIT, New Imaging Technologies and Native WDR, are trademarks of New Imaging Technologies.