Omnidirection Camera Series NM33

Software Development Kit

“ OptCamSDK “

User’s Manual

Rev. 1.05 2018/01/17

OPT Corporation

5423-2 Miyagawa, Chino-shi, Nagano 391-0013 Japan
Tel: 0266-82-0020 Fax: 0266-82-0022

Table of Contents

1. OVERVIEW

OptCamSDK is the software development kit that is possible to operate the major feature of the omnidirection camera NM33 series by linking to the user made application.

This user’s manual is written for the DLL for the firmware of the OPT standard driver version (“OPT version”) and the Windows standard driver version (“ UVC version “).

Applicable DLL version :

OPT version DLL :	v0.012 and later
UVC version DLL :	v1.012 and later

2. FEATURES

- ◆ To recognize the connected cameras and select the camera to use.
- ◆ To get the image from the camera.
- ◆ To get and set the camera view angle and shooting condition.
- ◆ To set the indication of the shooting information (valid or invalid)
- ◆ To initialize the camera settings

3. CONFIGURATION

- SSK.dll ・・・DLL
- SSK.lib ・・・Library for making application
- SSK.h ・・・Function header for making application

4. OPERATING ENVIRONMENT

OS:	Windows Vista, Windows 7, 8, 8.1, and 10
PC:	The environment for proper functioning for above OS.

Recommended Specification:

CPU : Pentium３and more (Pentium４/M/D/Dual-Core,Core Solo/Duo)
Memory size : 256 Mbyte (768Mbyte for Vista) and more
HDD : 20 Gbyte and more

5. USE of FUNCTIONS

(1) Environment to use

With this DLL, the NM33 camera can be operated via USB.

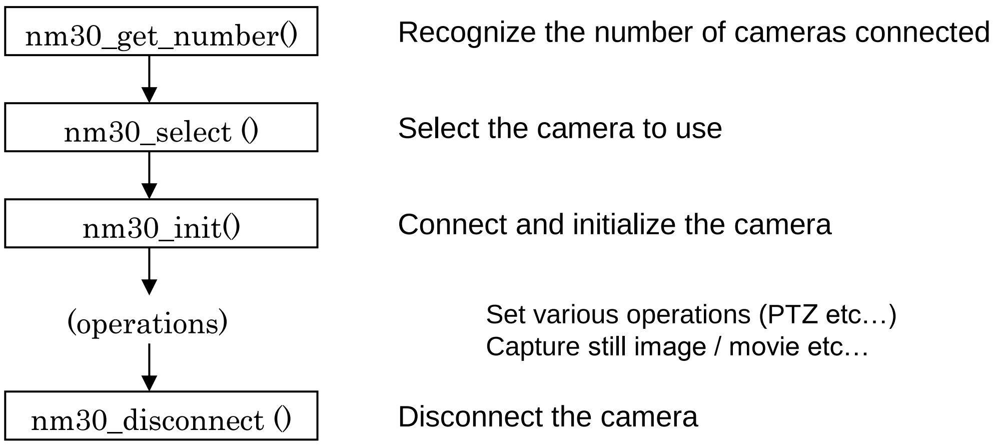
This DLL does not use RS232C though the NM33 camera supports the communication via RS232C.

(2) Device recognition, setting, and from initialization to exit

After recognized the number of cameras connected and selected the camera to use, execute the connection and initialization.

When the connection and initialization is completed properly, it becomes possible to set various parameters and/or capture still images and movie.

By ending the application, disconnect the camera and the device.


nm30_get_number()
Recognize the number of cameras connected
nm30_select ()
Select the camera to use
nm30_init()
Connect and initialize the camera
(operations)
Set various operations (PTZ etc…)
Capture still image / movie etc…
nm30_disconnect ()
Disconnect the camera

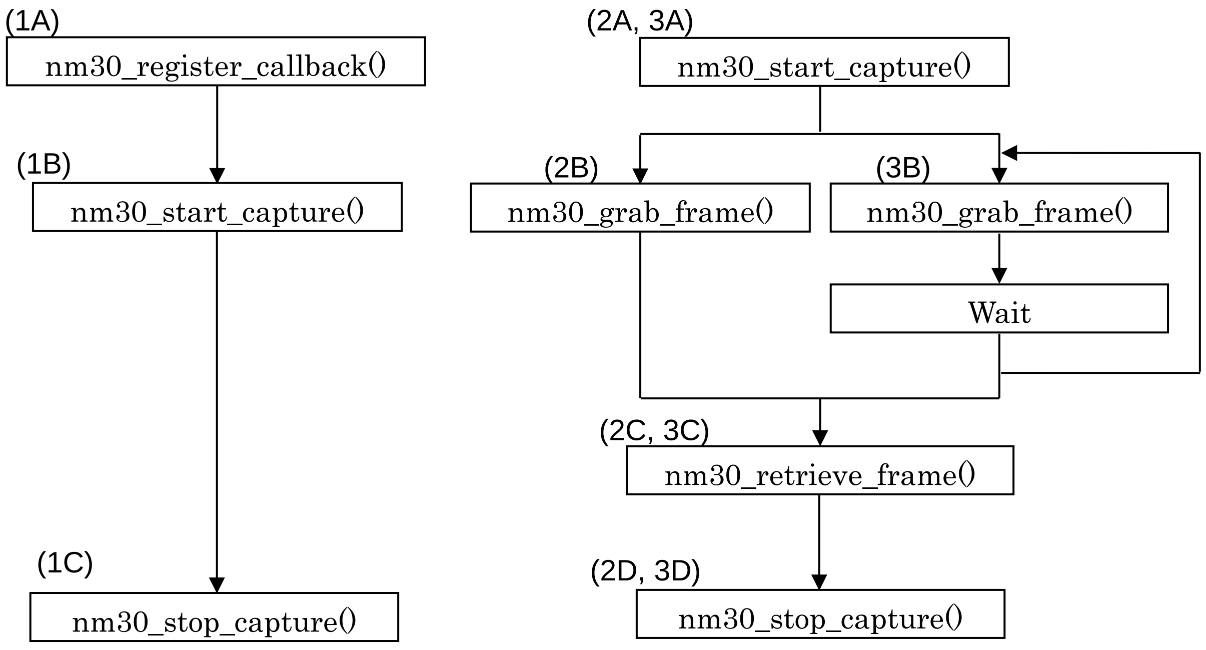
(3) Get the image frame

One of three methods is selectable to grab the image frame according to the format to make an application.

1. Method to register the callback function and make it recallable after completing capturing.
   Call up the functions in the following order:
   (A) Register the callback function → (B) Start capturing → (C) Stop capturing

2. Method to wait until grabbing the frame after calling up the capturing function.
   Call up the functions in the following order:
   (A) Start capturing → (B) Confirm the reception of the image frame → (C) Grab the image frame → (D) Stop capturing

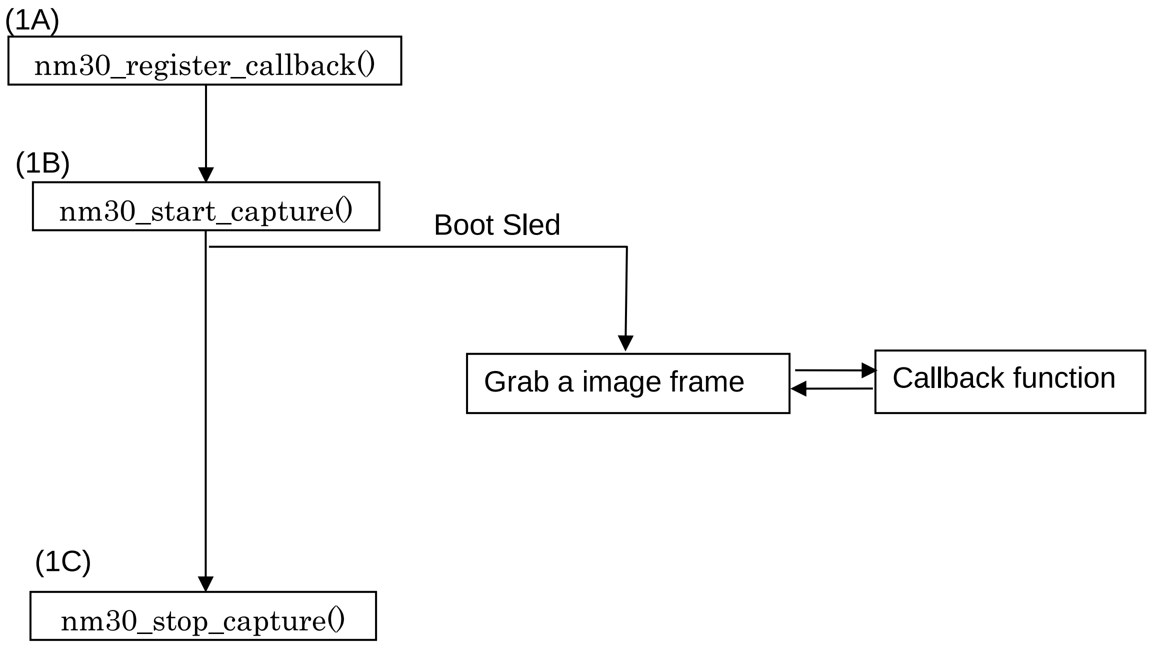
3. Method to wait until grabbing the frame after calling up the capturing function in the poling mode.
   Call up the functions in the following order:
   (A) Start capturing → (B) Confirm the reception of the image frame → (repeat) ・・・ → (B) Confirm the reception of the image frame → (C) Grab the image frame → (D) Stop capturing



< Detail of each type >

Type 1:

Execute nm30_start_capture() after registered the callback function by nm30_register_callback().
The sled to grab a image frame is booted by nm30_start_capture() .
After grabbing one image frame, the registered callback function is called.



After completing the treatment of the callback function, the controlling returns to the part of the retrieving the image frame inside of DLL, and then starts grabbing the following image frames.
This will be repeated until stopping the capturing by nm30_stop_capture().
nm30_grab_frame() and nm30_retrieve_frame() are not usable in Type 1.

Type 2 and 3 :

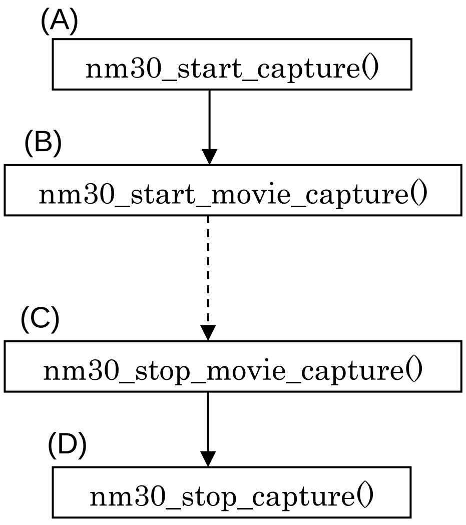
Use nm30_grab_frame() and nm30_retrieve_frame() after executing nm30_start_capture() in Type 2 and 3. Type 2 or 3 is decided according to the parameter value to pass to nm30_grab_frame().
Grabbing the following frames can be done by executing nm30_grab_frame() after retrieving the image frame by nm30_retrieve_frame().
It is not necessary to use nm30_stop_capture() for each image frame.

(4) Capturing movie

Captured movie is output directly to the file.

Call up the functions in the following order:

(A) Start capturing → (B) Start output the movie → (C) Complete the output of the movie → (D) Stop capturing



(5) About the Large size (1536x1536) image

As for the Large size image, the parameters for the capture image size and the type of panorama image are changed correspondingly.

(Setting / grabbed function)

Type of panorama image : nm30_set/get_panorama_mode()

Size of captured image : nm30_set_capture_size()

nm30_get_capture_width()/nm30_get_capture_height()

◆ If designating 1536x1536 by nm30_set_capture_size()

The type of panorama image becomes the one designating 9(circle(Large size)).

◆ If designating other than 1536x1536 by nm30_set_capture_size()

The type of panorama image is changed into the image mode prior to shift to the Large size.

If the Large size was set at the initial booting, it is set as 0 (circle) .

Note: This is available only with the OPT standard version. The UVC version does not support this function because it is not booted with the Large size.

◆ If designating other than 9(circle(Large size)) by nm30_set_panorama_mode()
The size of captured image is changed automatically to the VGA size.
If you want to display the image in other than VGA size, designate the desired size by nm30_set_capture_size().

6. List of Functions

Table 1 : List of Functions

Function	Feature
nm30_init()	Initialize the camera
nm30_get_number()	Get the number of camera connected
nm30_get_id()	Get the ID of the designated camera
nm30_param_save()	Save the set parameter to the camera
nm30_select()	Select the camera
nm30_set_panorama_mode()	Switch the type of panorama image
nm30_get_panorama_mode()	Get the type of current panorama image
nm30_set_pan()	Set the value for panning
nm30_get_pan()	Get the current value for panning
nm30_set_tilt()	Set the value for tilting
nm30_get_tilt()	Get the current value for tilting
nm30_set_zoom()	Set the value for zooming
nm30_get_zoom()	Get the current value for zooming
nm30_set_roll()	Set the value for rolling
nm30_get_roll()	Get the current value for rolling
nm30_enable_information_display()	Display the information of frame rate etc…
nm30_disable_information_display()	Disappear the information of frame rate etc…
nm30_enable_overlay_display()	Enable the fisheye overlay image display
nm30_disable_overlay_display()	Disable the fisheye overlay image display
nm30_jpeg_get()	Get the current JPEG compression ratio
nm30_jpeg_set()	Set the JPEG compression ratio
nm30_set_autopan_speed()	Set the auto panning speed
nm30_get_autopan_speed()	Get the current auto panning speed
nm30_set_flip_screen()	Set the flip / mirror condition
nm30_get_flip_screen()	Get the current flip / mirror condition
nm30_set_sharpness()	Set the sharpness filter
nm30_get_sharpness()	Get the current sharpness filter

nm30_set_exposure_time()	Set the shutter speed
nm30_get_exposure_time()	Get the current shutter speed
nm30_set_gain()	Set the value for gain
nm30_get_gain()	Get the current value for gain
nm30_enable_auto_exposure()	Enable the AE (Auto Exposure)
nm30_disable_auto_exposure()	Disable the AE
nm30_set_capture_size()	Set the width & height of the image to capture
nm30_get_capture_width()	Get the width of the current captured image
nm30_get_capture_height()	Get the height of the current captured image
nm30_set_capture_fps()	Set the value for frame rate
nm30_get_capture_fps()	Get the current value for frame rate
nm30_get_actual_fps()	Get the actual frame rate
nm30_start_capture()	Start capturing
nm30_register_callback()	Register the call back function of capturing
nm30_grab_frame()	Grab the image frame
nm30_retrieve_frame()	Retrieve the image frame
nm30_start_movie_capture()	Start capturing of movie
nm30_stop_movie_capture()	Stop capturing of movie
nm30_stop_capture()	Stop capturing
nm30_disconnect()	Disconnect the camera

7. Functions

- int nm30_init (void)

  Check the connection with camera and initialize

  Return value : 0 : properly completed, 1 : connection error, 2 : initialization error

- int nm30_get_number (void)

  Get the number of cameras currently connected.

  Return value : number of cameras connected

  Note : with the OPT standard version, only one camera is allowed to connect at once, therefore the return value is only 0 or 1.

- char* nm30_get_id (int id)

  Get the ID unique for each camera

  Parameter : camera number (1 ~)

  Return value : a pointer to the char type line where the ID unique for a camera (ex. NM33-012345)

- int nm30_param_save (int id)

  Save the current parameter

  Parameter : the camera number (1~)

  Return value : 0 : properly completed, 1 : error

- int nm30_select (int id)

  Select the camera

  Parameter : the camera number (1~)

  Return value : 0 : properly completed, 1 : error

  Note: the operation of functions described following here is applied to the camera selected here.

- int nm30_set_panorama_mode (int mode)

  Switch the type of panorama image

  Parameter : 0 : circle, 1 : fringe rotation, 2 : fringe rotation (up-side-down),
  3 : two-tiered panorama, 4 : two-tiered panorama (up-side-down),
  5 : quad, 6 : quad (up-side-down), 9 : circle (Large size),
  11 : 1 screen – orthogonal direction move

  Return value : 0 : properly completed, 1 : error

Note： with this function, the status of image (normal / up-side-down) is switched by the setting of flip condition and revise the set value for the flip / mirror condition (nm30_set_flip_screen())

- int nm30_get_panorama_mode (void)
  Get the type of the current panorama image
  Return value : 0 : circle, 1 : fringe rotation, 2 : fringe rotation (up-side-down),
  3 : two-tiered panorama, 4 : two-tiered panorama (up-side-down),
  5 : quad, 6 : quad (up-side-down), 9 : circle (Large size),
  11 : 1 screen – orthogonal direction move

  Note： with this function, since the status of image (normal / up-side-down) is judged by the setting of flip condition, the return value may be different from the one originally set if the set value for the flip / mirror condition (nm30_set_flip_screen()) was changed after setting the type of panorama image.

- int nm30_set_pan (float pan)
  Set the value for panning of panorama image
  Parameter : panning angle [deg]
  Return value : 0 : properly completed, 1 : error

- float nm30_get_pan (void)
  Get the value for the current panning of panorama image
  Return value : panning angle [deg]

- int nm30_set_tilt (float tilt)
  Set the value for titling of panorama image
  Parameter : tilt angle [deg]
  Return value : 0 : properly completed, 1 : error

- float nm30_get_tilt (void)
  Get the value for the current titling of panorama image
  Parameter : tilt angle [deg]

- int nm30_set_zoom (float zoom)
  Set the value for zooming of panorama image
  Parameter : OPT version : zoom angle [deg]
  UVC version : zoom magnification
  Return value : 0 : properly completed, 1 : error

- float nm30_get_zoom (void)
  Get the value for the current zooming of panorama image
  Parameter : OPT version : zoom angle [deg]
  UVC version : zoom magnification

- int nm30_set_roll (float roll)
  Set the value for rolling of panorama image
  Parameter : rolling angle [deg]
  Return value : 0 : properly completed, 1 : error

- float nm30_get_roll (void)
  Get the value for the current rolling of panorama image
  Parameter : rolling angle [deg]

- int nm30_enable_information_display (void)
  Enable to display the frame rate information
  Return value : 0 : properly completed, 1 : error

  Note: The information will overlay on the image

- int nm30_disable_information_display (void)
  Disable to display the frame rate information
  Return value : 0 : properly completed, 1 : error

- int nm30_enable_overlay_display (void)
  Enable to display the fisheye image overlay
  Return value : 0 : properly completed, 1 : error

- int nm30_disable_overlay_display (void)
  Disable to display the fisheye image overlay
  Return value : 0 : properly completed, 1 : error

- int nm30_jpeg_get(void)
  Get the current JPEG compression ratio
  Return value : JPEG compression ratio (integer number of 1 to 99)

- int nm30_jpeg_set(int jpegcomp)
  Set the PJEG compression ratio
  Parameter : JPEG compression ratio
  Return value : 0 : properly completed, 1 : error

- int nm30_set_autopan_speed (int speed)
  Set the auto panning speed
  Parameter : auto panning speed (-3 ~ 3)
  Return value : 0 : properly completed, 1 : error

- int nm30_get_autopan_speed(void)
  Get the current auto panning speed
  Return value : auto panning speed

- int nm30_set_flip_screen (int mode)
  Set the flip / mirror condition
  Parameter : flip / mirror conditions
  0 : no flip
  1 : flip in horizontal
  2 : flip in vertical
  3 : flip in horizontal and vertical
  Return value : 0 : properly completed, 1 : error

  Note : this set value may be changed according to the setting of panorama image by (nm30_set_panorama_mode()).

- int nm30_get_flip_screen(void)
  Get the current flip / mirror condition
  Return value : flip / mirror condition

  Note : this set value may be changed according to the setting of panorama image by (nm30_set_panorama_mode()).

- int nm30_set_sharpness (int filter)
  Set the sharpness filter
  Parameter : filter number (0~8)
  Return value : 0 : properly completed, 1 : error

- int nm30_get_sharpness(void)
  Get the current sharpness filter
  Return value : filter number

- int nm30_set_exposure_time (int time)
  Set the shutter speed (exposure time)
  Parameter : OPT Version : shutter speed (5~500,000μs)
  UVC version : shutter speed (classify the exposure time by 2^n,
  From -10 to -1
  Return value : 0 : properly completed, 1 : error

- int nm30_get_exposure_time(void)
  Get the current shutter speed (exposure time)
  Return value : shutter speed

- int nm30_set_gain (int gain)
  Set the value for gain
  Parameter : gain value x 1000 (10~32,000)
  Return value : 0 : properly completed, 1 : error

- int nm30_get_gain(void)
  Get the current value for gain
  Return value : gain value x 1000

- int nm30_enable_auto_exposure (void)
  Enable the AE (auto exposure)
  Return value : 0 : properly completed, 1 : error

- int nm30_disable_auto_exposure (void)
  Disable the AE (auto exposure)
  Return value : 0 : properly completed, 1 : error

- int nm30_set_capture_size (int width, int height)

  Set the width & height of the image to caputure

  Parameter : OPT version : width (pixel : 112~640 (multiple of 16), 1536)

  : height (pixel : 2~480 (multiple of 2), 1536)

  UVC version : designate one combination of the below factors

Size	Width	height
QVGA	320	240
VGA	640	480
LARGE	1536	1536

  Return value : 0 : properly completed, 1 : error

- int nm30_get_capture_width (void)

  Get the width of the current captured image

  Return value : width (pixel)

- int nm30_get_capture_height (void)

  Get the height of the current captured image

  Return value : height (pixel)

- int nm30_set_capture_fps (float fps)

  Set the value for frame rate (frame/sec) when capturing image

  Parameter : fps (0, 6.4～16.0)

  0 : switch to the auto frame rate

  other than 0 : switch to the fixed frame rate

  Return value : 0 : properly completed, 1 : error

- float nm30_get_capture_fps (void)

  Get the current value for frame rate (frame/sec) when capturing image

  Return value : 0 : switch to the auto frame rate

  other than 0 : switch to the fixed frame rate and the current vale

- float nm30_get_actual_fps (void)

  Get the actual frame rate when capturing

  Return value : fps

- int nm30_start_capture (void)
  Start capturing of image to the frame buffer
  Return value : 0 : properly completed, 1 : error

- nm30_register_callback(ImagecallbackType)
  Register the call back function of capturing
  Parameter : pointer (specified by SSK.h) to the call back function
  Return value : 0 : fail in setting, 1 : complete in setting

  ◎ format of the call back function

```
typedef INT (nudecl *ImagecallbackType)
    (INT nSize,                 : size of image frame
      UCHAR* pData,             : pointer for the image frame data
      USHORT Width,             : width of the image frame
      USHORT Height,            : height of the image frame
      USHORT Format             : type of the image frame (fixed as JPEG)
      SYSTEMTIME Time);         : time to get the image frame
```

- int nm30_grab_frame (int mode)
  Grab the image frame
  Parameter : 0 : wait until to get the frame
                 1 : not wait until to get the frame (= polling mode)
  Return value : 0 : fail in grabbing, 1 : complete in grabbing
  Note : The grabbed image frame will be stored in the memory area kept in the inside of DLL.

- ImageStract* nm30_retrieve_frame (void)
  Return a pointer to the image frame grabbed.
  Return value : pointer (specified by SSK.h) to the image frame

  ◎ format of the image frame information

```
typedef struct {
    BYTE*   StreamImageBytes;        : pointer of the image frame
    long    StreamLength;            : size of the image frame
    long    ImageWidth;              : width of the image frame
    long    ImageHeight;             : height of the image frame
    SYSTEMTIME Time;                 : time to get the image frame
} ImageStruct;
```

Note: Since the StreamImageBytes contains the whole of JPEG image data including the JPEG header, it is usable as a JPEG data as is.
The pointer grabbed will be released in the DLL side. Don't release in the application side.

- int nm30_start_movie_capture (char* filepath, int filetype)
  Start capturing the movie
  Parameter : filepath : Path (full path) of the output file
  filetype : type of the output file
  0(TYPE_AVI) : AVI format
  1(TYPE_MJPEG) : MJPEG format
  Return value : 0 : properly completed, 1: error

- int nm30_stop_movie_capture (void)
  Stop capturing the movie
  Parameter : no
  Return value : 0 : properly completed, 1: error

- int nm30_stop_capture (void)
  Stop capturing the image into the frame buffer
  Return value : 0 : properly completed, 1: error

- int nm30_disconnect (void)
  Disconnect the camera
  Return value : 0 : properly completed, 1: error

8. Sample Program

◎ Sample 1

This program is a sample to grab one image frame and save the JPEG data into a file.

Executing CaptureSingleFrame() will proceed the followings and conclude.

Recognize the number of camera connected, select the camera 1, and initialize then → grab the image frame → save the JPEG data in a file → disconnect the camera

Sample 1

```
#include "SSK.h"

// declare Prototype
int         CaptureSingleFrame( void );
int         SaveImage( BYTE *p, long len );

// function name : CaptureSingleFrame()
// feature :
//     grab images from the camera
//     save in afile
//
//     treatment procedure
//       connect the camera
//       → grab the image frames → save as a JPEG file
//       → disconnect the camera
//
// return value : (properly completed)    other than 0 (error)
//
int CaptureSingleFrame( void )
{
            int ret;
            int fret;       // return value for this function
            ImageStruct *imageInfo;         // image frame information

            fret = 0;
            imageInfo = NULL;

            // recognize the number of cameras connected
            ret = nm30_get_number();
            if ( ret <= 0 ) return -1;

            // select the camera 1
            ret = nm30_select(1);
            if ( ret != 0 ) return -2;

            // initialize the camera
            ret = nm30_init();
            if ( ret != 0 ) return -3;

            // start capturing into the frame buffer (inside of DLL)
            ret = nm30_start_capture();
            if ( ret != 0 ) {
                fret = -4;
                goto CAM_CLOSE;
            }

            // set the capture size
            ret = nm30_set_capture_size( 640, 480 );            // VGA
            if ( ret != 0 ) {
                fret = -5;
                goto CAM_CLOSE;
            }

            // capture the image frame (the image is captured at this time)
            ret = nm30_grab_frame(0);
            if ( ret != 1 ) {
                fret = -6;
                goto CAM_CLOSE;
            }

            // grab the point for the image data
            imageInfo = nm30_retrieve_frame();
            if ( imageInfo == NULL ) {
                fret = -7;
                goto CAM_CLOSE;
            }

            // save in aJPEG file
            SaveImage( imageInfo->StreamImageBytes, imageInfo->StreamLength );

CAM_CLOSE:
            // stop capturing into the frame buffer
            ret = nm30_stop_capture();
            if ( ret != 0 ) fret = -8;

            // disconenct the camera
            ret = nm30_disconnect();
            if ( ret != 0 ) fret = -9;

            return fret;

}

// save the image data into a file
int         SaveImage( BYTE *p, long len )
{
            HANDLE hf;

            hf = CreateFile( L"NM30Image.jpg", GENERIC_WRITE, 0, NULL,
                CREATE_ALWAYS, FILE_ATTRIBUTE_NORMAL, NULL );
            if ( hf == NULL ) {
                return -1;
            }

            DWORD wlen;
            BOOL bret;
            bret = WriteFile( hf, p, len, &wlen, NULL );

            CloseHandle(hf);

            if ( bret == false ) return -2;
            return 0;
}
```