The SDK of Fisheye IP Camera

Revision: v2.1

Date: 2021/3/16

TABLE of CONTENTS

I. Overview

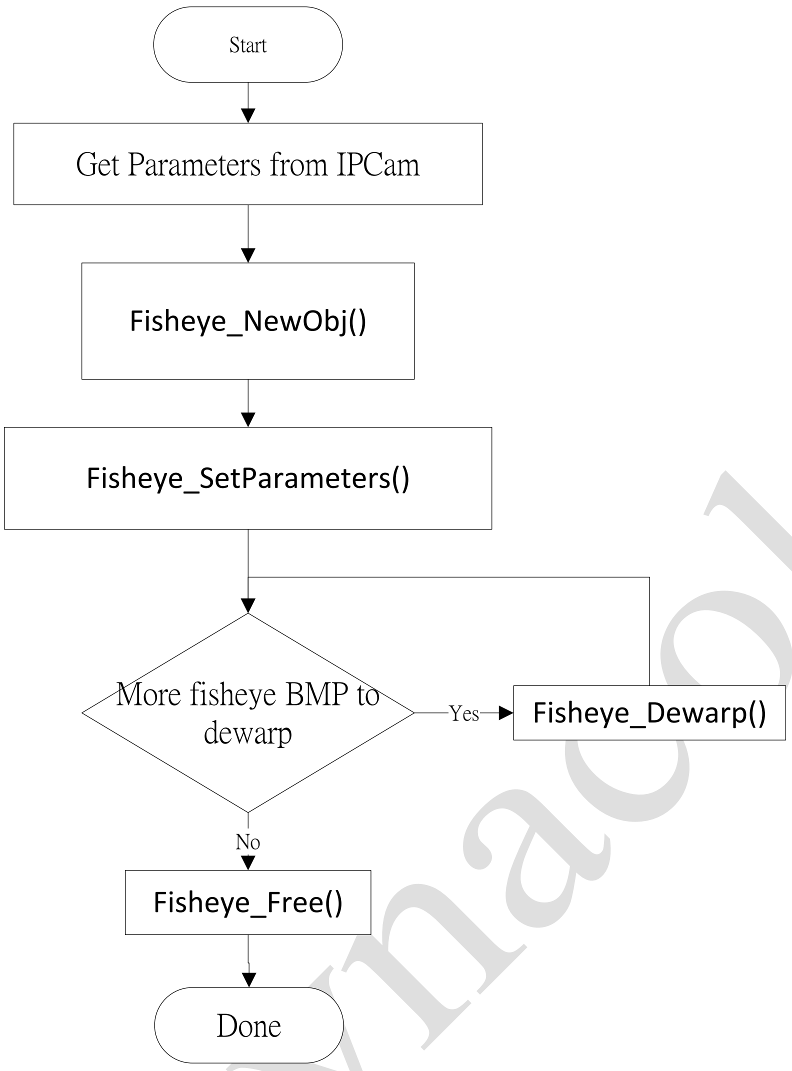
This document specifies the Fisheye IP Camera SDK Version 1.0.0.1 and above. This SDK programming guide is to provide a guideline of 1) establishing object for un-circling the original fisheye image, 2) establishing object for de-warpping the original fisheye image.

II. Functions

- **Outline of Contents**
  - Flow Chart
  - Get Related Parameters
  - Control Functions
  - Performance Issue

Flow Chart

Use Fisheye processing object to dewarp fisheye BMP streaming.



Get Related Parameters

Command syntax:

`http://<servername>/cgi-bin/admin/param.cgi?action=list[&group=<parameter>]`

Description: Get fisheye related parameters for image processing object by set the parameters in Fisheye_SetParameters() to have 360-degree panorama view and dewarp view.

Configuration file: /etc/sysconfig/fisheye.conf

[Fisheye.F0]

Parameter name	Value	Description
Projection	stereographic, equidistant	This parameter is read only.
View.Stream1	V#	Stream1 fisheye original view (refer to View.V#). The # is replaced with a view number. This parameter is read only.
View.Stream2	V#	The # is replaced with a view number. This parameter is read only.
View.Stream3	V#	The # is replaced with a view number. This parameter is read only.
View.Stream4	V#	The # is replaced with a view number. This parameter is read only.
View.V#.Width	An integer	The width of fisheye original view. This parameter is read only.
View.V#.Height	An integer	The height of fisheye original view. This parameter is read only.
View.V#.CenterX	An integer	x coordinate of fisheye original view center. This parameter is read only.
View.V#.CenterY	An integer	y coordinate of fisheye original view center. This parameter is read only.
View.V#.Diameter	An integer	The diameter of fisheye original view. This parameter is read only.
View.V#.FocalLength	An integer	The focal length of the fisheye lens. This parameter is read only.

About the View.Width & View.Height, CenterX, CenterY & Diameter:

Use the param.cgi to get proper width, height, CenterX, CenterY & Diameter.
The following topic provides more details which related to these parameters.
The View.V#.Width & View.V#.Height is not absolutely equal to the image resolution.
For example, for a fisheye image:

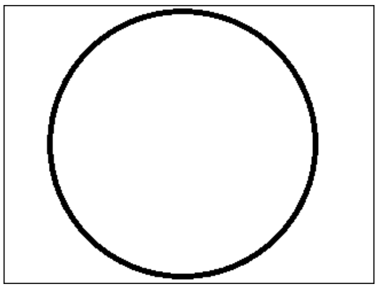
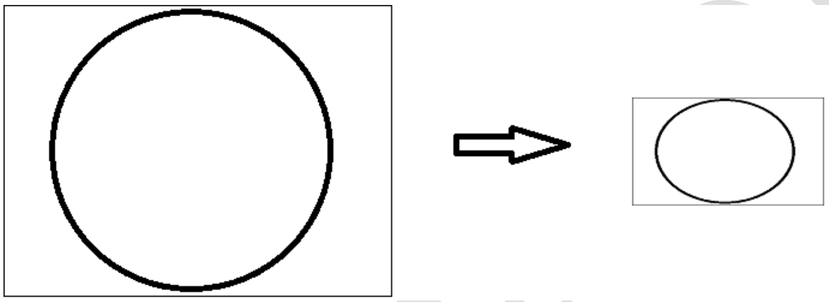


Image resolution: 2592x1944
View.V#.Width : 2592
View.V#.Height: 1944
View.V#.Diameter : 1930
View.V#.CenterX : 1296
View.V#.CenterY : 972

The fisheye dll can dewarp the fisheye image successfully while the correct fisheye related parameters are input.
For a D1 or CIF image got by scale down from 2592x1944 resolution picture as below:



The fisheye image of D1 or CIF will not be a circle after scaling, so the Diameter, CenterX and CenterY cannot be got directly from the D1 or CIF image.
Therefore, if the input image is scaled, user has to send the original View Width, Height, CenterX, CenterY & Diameter.

For example:
A D1 resolution image scaled down from 2592x1944 fisheye image and input to fisheye dll, the following parameters should be used for Fisheye_SetParameters:
Image resolution: 720x576
view_w: 2592
view_h: 1944
diameter : 1930
cx : 1296
cy : 972
After setting these parameters, fisheye dll can recognize that this image is scaled from the "original fisheye view", so the dll can dewarp the image properly. (this issue won't occur if view_w and view_h is gotten by using param.cgi. The descriptions

above just provide more information about it.)

And if the input image is cropped into 1920x1080 (from 2592x1944), what kind of parameters should be input?
Input the view_w, view_h : 1920, 1080
Then calculate appropriate cx, cy of the 1920x1080 image.
Diameter and FocalLength does not change after cropping.

● Control Functions

Fisheye_NewObj

Get a handle of fisheye image processing object.

Function Prototype:

```
int Fisheye_NewObj();
```

Return Value:

<0	error
>=0	handle of fisheye image processing object

Fisheye_Free

Destruct the fisheye image processing object; release memory resource of the fisheye image processing object.

Function Prototype:

```
int Fisheye_Free(int hd);
```

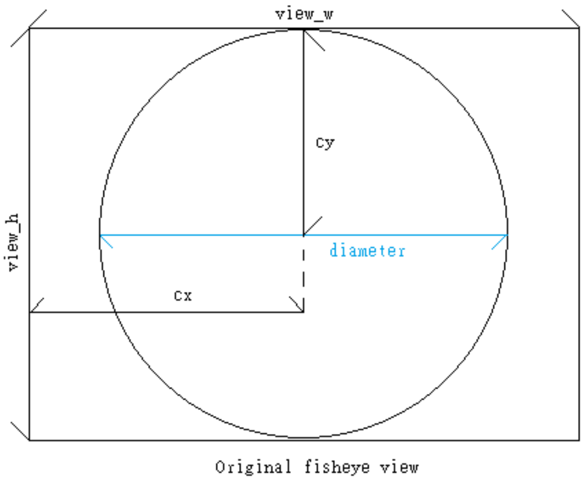
hd	handle of fisheye image processing object

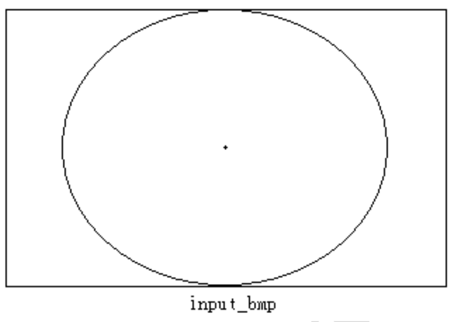
Return Value:

<0	error
0	success

Fisheye_SetParameters

Set parameters to the fisheye image processing object, e.g.: focal length, fisheye image diameter, center…





Function Prototype:

```
int Fisheye_SetParameters(int hd, int view_w, int view_h,
        int cx, int cy, int diameter, float focalLength, int projection,
        int rotation, const BITMAPINFOHEADER * input_header);
```

hd	handle of fisheye image processing object.
view_w	width of the original fisheye view.
view_h	height of the original fisheye view.
cx	x coordinate of the original fisheye view center.
cy	y coordinate of the original fisheye view center.
diameter	diameter of the original fisheye view.
focalLength	focal length of the fisheye lens.
projection	the fisheye lens projection method(stereographic projection, equidistant …)
rotation	refer to FISHEYE_ROTATION enumeration.
input_header	fisheye BMP info header. (The decoded image from IPCam Streaming)

```
enum FISHEYE_ROTATION {
        FISHEYE_NORMAL = 0,
        FISHEYE_FLIP = 1,
        FISHEYE_MIRROR = 2,
        FISHEYE_ROTATE = 3,
        FISHEYE_CLOCKWISE = 4,
        FISHEYE_COUNTERCLOCKWISE = 5
};
```

Return Value:

<0	Error
0	Success

Fisheye_Dewarp

PTZ mode, to dewarp a fisheye image.

Function Prototype:

```
int Fisheye_Dewarp(int hd, int pan, int tilt, float zoom,
        const BITMAPINFOHEADER * input_header, const unsigned char *input_bmp,
        BITMAPINFOHEADER * output_header, unsigned char *output,
        int out_imgW, int out_imgH);
```

hd	handle of fisheye image processing object.
pan	the pan angle. The recommended value can be obtained by using Fisheye_GetPTFromOutImg() or Fisheye_GetPTValue().
tilt	the tilt angle. The recommended value can be obtained by using Fisheye_GetPTFromOutImg() or Fisheye_GetPTValue().
zoom	zoom ratio of the dewarping output image.
input_header	info header of the input fisheye image.
input_bmp	bitmap of the input fisheye image.
output_header	info header of the dewarping image.
output	bitmap of the dewarping image.
out_imgW	the width of the output dewarping image.
out_imgH	the height of the output dewarping image.

Return Value:

<0	Error
0	Success

See also:

Fisheye_GetPTFromOutImg()

Fisheye_GetPTValue()

Fisheye_GetPTFromOutImg

Get recommended pan & tilt value for Fisheye_Dewarp() from a processed image and a point (x, y). The point (x, y) will be at the center of the new image by

Fisheye_Dewarp().

Function Prototype:

```
int Fisheye_GetPTFromOutImg(int hd, int x, int y, int *panPt, int *tiltPt);
```

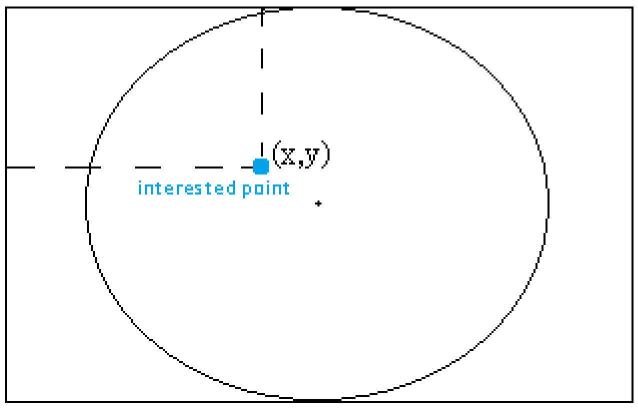
hd	handle of fisheye image processing object
x	x coordinate of the processed image
y	y coordinate of the processed image
panPt	the recommended pan value for Fisheye_Dewarp()
tiltPt	the recommended tilt value for Fisheye_Dewarp()

Return Value:

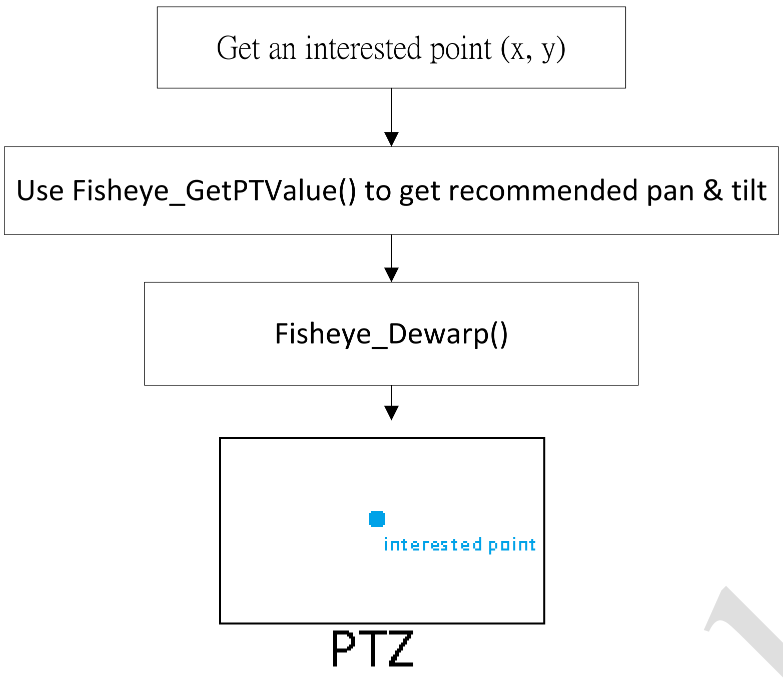
<0	Error
0	Success

Fisheye_GetPTValue

Get recommended pan & tilt value for Fisheye_Dewarp() from a fisheye bitmap and a interested point (x, y) of bitmap. The point (x, y) will be at the center of the new image by Fisheye_Dewarp().



input_bmp


Get an interested point (x, y)
Use Fisheye_GetPTValue() to get recommended pan & tilt
Fisheye_Dewarp()
interested point
PTZ

Function Prototype:

```
int Fisheye_GetPTValue(int hd, int x, int y, int *panPt, int *tiltPt);
```

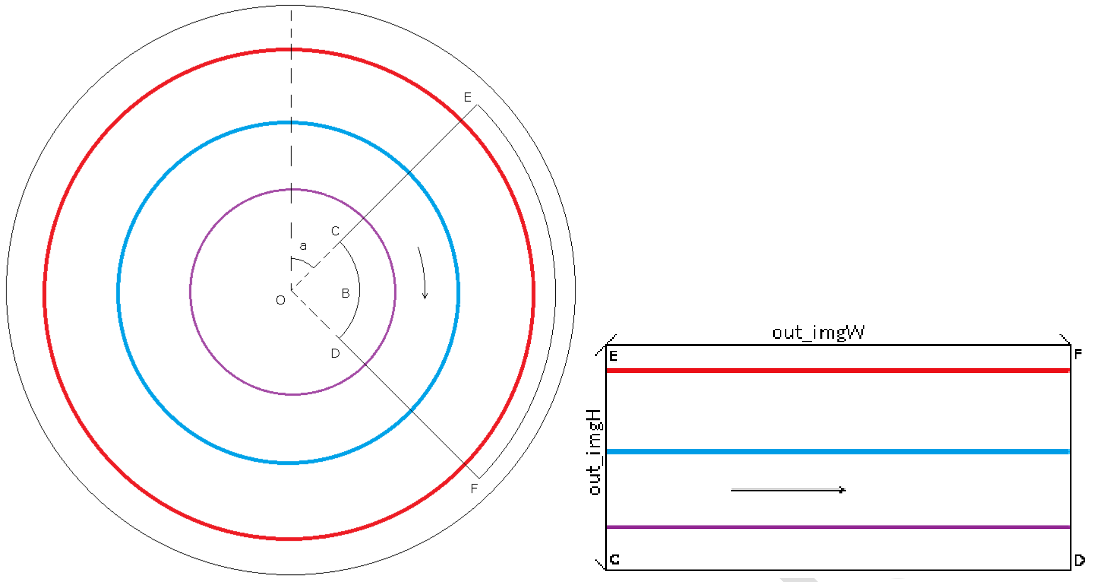
hd	handle of fisheye image processing object
x	x coordinate of the bitmap
y	y coordinate of the bitmap
panPt	the recommended pan value for Fisheye_Dewarp()
tiltPt	the recommended tilt value for Fisheye_Dewarp()

Return Value:

<0	error
0	success

Fisheye_360_Standard

Provide 360-degree panoramic view from original fisheye view.



Function Prototype:

```
int Fisheye_360_Standard(int hd, int alpha, int beta, int isClockwise, int inStart,
        int inEnd, const BITMAPINFOHEADER * input_header,
        const unsigned char *input_bmp, BITMAPINFOHEADER * output_header,
        unsigned char *output, int out_imgW, int out_imgH);
```

hd	handle of fisheye image processing object.
alpha	the angle, a, from 12 o'clock.
beta	the degree, b, of circular sector needs to be un-circled.
isClockwise	1: Un-circle in clockwise; 0: Un-circle in counter clockwise
inStart	the distance, $\overline{OC}$, from center of the original fisheye view to the initial point of the circular sector.
inEnd	the distance, $\overline{OE}$, from center of the original fisheye view to the end point of the circular sector.
input_header	info header of the input fisheye image.
input_bmp	bitmap of the input fisheye image.
output_header	info header of the 360 mode output image.
output	bitmap of the 360 mode output image.
out_imgW	the width of the 360 mode output image. The recommended value can be obtained by using Fisheye_360_Standard_Get_OutImgW().
out_imgH	the height of the 360 mode output image. The recommended value can be obtained by using

Fisheye_360_Standard_Get_OutImgH().

Return Value:

<0	Error
0	Success

See also:
Fisheye_360_Standard_Get_OutImgW()
Fisheye_360_Standard_Get_OutImgH()

Fisheye_360_Standard_Get_OutImgW

Get a recommended width for Fisheye_360_Standard() to avoid the output image is out of shape.

Function Prototype:

```
int Fisheye_360_Standard_Get_OutImgW(int hd, int beta, int inStart, int inEnd, int
out_imgH, int * out_imgWPt);
```

hd	handle of fisheye image processing object
beta	parameter beta used in Fisheye_360_Standard()
inStart	parameter inStart used in Fisheye_360_Standard()
inEnd	parameter inEnd used in Fisheye_360_Standard()
out_imgH	parameter out_imgH used in Fisheye_360_Standard()
out_imgWPt	the recommended width for Fisheye_360_Standard()

Return Value:

<0	error
0	success

Fisheye_360_Standard_Get_OutImgH

Get a recommended height for Fisheye_360_Standard() to avoid the output image is out of shape.

Function Prototype:

```
int Fisheye_360_Standard_Get_OutImgH(int hd, int beta, int inStart, int inEnd, int
out_imgW, int * out_imgHPt);
```

hd	handle of fisheye image processing object
beta	parameter beta used in Fisheye_360_Standard()
inStart	parameter inStart used in Fisheye_360_Standard()
inEnd	parameter inEnd used in Fisheye_360_Standard()
out_imgW	parameter out_imgW used in Fisheye_360_Standard()
out_imgHPt	the recommended height for Fisheye_360_Standard()

Return Value:

<0	error
0	success

Fisheye_360_Mercator

Use Mercator projection to process the fisheye image.

Function Prototype:

```
int Fisheye_360_Mercator (int hd, int alpha, int beta, int isClockwise,
    const BITMAPINFOHEADER * input_header, const unsigned char *input_bmp,
    BITMAPINFOHEADER * output_header, unsigned char *output, int out_imgW,
    int out_imgH);
```

Hd	handle of fisheye image processing object
alpha	the angle, a, from 12 o'clock
beta	the degree, b, of circular sector needs to be un-circled.
isClockwise	1: Un-circle in clockwise; 0: Un-circle in counter clockwise
input_header	info header of the input fisheye image.
input_bmp	bitmap of the input fisheye image.
output_header	info header of the 360 mode output image.
output	bitmap of the 360 mode output image.
out_imgW	the width of the 360 mode output image.
out_imgH	the height of the 360 mode output image.

Return Value:

<0	error
0	success

Wall-mounted Related function

Fisheye_Wall_Dewarp

To dewarp a wall-mounted fisheye image.

Function Prototype:

```
int Fisheye_Wall_Dewarp(int hd, int longitude, int latitude, float zoom,
    const BITMAPINFOHEADER * input_header, const unsigned char *input_bmp,
    BITMAPINFOHEADER * output_header, unsigned char *output, int out_imgW,
    int out_imgH);
```

hd	handle of fisheye image processing object.
longitude	the longitude of interest position. The recommended value can be obtained by using Fisheye_GetPosFromOutImg() or Fisheye_GetPosValue()
latitude	the latitude of interest position. The recommended value can be obtained by using Fisheye_GetPosFromOutImg() or Fisheye_GetPosValue()
zoom	zoom ratio of the dewarping output image.
input_header	info header of the input fisheye image.
input_bmp	bitmap of the input fisheye image.
output_header	info header of the dewarping image.
output	bitmap of the dewarping image.
out_imgW	the width of the output dewarping image.
out_imgH	the height of the output dewarping image.

Return Value:

<0	error
0	success

See also:

Fisheye_GetPosFromOutImg()

Fisheye_GetPosValue()

Fisheye_GetPosFromOutImg

Get recommended longitude & latitude value for Fisheye_Wall_Dewarp() from a processed image and point (x, y). The point (x, y) will be the center of new image generated by Fisheye_Wall_Dewarp().

Function Prototype:

```
int Fisheye_GetPosFromOutImg(int hd, int x, int y, int *longitudePt, int * latitudePt);
```

hd	handle of fisheye image processing object
x	x coordinate of the processed image
y	y coordinate of the processed image
longitudePt	the recommended longitude for Fisheye_Wall_Dewarp()
latitudePt	the recommended latitude for Fisheye_Wall_Dewarp()

Return Value:

<0	error
0	success

Fisheye_GetPosValue

Get recommended longitude & latitude value for Fisheye_Wall_Dewarp() from a fisheye bitmap and a interested point (x, y) of bitmap. The point (x, y) will be the center of new image generated by Fisheye_Wall_Dewarp().

Function Prototype:

```
int Fisheye_GetPosValue(int hd, int x, int y, int *longitudePt, int * latitudePt);
```

hd	handle of fisheye image processing object
x	x coordinate of the bitmap
y	y coordinate of the bitmap
longitudePt	the recommended pan value for Fisheye_Wall_Dewarp()
latitudePt	the recommended tilt value for Fisheye_Wall_Dewarp()

Return Value:

<0	error
0	success

Fisheye_Wall_180_Panorama

Generate the 180o panorama image from wall-mounted fisheye image.

Function Prototype:

```
int Fisheye_Wall_180_Panorama(int hd, const BITMAPINFOHEADER * input_header,
const unsigned char *input_bmp, BITMAPINFOHEADER * output_header,
unsigned char *output, int out_imgW, int out_imgH);
```

hd	handle of fisheye image processing object
input_header	info header of the input fisheye image.
input_bmp	bitmap of the input fisheye image.
output_header	info header of the panorama image.
output	bitmap of the panorama image.
out_imgW	the width of the output panorama image.
out_imgH	the height of the output panorama image.

Return Value:

<0	error
0	success

Other (misc.)

Fisheye_SetOptionInt

To set some fisheye object related parameters.

Function Prototype:

```
int Fisheye_SetOptionInt(int hd, const char * paramName, int value);
```

hd	handle of fisheye image processing object
paramName	parameter name
value	to set parameter to the value

paramName	valid value
“WALL_ROTATE_BIAS”	-180~180
“OUTPUT_BITMAP_ORIENTATION”	FISHEYE_BOTTOMUP, FISHEYE_TOPDOWN

```
enum { FISHEYE_BOTTOMUP = 0, FISHEYE_TOPDOWN = 1};
```

WALL_ROTATE_BIAS

To adjust the rotation degree of wall-mount fisheye camera.
While install the fisheye camera with wall-mount type, there may be a bias that causes Fisheye_Wall_180_Panorama provide non-horizontal image. So, WALL_ROTATE_BIAS can correct this problem.
Note:
> 0 : clockwise rotation
< 0 : counter-clockwise rotation

OUTPUT_BITMAP_ORIENTATION

Application of dll may include different kinds of bitmap, some bitmaps are from bottom to top (default), but some are from top to bottom.
These parameters can only be applied to the output bitmap. (For more detailed information, please refer to the bitmap orientation chapter)

Different kinds of Bitmap

In Fisheye SDK V1002 and above
The dll support RGB24, RGB32 and YUY2, three kinds of bitmap.

To use different kind of fisheye bitmap
The input bitmap header (const BITMAPINFOHEADER * input_header) into the dll should be set properly:
RGB24:
input_header->biCompression = BI_RGB;
input_header->biBitCount = 24;

RGB32:
input_header->biCompression = BI_RGB;
input_header->biBitCount = 32;

YUY2:
input_header->biCompression = MAKEFOURCC(‘Y’, ‘U’, ‘Y’, ‘2’);
input_header->biBitCount = 16;

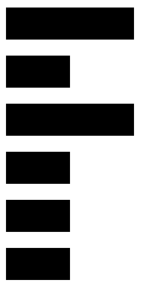
The Bitmap Orientation

Bottom-Up image:

There are some image pixel array definition is from bottom to top.
The small index of pixelArray(const unsigned char *input_bmp) elements are those

pixels at the bottom left of the image.

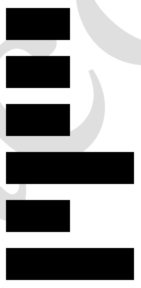
For example:

The bitmap array:	The image is something like:
011000 011000 011000 011110 011000 011110	

Top-Down image:

The small index of pixelArray(const unsigned char *input_bmp) elements are those pixels at the top left of the image.

For example:

The same bitmap array above:	The image is something like:
011000 011000 011000 011110 011000 011110	

Use of SDK & different kind of bitmap orientation

The image input into fisheye dll can be bottom-up image/top-down image.

The biHeight of BITMAPINFOHEADER should be set properly to let fisheye dll recognize the input bitmap orientation.

If the input bitmap array is bottom-up, the biHeight has to be set as an integer greater than 0.

e.g.

The input bitmap array is bottom-up, and the image height is 1944.

input_header->biHeight = 1944;

If the input bitmap array is top-down, the biHeight has to be set as an integer less than 0.

e.g.

The input bitmap array is top-down, and the image height is 1944.

input_header->biHeight = -1944;

Use Fisheye_SetOptionInt() to set the “OUTPUT_BITMAP_ORIENTATION” parameters to get bottom-up/top-down output image from fisheye dll.

.

```
int Fisheye_SetOptionInt(int hd, const char * paramName, int value);
```

paramName: “OUTPUT_BITMAP_ORIENTATION”

Value:	Description:
FISHEYE_BOTTOMUP	the output image from fisheye dll is bottom-up bitmap
FISHEYE_TOPDOWN	the output image from fisheye dll is top-down bitmap

Note:

Set the fisheye parameter “OUTPUT_BITMAP_ORIENTATION” as FISHEYE_TOPDOWN, the output pixel array (unsigned char *output) will be from top to bottom.
And the image header(BITMAPINFOHEADER * output_header) biHeight should be less than 0. (output_header->biHeight < 0)

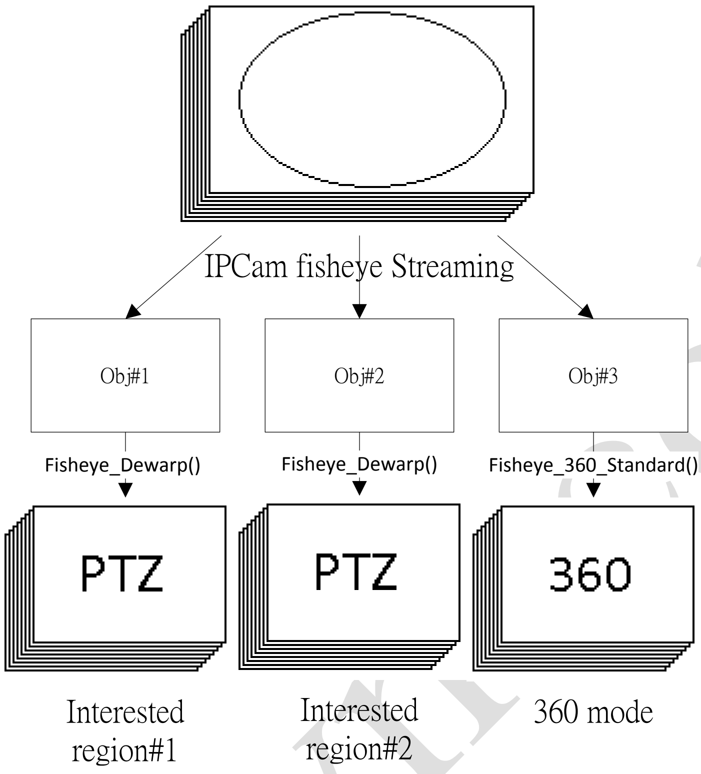
Note:

If the top-down bitmap Input with biHeight < 0 and don’t set the “OUTPUT_BITMAP_ORIENTATION” parameters, the output fisheye bitmap will be bottom-up bitmap(by default).

● Performance Issue

If more than one interested region (PTZ or 360 mode) is needed, use more image processing objects to shorten the processing time.

One specific object for one region (the same pan, tilt, zoom parameters) and one size (same BMP image size) to process the fisheye bmp streaming (many BMP images) can save the processing time.


IPCam fisheye Streaming
Obj#1
Obj#2
Obj#3
Fisheye_Dewarp()
Fisheye_Dewarp()
Fisheye_360_Standard()
PTZ
PTZ
360
Interested region#1
Interested region#2
360 mode