VNDP VIVOTEK Network Development Platform

VIVOTEK NETWORK DEVELOPMENT PLATFORM

fisheyesw

Version 3.2.1.0

2017/08/01

REVISION HISTORY

Revision	Date	Description
3.2.0.0	2016/11/02	• Support bilinear interpolation
3.2.1.0	2017/08/01	• Support FE9182, FE9382, FE9191, and FE9391 • Support panorama tilt control in wall mount

TABLE OF CONTENTS

1. Overview

1.1 Introduction

This manual document describes the properties and methods supported by the VIVOTEK fisheyesw module.

1.2 Getting Started with fisheyesw Module

The main function of fisheyesw module is to dewarp the video frames which come from VIVOTEK fisheye camera.

1.3 File Structure

Table 1- File Structure

File	Description
$(SDKPATH)\doc\	This manual document
$(SDKPATH)\include\	Header files
$(SDKPATH)\lib\x86\	Binary files and import library files (32-bit)
$(SDKPATH)\lib\x64\	Binary files and import library files (64-bit)
$(SDKPATH)\sample	Sample projects
$(SDKPATH)\sample\image	Test images

2. Programmer's Guide

2.1 Using fisheyesw Module

Programmers can use fisheyesw module to dewarp video frames which are captured from VIVOTEK fisheye camera. The fisheyesw module supports both rectilinear and panorama projections as dewarp methods. Programmers can easily combine different dewarped contents to provide flexible layout of scenes in surveillance environment.

This module is cross-platform compatible on Windows(x86/x64), MacOS(x86/x64), and Linux(x86/x64). The minimum system requirement of Windows platform is Windows XP. The minimum system requirement of MacOS is OS X 10.6.

Enable VIVOTEK Fisheye Camera Watermark

The fisheyesw module only supports video frames that contain VIVOTEK watermark information. The watermark information is composed of white pixels on the bottom left and right corners of the video frame.

If there is no watermark on video frame, please send the following CGI command with VIVOTEK camera web browser to enable watermark: http://CAMERA_IP/cgi-bin/admin/setparam.cgi?videoin_c0_enablewatermark=1

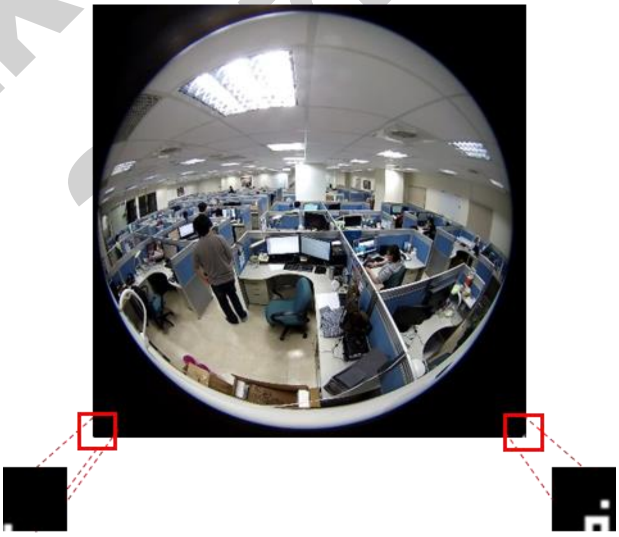


Figure 2-1 VIVOTEK watermark information

Dewarp using Rectilinear Projection

The rectilinear projection preserves more geometry characteristics view of objects in the scene.

After initializing a fisheyesw module, you have to choose a mount type of fisheye camera. Set FE_DEWARP_RECTILINEAR as the dewarp type and you will get the dewarp scene using rectilinear projection.

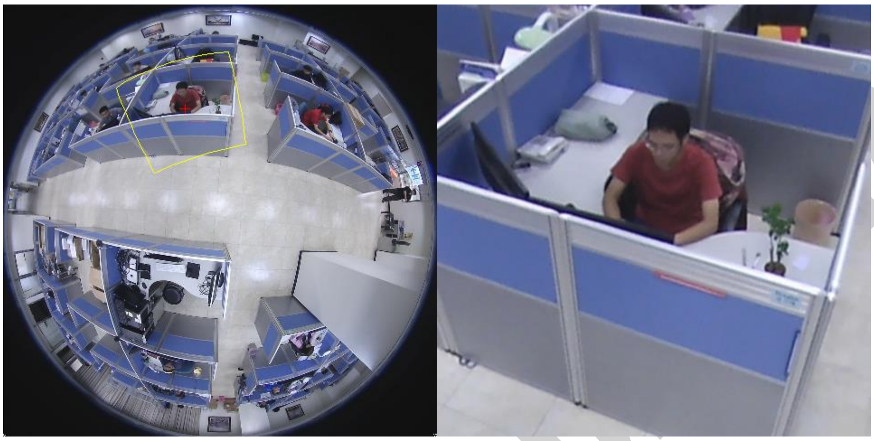


Figure 2-2 Rectilinear projection

Dewarp using Panorama Projection

The panorama projection provides near-complete view in the scene.

After initializing a fisheyesw module, you have to choose a mount type of fisheye camera. Set FE_DEWARP_FULLVIEWPANORAMA as the dewarp type and you can get the dewarp scene using panorama projection. Moreover, the fisheyesw module provides a dual-view panorama projection in ceiling and floor mount type. You can use FE_DEWARP_DUALVIEWPANORAMA as the as the dewarp type to get the dual-view panorama.

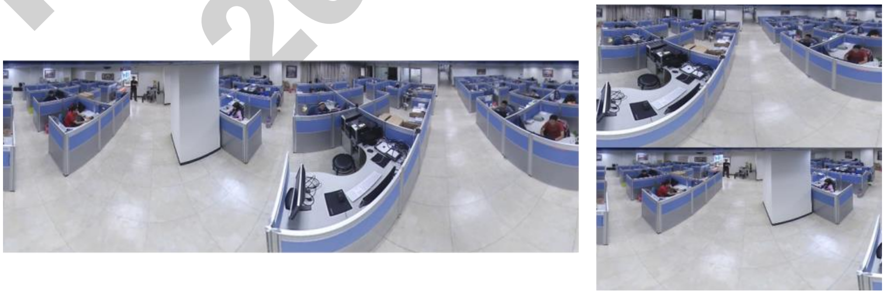


Figure 2-3 Panorama projection (Left: full-view panorama, Right: dual-view panorama)

3. Sample Code

This section contains sample projects provided by fisheyesw.

To demonstrate how to use the fisheyesw library and fisheye ePTZ control, we use Simple DirectMedia Layer (SDL2) library to access keyboard and mouse event. Also, we use it to present the dewarped scene on the window. As a result, please check the SDL2 library is available on your platform before running these samples.

All samples are cross-platform on Windows, MacOS and Linux, but only support x86 platform.

How to Run the Sample

Please check the SDL2 library is available on your platform before running these samples.

Windows

1. Launch Visual Studio.
2. Open the $(SAMPLENAME).sln solution under $(SDKPATH)\sample\$(SAMPLENAME).
3. Build the solution and run the sample.

Mac OS

1. Launch XCode and open $(SAMPLENAME).xcodeproj under $(SDKPATH)\sample\$(SAMPLENAME).
2. Build the solution and run the sample.

Linux

1. Locate the folder to $(SDKPATH)\sample\$(SAMPLENAME)\vcproj.
2. Make the file than type "./build.sh".
3. The sample will run after being successfully built.

3.1 dewarp_fisheyemode

Description

This sample loads a fisheye Bitmap image and displays a rectilinear or panorama dewarped scene on client window. It also shows how to use keyboard and mouse event to perform ePTZ control with new APIs.

Sample Code

Step 1. Initialize fisheyesw module

```
Fisheye_Initial(&hFECtx, LIBFISHEYE_VERSION);
```

Step 2. Set the options

In this sample, we use a SDL surface from Bitmap as the input and allocate an output buffer. First of all, we initialize the values of fisheye option. Secondly, we set the dewarp type and the mount type of fisheye camera. After fill these fields in option, you have to set the option flags to indicate what specific fields are filled with new values. And use Fisheye_SetOption to update the option. Finally, set default ePTZ position by Fisheye_SetPanTiltZoom with absolute position flag.

Step 3. Event loop

According to the keyboard and mouse event, we perform the ePTZ control by Fisheye_SetPanTiltZoom with relative position flag. Then, use Fisheye_OneFrame to dewarp the input image and update the dewarped scene to the output buffer. Here, in order to show the dewarped scene, we update SDL texture with new output pixel data and use SDL render to present the scene on client window.

Step 4. Release fisheyesw module

```
Fisheye_Release(&hFECtx);
```

Tips

1. The pixel format of InVPicture and OutVPicture should be the same.
2. Before first Fisheye_OneFrame is called, InVPicture, OutVPicture, FOVCenter, FOVRadius and OutROI fields must be set with valid values.
3. Users can obtain fisheye streaming from VIVOTEK fisheye camera by setting Media->Video->FOV as Fisheye mode.

3.2 dewarp_fullhdmode

Description

This sample loads a Full-HD Bitmap image and displays a clip-view panorama dewarped scene on client window.

The ePTZ control is not available in this dewarp type.

Sample Code

Step 1. Initialize fisheyesw module

```
Fisheye_Initial(&hFECtx, LIBFISHEYE_VERSION);
```

Step 2. Set the options

In this sample, we use a SDL surface from Bitmap as the input and allocate an output buffer. First of all, we initialize the values of fisheye option. Secondly, we set the mount type of fisheye camera and FE_DEWARP_CLIPVIEWPANORAMA as the dewarp type. After fill these fields in option, you have to set the option flags to indicate what specific fields are filled with new values. Finally, use Fisheye_SetOption to update the option.

Step 3. Event loop

After applying these parameters via Fisheye_SetOption, use Fisheye_OneFrame to dewarp the input image and update the dewarped scene to the output buffer. Here, in order to show the dewarped scene, we update SDL texture with new output pixel data and use SDL render to present the scene on client window.

Step 4. Release fisheyesw module

```
Fisheye_Release(&hFECtx);
```

Tips

1. The pixel format of InVPicture and OutVPicture should be the same.
2. Before first Fisheye_OneFrame is called, InVPicture, OutVPicture, FOVCenter, FOVRadius and OutROI fields must be set with valid values.
3. ePTZ control is not available in clip panorama projection.
4. The Width and Height fields of InVPicture must be 1920 and 1080.
5. Users can obtain fisheye streaming from VIVOTEK fisheye camera by setting Media->Video->FOV as 1080P Full HD.

3.3 dewarp_4RQuadView

Description

This sample loads a Bitmap image and displays a quad view with four rectilinear dewarped scenes on client window. It also shows how to use mouse event to perform ePTZ control.

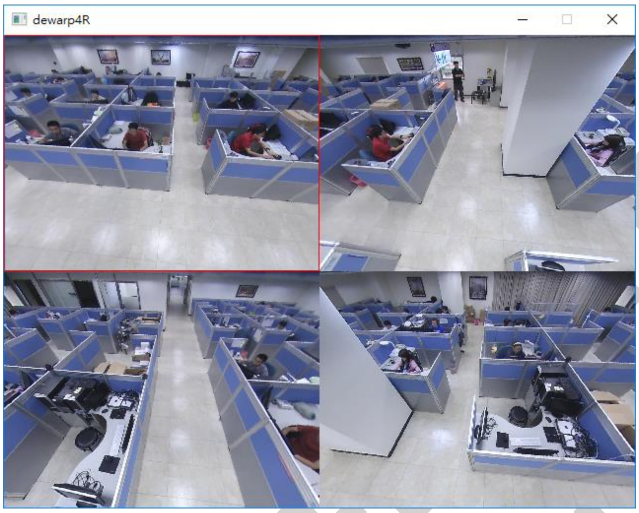


Figure 3-1 Quad view of rectilinear projection

Sample Code

Step 1. Initialize four fisheyesw module

```
Fisheye_Initial(&hFECtx[i], LIBFISHEYE_VERSION);
```

Step 2. Set the options

In this sample, we use a SDL surface from Bitmap as the input and allocate an output buffer. First of all, we initialize the values of fisheye option. Secondly, we set the mount type of fisheye camera and FE_DEWARP_RECTILINEAR as the dewarp type. After fill these fields in option, you have to set the option flags to indicate what specific fields are filled with new values and use Fisheye_SetOption to update the option. Finally, set default ePTZ position of each view by Fisheye_SetPanTiltZoom with absolute position flag.

Step 3. Event loop

In the event loop, this sample uses mouse event to control the Pan, Tilt and Zoom of the focused view. After applying these parameters via Fisheye_SetPanTiltZoom, use Fisheye_OneFrame to dewarp the input image and update four dewarped scenes to their OutROIs in output buffer. Here, in order to show the dewarped scene, we update SDL texture with new output pixel data and use SDL render to present the scene on client window.

Step 4. Release fisheyesw module

```
Fisheye_Release(&hFECtx);
```

Tips

1. The pixel format of InVPicture and OutVPicture should be the same.
2. Before first Fisheye_OneFrame is called, InVPicture, OutVPicture, FOVCenter, FOVRadius and OutROI fields must be set with valid values.
3. The source fisheye image of four views are the same, so InVPicture, OutVPicture, FOVCenter, FOVRadius, DewarpType and MountType fields are also the same.
4. OutROI field must be adjusted by each view location.

3.4 dewarp_OcontrolR

Description

This sample loads a Bitmap image and demonstrates how to control pan and tilt of rectilinear dewarped scenes(R) by the point in original fisheye scene (O). It also shows how to use mouse event to perform ePTZ control.

Sample Code

Step 1. Initialize fisheyesw module

```
Fisheye_Initial(&hFECtx, LIBFISHEYE_VERSION);
```

Step 2. Set the options

In this sample, we use a SDL surface from Bitmap as the input and allocate an output buffer. First of all, we initialize the values of fisheye option. Secondly, we set the mount type of fisheye camera and FE_DEWARP_RECTILINEAR as the dewarp type. After fill these fields in option, you have to set the option flags to indicate what specific fields are filled with new values and use Fisheye_SetOption to update the option. Finally, set default ePTZ position of each view by Fisheye_SetPanTiltZoom with absolute position flag.

Step 3. Event loop

In the event loop, we use mouse event to retrieve the point in original fisheye scene (O). Then use Fisheye_InVPicturePointToPanTilt to get pan and tilt of rectilinear dewarp type. After applying these parameters via Fisheye_SetPanTiltZoom, use Fisheye_OneFrame to dewarp the input image and update the dewarped scenes(R) to the output buffer. Here, in order to show the dewarped scene, we update SDL texture with new output pixel data and use SDL render to present the scene on client window.

Step 4. Release fisheyesw module

```
Fisheye_Release(&hFECtx);
```

Tips

1. The pixel format of InVPicture and OutVPicture should be the same.
2. Before first Fisheye_OneFrame is called, InVPicture, OutVPicture, FOVCenter, FOVRadius and OutROI fields must be set with valid values.

3.5 dewarp_OcontrolR2

Description

This sample loads a Bitmap image and demonstrates how to control pan, tilt and zoom of rectilinear dewarped scenes(R) by the rectangle region in original fisheye scene (O). It also shows how to use mouse event to perform ePTZ control.

Sample Code

Step 1. Initialize fisheyesw module

```
Fisheye_Initial(&hFECtx, LIBFISHEYE_VERSION);
```

Step 2. Set the options

In this sample, we use a SDL surface from Bitmap as the input and allocate an output buffer. First of all, we initialize the values of fisheye option. Secondly, we set the mount type of fisheye camera and FE_DEWARP_RECTILINEAR as the dewarp type. After fill these fields in option, you have to set the option flags to indicate what specific fields are filled with new values and use Fisheye_SetOption to update the option. Finally, set default ePTZ position of each view byFisheye_SetPanTiltZoom with absolute position flag.

Step 3. Event loop

In the event loop, we use mouse event to retrieve the point in original fisheye scene(O). Then use Fisheye_InVPicturePolygonToPanTiltZoom to get pan, tilt and zoom of rectilinear dewarp type. After applying these parameters via Fisheye_SetPanTiltZoom, use Fisheye_OneFrame to dewarp the input image and update the dewarped scenes(R) to the output buffer. Here, in order to show the dewarped scene, we update SDL texture with new output pixel data and use SDL render to present the scene on client window.

Step 4. Release fisheyesw module

```
Fisheye_Release(&hFECtx);
```

Tips

1. The pixel format of InVPicture and OutVPicture should be the same.
2. Before first Fisheye_OneFrame is called, InVPicture, OutVPicture, FOVCenter, FOVRadius and OutROI fields must be set with valid values.
3. The following picture shows a sample result of mouse event control. When mouse is clicked and dragged on the left side of the display window, there is a blue rectangle showing a wanted dewarped region. When release the mouse button, the dewarped scenes(R) is updated according to the ePTZ parameters retrieved from Fisheye_InVPicturePolygonToPanTiltZoom. The yellow polygon is the actual dewarped

region of the fisheye image. Each point of the yellow polygon is retrieved from Fisheye_OutRoiPointToInVPicturePoint.

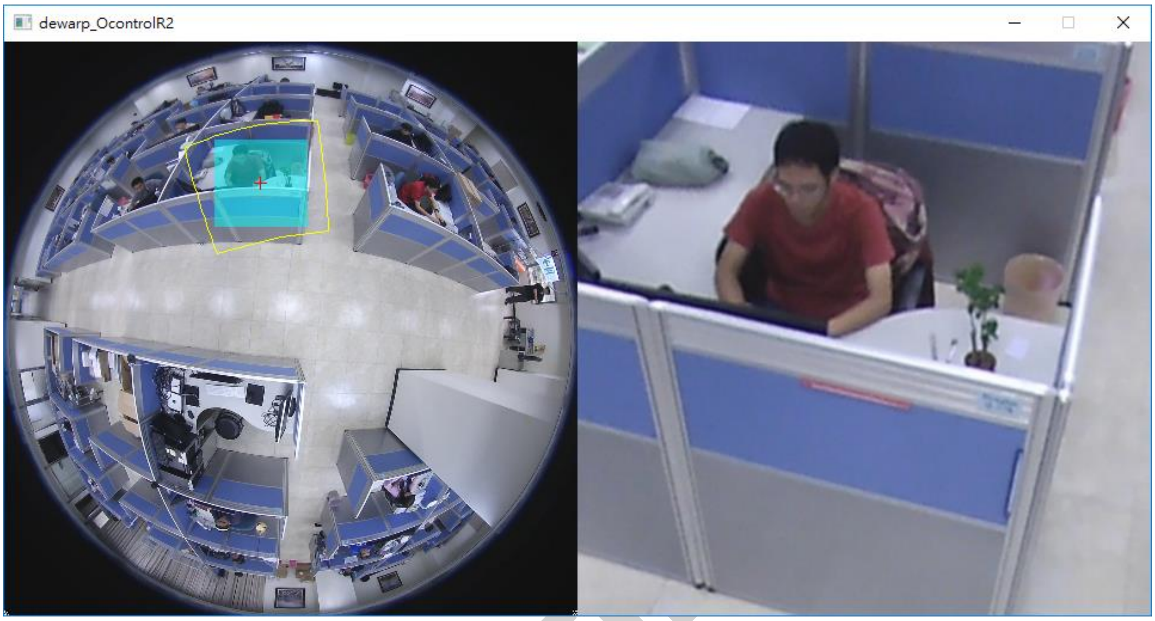


Figure 3-2 Polyons on O control position of R

4. API Reference

This chapter contains the API functions and elements provided by fisheye.

4.1 Enumeration

This section depicts the fisheye enumerations.

- FEDEWARPTYPE
- FEMOUNTTYPE
- FEOPTIONFLAG
- FEVPIXELFORMAT
- FEPTZPOSITIONFLAG
- FEFILTERTYPE

4.1.1 FEDEWARPTYPE

This enumeration specifies the dewarp type.

```
typedef enum _FisheyeDewarpType {
                                        FE_DEWARP_RECTILINEAR,
                                        FE_DEWARP_FULLVIEWPANORAMA,
                                        FE_DEWARP_DUALVIEWPANORAMA,
                                        FE_DEWARP_CLIPVIEWPANORAMA,
                                        FE_DEWARP_AERIALVIEW
} FEDEWARPTYPE;
```

Values

FE_DEWARP_RECTILINEAR

Specifies the dewarp type is rectilinear projection.

FE_DEWARP_FULLVIEWPANORAMA

Specifies the dewarp type is panorama projection. In ceiling and floor mount, the output is a single-view content with 360 degrees in horizontal. In wall mount, the output is a single-view content with 180 degrees in horizontal.

FE_DEWARP_DUALVIEWPANORAMA

Specifies the dewarp type is panorama projection. In ceiling and floor mount, the output is a dual-view content with 180 degrees per view in horizontal. In wall mount, the output is the same as FE_DEWARP_FULLVIEWPANORAMA.

FE_DEWARP_CLIPVIEWPANORAMA

Specifies the dewarp type is clipped panorama projection. The output is a single-view content with 180 degrees in horizontal.

FE_DEWARP_AERIALVIEW

Specifies the dewarp type is aerial view projection. The output is a virtual fisheye content.

Remarks

1. The FE_DEWARP_CLIPVIEWPANORAMA is a specific dewarp type for the 1080P Full HD streaming of fisheye camera. FE_DEWARP_CLIPVIEWPANORAMA is similar to FE_DEWARP_FULLVIEWPANORAMA in wall mount. However, it is designed only for the image which comes from Full HD streaming. Users should provide the Full HD video streaming to get correct dewarped result. The ePTZ control will be disabled in FE_DEWARP_CLIPVIEWPANORAMA.
2. The output of FE_DEWARP_AERIALVIEW is a virtual fisheye content which is similar to three-dimensional fisheye. By increasing zoom value, the scene will change from a virtual fisheye view to a rectilinear projection view.

Requirements

fisheyesw.h

4.1.2 FEMOUNTTYPE

This enumeration specifies the mount type of the installed fisheye camera.

```
typedef enum _FisheyeMountType {
                                        FE_MOUNT_WALL,
                                        FE_MOUNT_CEILING,
                                        FE_MOUNT_FLOOR,
} FEMOUNTTYPE;
```

Values

FE_MOUNT_WALL

Specifies the mount type is wall-mounted. The horizontal and vertical fields of view (FOV) are 180 degrees.

FE_MOUNT_CEILING

Specifies the mount type is ceiling-mounted. The horizontal FOV 360 degrees and vertical FOV is 90 degrees.

FE_ MOUNT_FLOOR

Specifies the mount type is floor-mounted. The horizontal FOV 360 degrees and vertical FOV is 90 degrees.

Remarks

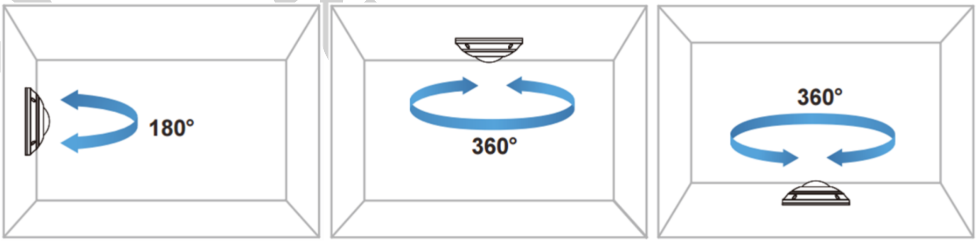


Figure 4-1 MountType (From left to right: wall mount, ceiling mount, floor mount)

Requirements

fisheyesw.h

4.1.3 FEOPTIONFLAG

This enumeration is a set or bit flags that specify options in fisheyesw module.

```
typedef enum _ FisheyeOptionFlag{
                    FE_OPTION_INIMAGEHEADER      = (1 << 0),
                    FE_OPTION_INIMAGEBUFFER      = (1 << 1),
                    FE_OPTION_OUTIMAGEHEADER     = (1 << 2),
                    FE_OPTION_OUTIMAGEBUFFER     = (1 << 3),
                    FE_OPTION_FOVCENTER          = (1 << 4),
                    FE_OPTION_FOVRADIUS          = (1 << 5),
                    FE_OPTION_MOUNTTYPE          = (1 << 6),
                    FE_OPTION_DEWARPTYPE         = (1 << 7),
                    FE_OPTION_YAW                = (1 << 8),
                    FE_OPTION_PITCH              = (1 << 9),
                    FE_OPTION_ROLL               = (1 << 10),
                    FE_OPTION_SCALE              = (1 << 11),
                    FE_OPTION_SCROLL             = (1 << 12),
                    FE_OPTION_OUTROI              = (1 << 13),
} FEOPTIONFLAG;
```

Values

FE_OPTION_INIMAGEHEADER

Specifies the InVPicture structure’s header fields.

FE_OPTION_INIMAGEBUFFER

Specifies the InVPicture structure’s buffer fields.

FE_OPTION_OUTIMAGEHEADER

Specifies the OutVPicture structure’s header fields.

FE_OPTION_OUTIMAGEBUFFER

Specifies the OutVPicture structure’s buffer fields.

FE_OPTION_FOVCENTER

Specifies the fisheye center in the field of view(FOV).

FE_OPTION_FOVRADIUS

Specifies the fisheye radius in the field of view(FOV).

FE_OPTION_MOUNTTYPE

Specifies the mount type.

FE_OPTION_DEWARPTYPE

Specifies the dewarp type.

FE_OPTION_ YAW (deprecated)

Specifies the yaw angle. This flag is obsolete and will be removed in one of the future releases. Use pan value Fisheye_SetPanTiltZoom instead of this option.

FE_OPTION_PITCH (deprecated)

Specifies the pitch angle. This flag is obsolete and will be removed in one of the future releases. Use tilt value of API Fisheye_SetPanTiltZoom instead of this option.

FE_OPTION_ ROLL (deprecated)

Specifies the roll angle. This flag is obsolete and will be removed in one of the future releases. Use pan value of Fisheye_SetPanTiltZoom instead of this option.

FE_OPTION_ SCALE (deprecated)

Specifies the scaling factor. This flag is obsolete and will be removed in one of the future releases. Use zoom value of Fisheye_SetPanTiltZoom instead of this option.

FE_OPTION_ SCROLL (deprecated)

Specifies the scroll angle. This flag is obsolete and will be removed in one of the future releases. Use zoom value of Fisheye_SetPanTiltZoom instead of this option.

FE_OPTION_OUTROI

Specifies the output region-of interest(ROI).

Remarks

You can combine the bit flags using bitwise-OR.

Requirements

fisheyesw.h

See Also

FEOPTION

4.1.4 FEVPIXELFORMAT

This enumeration specifies the color format for each pixel in the picture.

```
typedef enum _FisheyeVPixelFormat{
                                        FE_PIXELFORMAT_YUV420P,
                                        FE_PIXELFORMAT_RGB32,
} FEVPICTUREFORMAT;
```

Values

FE_PIXELFORMAT_YUV420P

Specifies the format is a YUV4:2:0 planner format. Y, U and V components are grouped together.

FE_PIXELFORMAT_RGB32

Specifies the format is 32bit per pixel; each 8 bits are used in red, green and blue components. The rest 8bits are not used.

Remarks

Requirements

fisheyesw.h

4.1.5 FEPTZPOSITIONFLAG

This enumeration specifies the ePTZ position type.

```
typedef enum _FisheyePTZPositionFlag{
                                        FE_POSITION_ABSOLUTE,
                                        FE_POSITION_RELATIVE,
} FEPTZPOSITIONFLAG;
```

Values

FE_POSITION_ABSOLUTE

Specifies the ePTZ values are an absolute position.

FE_POSITION_RELATIVE

Specifies the ePTZ values are a relative position.

Remarks

Requirements

fisheyesw.h

4.1.6 FEFILTERTYPE

This enumeration specifies the image filter type.

```
typedef enum _FisheyeFilterType{
                                        FE_FILTER_POINT,
                                        FE_FILTER_BILINEAR,
} FEFILTERTYPE;
```

Values

FE_FILTER_POINT

Specifies the filter type is nearest-neighbor interpolation.

FE_FILTER_BILINEAR

Specifies the filter type is bilinear interpolation.

Remarks

Requirements

fisheyesw.h

4.2 Data Structure

This section depicts the fisheyesw structure.

- FEOPTION
- FEPOINT
- FERECT
- FEVPICTURE

4.2.1 FEOPTION

This struct specifies options of fisheyesw module.

```
typedef struct _FisheyeOption {
                                    DWORD                   Flags;
                                    FEVPICTURE              InVPicture;
                                    FEVPICTURE              OutVPicture;
                                    FEPOINT                 FOVCenter;
                                    unsigned int            FOVRadius;
                                    FEMOUNTTYPE             MountType;
                                    FEDEWARPTYPE            DewarpType;
                                    float                   Yaw;
                                    float                   Pitch;
                                    float                   Roll;
                                    unsigned int            Scale;
                                    float                   Scroll;
                                    FERECT                  OutRoi;
} FEOPTION;
```

Members

Flags

Specifies the option flags. You can combine the bit flags which are defined in FEOPTIONFLAG using bitwise-OR.

InVPicture

Specifies fields to describe the input picture.

OutVPicture

Specifies fields to describe the output picture.

FOVCenter

Specifies the pixel coordinate of the center in the field of view(FOV) of the input picture.

FOVRadius

Specifies the radius of FOV in the input picture.

MountType

Specifies the mount type of fisheye camera. The values are defined in FEMOUNTTYPE.

DewarpType

Specifies the mount type of fisheye camera. The values are defined in FEDEWARPTYPE.

Yaw (deprecated)

Specifies the yaw angle(in degree) around the y-axis. It is only available in FE_MOUNT_WALL. This option is obsolete and will be removed in one of the future releases. Use pan value Fisheye_SetPanTiltZoom instead of this option.

Pitch (deprecated)

Specifies the pitch angle(in degree) around x-axis. This option is obsolete and will be removed in one of the future releases. Use tilt value Fisheye_SetPanTiltZoom instead of this option.

Roll (deprecated)

Specifies the pitch angle(in degree) around z-axis, which is straight toward the screen. It is available in FE_MOUNT_CEILING and FE_MOUNT_FLOOR. This option is obsolete and will be removed in one of the future releases. Use pan value Fisheye_SetPanTiltZoom instead of this option.

Scale (deprecated)

Specifies the scaling factor. The default value is 100, which means that 1x zoom and no scaling. You can use a number which is larger than 100 to perform scaling. For example, passing 120 will produce a 1.2x zoom-in effect. The minimum and maximum values are 70 and 1200. Scale is available in FE_DEWARP_RECTILINEAR. This option is obsolete and will be removed in one of the future releases. Use pan value Fisheye_SetPanTiltZoom instead of this option.

Scroll (deprecated)

Specifies the scroll offset (in degree) in the panorama projection. The default value is 0, which means no offset. You can adjust this parameter to change the start position of panorama projection. This option is obsolete and will be removed in one of the future releases. Use pan value Fisheye_SetPanTiltZoom instead of this option.

OutRoi

Specifies the destination rectangle in output picture.

Remarks

1. Users need to handle the input and output buffer pointers. Before updating the buffer pointers in InVPicture and OutVPicture, you need to allocate buffers with sufficient size. When leaving the program, free the above buffers by yourselves. The pixel format must be the same in InVPicture and OutVPicture.
2. The FOV means the circular region of the input picture which is captured by VIVOTEK fisheye lens. When the width and height of input picture are changed, you need to update the FOV's center and radius.
3. Yaw, Pitch, Roll and Scale are used for ePTZ control in rectilinear projection. Scroll is used for sliding control in panorama projection. The minimum and maximum values of these parameters are controlled by fisheyesw module.
4. If you want to display several dewarp results in one view, you can use OutRoi to control each output coordinate.
5. The following table shows the ralationships between deprecated options(Yaw, Pitch, Roll, Scroll and Scale) and new parameters(Pan, Tilt and Zoom) of Fisheye_SetPanTiltZoom API. The values in this table are the ranges of Pan, Tilt and Zoom.

Table 4- The ralationships between deprecated options and new parameters

DewarpType	MountType	Pan		Tilt		Zoom	
Rectilinear	Wall	Yaw	-90~90	Pitch	-90~90	Scale / 100	
	Ceiling	Roll	0~360		0~90		
	Floor				-90~0		
FullPanorama DullPanorama	Wall	N/A	0	N/A	0	N/A	1
	Ceiling	Scroll	0~360	N/A		N/A	
	Floor			N/A		N/A	
ClipPanorama	All	N/A	0	N/A	0	N/A	1

Requirements

fisheyesw.h

4.2.2 FEPOINT

This structure specifies a point in 2-D coordinate.

```
typedef struct _FisheyePoint{
                                        int                     X;
                                        int                     Y;
} FEPOINT;
```

Members

X

Specifies the x-coordinate of the point.

Y

Specifies the y-coordinate of the point.

Remarks

Requirements

fisheyesw.h

4.2.3 FERECT

This structure specifies a rectangle in 2-D coordinate.

```
typedef struct _FisheyeRect{
                                        int                     Left;
                                        int                     Top;
                                        int                     Right;
                                        int                     Bottom;
} FERECT;
```

Members

Left

Specifies the x-coordinate of the upper-left point of the rectangle.

Top

Specifies the y-coordinate of the upper-left point of the rectangle.

Right

Specifies the x-coordinate of the lower-right point of the rectangle.

Bottom

Specifies the y-coordinate of the lower-right point of the rectangle.

Remarks

Requirements

fisheyesw.h

4.2.4 FEVPICTURE

This structure specifies fields to describe a picture.

```
typedef struct _FisheyeVPicture {
                                   unsigned int        Width;
                                   unsigned int        Height;
                                   unsigned int        Stride;
                                   FEVPIXELFORMAT      Format;
                                   BYTE                *Buffer;
} FEVPICTURE;
```

Members

Width

Specifies the width (in pixels) of the picture.

Height

Specifies the height (in pixels) of the picture.

Stride

Specifies the number of bytes in one row of the picture.

Format

Specifies the picture format, which is one of values defined in FEVPIXELFORMAT.

***Buffer**

Specifies the pointer of data bits.

Requirements

fisheyesw.h

4.3 API Definition

This section depicts the fisheyesw APIs.

- Fisheye_Initial
- Fisheye_Release
- Fisheye_SetOption
- Fisheye_OneFrame
- Fisheye_SetModelName
- Fisheye_SetPanTiltZoom
- Fisheye_GetPanTiltZoom
- Fisheye_SetFilterType
- Fisheye_GetFilterType
- Fisheye_InVPicturePointToPanTilt
- Fisheye_PanTiltToInVPicturePoint
- Fisheye_InVPicturePolygonToPanTiltZoom
- Fisheye_OutRoiPointToInVPicturePoint
- Fisheye_PositionToYawPitchRoll (deprecated)

4.3.1 Fisheye_Initial

Call this function to initialize the fisheyesw module.

Syntax

```
SCODE Fisheye_Initial(
                                        HANDLE                  *phObject,
                                        DWORD                   dwVersion
);
```

Parameters

***phObject**

[out] Pointer to the handle of fisheye object.

dwVersion

[in] The version of fisheyesw library. Use **LIBFISHEYE_VERSION** as the parameter.

Return Values

FISHEYE_S_OK

Initialize the fisheyesw library successfully.

FISHEYE_E_INVALID_ARG

The pointer to handle of fisheye object is NULL.

FISHEYE_E_INVALID_VERSION

The versions of binary library and header file don’t match.

FISHEYE_E_OUT_OF_MEMORY

The library cannot acquire enough memory space.

FISHEYE_E_FAIL

Fail to initialize fisheyesw library.

Remarks

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.2 Fisheye_Release

Call this function to release the fisheyesw module.

Syntax

```
SCODE Fisheye_Release (
                                        HANDLE                  *phObject,
);
```

Parameters

***phObject**

[in] Pointer to the handle of fisheye object.

Return Values

FISHEYE_S_OK

Release the fisheyesw library successfully.

FISHEYE_E_INVALID_ARG

The pointer to handle of fisheye object is NULL.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

Remarks

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.3 Fisheye_SetOption

Call this function to set option fields.

Syntax

```
SCODE Fisheye_SetOption(
                                        HANDLE              hObject,
                                        FEOPTION            *pOption
);
```

Parameters

hObject

[in] The handle of fisheye object.

***pOption**

[in/out] Pointer to FEOPTION which contains the option fields. Yaw, Ptich, Roll, Scale and Scroll field will be modified if they are out of boundary.

Return Values

FISHEYE_S_OK

Set option fields successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer to FEOPTION is NULL. One or more option fields are invalid, see Remarks section.

FISHEYE_E_NOT_SUPPORT_PTZ

PTZ parameters (which are Yaw, Pitch, Roll, Scale and Scroll) are not supported if current dewarp mode is FE_DEWARP_CLIPVIEWPANORAMA.

FISHEYE_E_OUT_OF_MEMORY

The library cannot acquire enough memory space.

FISHEYE_E_FAIL

Fail to set option fields.

Remarks

1. InVPicture and OutVPicture size must be even if their formats are FE_PIXELFORMAT_YUV420P.
2. Yaw, Pitch, Roll, Scale and Scroll field will be modified if they are out of boundary.
3. Fisheye library only updates the field whose flag is on. Be careful when you set several fields of the option by combining the bit flags. Some fields might never be set into library if one of those fields sets failed.

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.4 Fisheye_OneFrame

Call this function to dewarp the fisheye picture.

Syntax

```
SCODE Fisheye_OneFrame (
                                        HANDLE                  hObject,
);
```

Parameters

hObject

[in] The handle of fisheye object.

Return Values

FISHEYE_S_OK

Dewarp successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_NOT_INITIALIZED_OPTION

One or more essential option fields are not provided. See Remarks section.

FISHEYE_E_WATERMARK_CHECK_FAIL

Fail to identify proprietary VIVOTEK watermark information. See Remarks section.

FISHEYE_E_FAIL

Fail to dewarp.

Remarks

1. The library needs to calculate internal parameters using the information of essential option fields. These options are InVPicture, OutVPicture, FOVCenter, FOVRadius, OutRoi and DewarpType. Before first Fisheye_OneFrame is called, these fields must have been set with valid values.
2. If incoming fisheye images do not have watermark, Fisheye_OneFrame will return **FISHEYE_E_WATERMARK_CHECK_FAIL** after 4 times calling. See 2.1 to enable watermark.
3. The OutVPicture buffer will be clear to zero when this API return **FISHEYE_E_WATERMARK_CHECK_FAIL**. You will see a green scene if the picture format is FE_PIXELFORMAT_YUV420P. And see a black scene if the picture format is FE_PIXELFORMAT_RGB32.

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.5 Fisheye_SetModelName

Call this function to set a model name.

Syntax

```
SCODE Fisheye_SetModelName (
                                    HANDLE                  hObject,
                                    const char              *pModelName
);
```

Parameters

hObject

[in] The handle of fisheye object.

***pModelName**

[in] The string of VIVOTEK camera model name.

Return Values

FISHEYE_S_OK

Set model name successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer to character is NULL. Or this model name is not supported, see Remarks section.

FISHEYE_E_FAIL

Fail to set model name.

Remarks

1. Model name is an optional parameter for fisheyesw library. The library will pick a general method for dewarping when user doesn't use this function.
2. Because each lens has its own specific optical parameters, user can use this function to help library choose appropriate dewarping method.
3. All of VIVOTEK fisheye camera model names are supported, including suffix word "-V" for VIVOTEK outdoor camera models.("FE8171", "FE8172", "FE8173", "FE8174", "SF8174", "FE8180", "FE8181", "FE8191", "FE8391", "FE9181", "FE9381", "FE9182", "FE9382", "FE9191", and "FE9391)

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

Example: `Fisheye_SetModelName(hObject, "FE8174");`

4.3.6 Fisheye_SetPanTiltZoom

Call this function to set pan, tilt and zoom.

Syntax

```
SCODE Fisheye_SetPanTiltZoom (
                                        HANDLE                  hObject,
                                        FEPTZPOSITIONFLAG       Flag,
                                        float                   Pan,
                                        float                   Tilt,
                                        float                   Zoom
);
```

Parameters

hObject

[in] The handle of fisheye object.

Flag

[in] The flag to specify the ePTZ values are relative positions or absolute positions. The value is defined in FEPTZPOSITIONFLAG.

Pan

[in] The pan angle. The range of absolute position depends on FEMOUNTTYPE and FEDEWARPTYPE, see Remarks section.

Tilt

[in] The tilt angle. The range of absolute position depends on FEMOUNTTYPE and FEDEWARPTYPE, see Remarks section.

Zoom

[in] The zoom factor. The range of absolute position is from 0.7 to 12 that means 0.7x to 12x, see Remarks section.

Return Values

FISHEYE_S_OK

Set pan, tilt and zoom successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

One of the input variables is not valid.

FISHEYE_E_FAIL

Fail to set pan, tilt and zoom.

Remarks

1. The following table shows the ranges of pan, tilt and zoom absolute positions.

Table 4- The range of ePTZ

DewarpType	MountType	Pan	Tilt	Zoom
Rectilinear	Wall	-90~90	-90~90	0.7~12
	Ceiling	0~360	0~90	
	Floor		-90~0	
FullPanorama DullPanorama	Wall	-40~40	0	1
	Ceiling	0~360		
	Floor			
ClipPanorama	All	0	0	1
AerialView	Wall	-90~90	-90~90	1~12
	Ceiling	0~360	0~90	
	Floor		-90~0	

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.7 Fisheye_GetPanTiltZoom

Call this function to get pan, tilt and zoom.

Syntax

```
SCODE Fisheye_GetPanTiltZoom(
                                        HANDLE          hObject,
                                        float           *pPan,
                                        float           *pTilt,
                                        float           *pZoom,
);
```

Parameters

hObject

[in] The handle of fisheye object.

***pPan**

[out] The pan angle. The range of absolute position depends on FEMOUNTTYPE and FEDEWARPTYPE, see Table 4-1.

***pTilt**

[out] The tilt angle. The range of absolute position depends on FEMOUNTTYPE and FEDEWARPTYPE, see Table 4-1.

***pZoom**

[out] The zoom factor. The range of absolute position is from 0.7 to 12 that means 0.7x to 12x, see Table 4-.

Return Values

FISHEYE_S_OK

Get pan, tilt and zoom successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL.

FISHEYE_E_FAIL

Fail to get pan, tilt and zoom.

Remarks

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.8 Fisheye_SetFilterType

Call this function to set filter type.

Syntax

```
SCODE Fisheye_SetFilterType (
                                        HANDLE                  hObject,
                                        FEFILTERTYPE            FilterType
);
```

Parameters

hObject

[in] The handle of fisheye object.

FilterType

[in] The interpolation filter type. The value is defined in FEFILTERTYPE.

Return Values

FISHEYE_S_OK

Set filter type successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

This filter type is not supported.

Remarks

1. Nearest-neighbor interpolation is used by default.

Requirements

fisheyesw.h, fisheyesw_errordef.h

4.3.9 Fisheye_GetFilterType

Call this function to get filter type.

Syntax

```
SCODE Fisheye_GetFilterType (
                                        HANDLE                  hObject,
                                        FEFILTERTYPE            *pFilterType
);
```

Parameters

hObject

[in] The handle of fisheye object.

***pFilterType**

[out] The interpolation filter type. The value is defined in FEFILTERTYPE.

Return Values

FISHEYE_S_OK

Set filter type successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL.

Requirements

fisheyesw.h, fisheyesw_errordef.h

4.3.10 Fisheye_InVPicturePointToPanTilt

Call this function to get the pan and tilt which are mapped from fisheye picture.

Syntax

```
SCODE Fisheye_InVPicturePointToPanTilt (
                                        HANDLE          hObject,
                                        int             X
                                        int             Y
                                        float           *pPan,
                                        float           *pTilt
);
```

Parameters

hObject

[in] The handle of fisheye object.

X

[in] The x-coordinate in InVPicture of fisheye image.

Y

[in] The y-coordinate in InVPicture of fisheye image.

***pPan**

[out] The pan angle.

***pTilt**

[out] The tilt angle.

Return Values

FISHEYE_S_OK

Get pan and tilt successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL. The (x, y) is out of InVPicutre.

FISHEYE_E_NOT_INITIALIZED_OPTION

One or more essential option fields are not provided. See Remarks section.

FISHEYE_E_FAIL

Fail to get position in InVPicture. See Remarks section.

Remarks

1. This function will get pan and tilt based on some essential option fields. They are InVPicture, FOVCenter, FOVRadius, MountType and DewarpTyp.
2. Not support panorama dewarp types.
3. (X, Y) must locate from (0, 0) to (InVPicture.Width, InVPicture.Height)

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

3.4 dewarp_OcontrolR

4.3.11 Fisheye_PanTiltToInVPicturePoint

Call this function to get the position of fisheye picture from specific pan and tilt.

Syntax

```
SCODE Fisheye_PanTiltToInVPicturePoint (
                                              HANDLE                  hObject,
                                              float                   Pan
                                              float                   Tilt
                                              int                     *pX,
                                              int                     *pY
);
```

Parameters

hObject

[in] The handle of fisheye object.

Pan

[in] The pan angle.

Tilt

[in] The tilt angle.

***pX**

[out] The x-coordinate in InVPicture of fisheye image.

***pY**

[out] The y-coordinate in InVPicture of fisheye image.

Return Values

FISHEYE_S_OK

Get x and y successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL.

FISHEYE_E_NOT_INITIALIZED_OPTION

One or more essential option fields are not provided. See Remarks section.

FISHEYE_E_FAIL

Fail to get position in InVPicture. See Remarks section.

Remarks

1. This funciton will get position based on some essential option fields. They are InVPicture, FOVCenter, FOVRadius, OutROI, MountType and DewarpTyp.
2. Not support panorama dewarp types.

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

4.3.12 Fisheye_InVPicturePolygonToPanTiltZoom

Call this function to get the pan, tilt and zoom from the polygon region of fisheye picture.

Syntax

```
SCODE
Fisheye_InVPicturePolygonToPanTiltZoom (     HANDLE          hObject,
                                             unsigned int    Count,
                                             FEPOINT         *pPolygon,
                                             float           *pPan,
                                             float           *pTilt,
                                             float           *pZoom
);
```

Parameters

hObject

[in] The handle of fisheye object.

Count

[in] The number of the input polygon points.

***pPolygon**

[in] The input polygon points in InVPicture of fisheye picture.

***pPan**

[out] The pan angle.

***pTilt**

[out] The tilt angle.

***pZoom**

[out] The zoom factor.

Return Values

FISHEYE_S_OK

Get pan, tilt and zoom successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL. One of points is out of InVPicutre.

FISHEYE_E_NOT_INITIALIZED_OPTION

One or more essential option fields are not provided. See Remarks section.

FISHEYE_E_FAIL

Fail to get values. See Remarks section.

Remarks

1. This funciton will get position based on some essential option fields. They are InVPicture, FOVCenter, FOVRadius, MountType and DewarpTyp.
2. Not support panorama dewarp types.
3. Polygon points must be in the region which is from (0, 0) to (InVPicture.Width, InVPicture.Height)

Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

3.5 dewarp_OcontrolR2

4.3.13 Fisheye_OutRoiPointToInVPicturePoint

Call this function to get the fisheye source point according to the OutRoi point.

Syntax

```
SCODE Fisheye_OutRoiPointToInVPicturePoint (
                                        HANDLE          hObject,
                                        int             Xo
                                        int             Yo
                                        int             *pXi,
                                        int             *pYi
);
```

Parameters

hObject

[in] The handle of fisheye object.

Xo

[in] The x-coordinate in output region(OutRoi).

Yo

[in] The y-coordinate in output region(OutRoi).

***pXi**

[out] The x-coordinate in InVPicture of fisheye image.

***pYi**

[out] The y-coordinate in InVPicture of fisheye image.

Return Values

FISHEYE_S_OK

Get Xi and Yi successfully.

FISHEYE_E_INVALID_HANDLE

The handle of fisheye object is NULL.

FISHEYE_E_INVALID_ARG

The pointer of the input variable is NULL. The (Xo, Yo) is out of boundary. (The ranges of Xo and Yo are 0 to width and height of OutROI.)

FISHEYE_E_NOT_INITIALIZED_OPTION

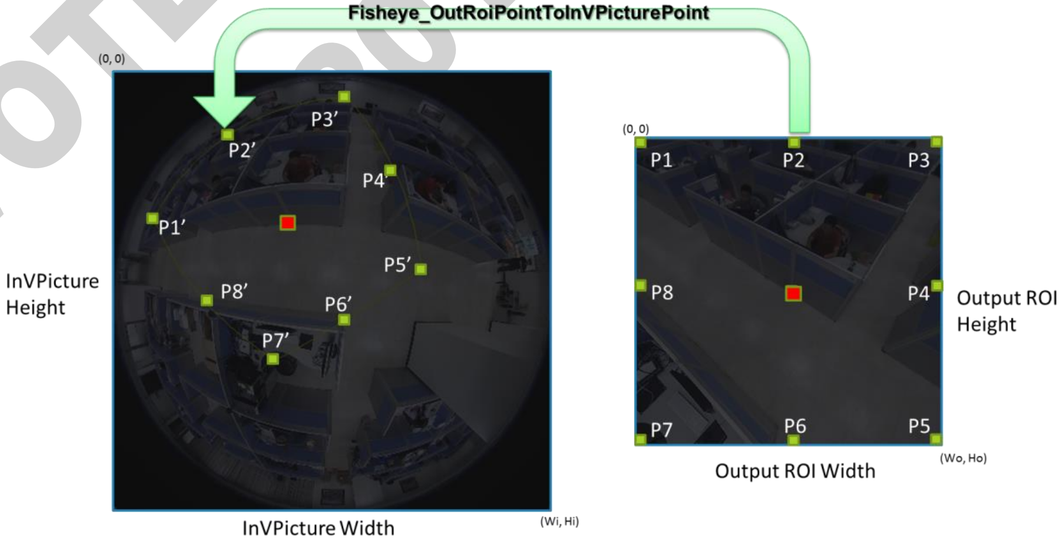
One or more essential option fields are not provided. See Remarks section.

FISHEYE_E_FAIL

Fail to get values. See Remarks section.

Remarks

1. This funciton will get position based on some essential option fields. They are InVPicture, FOVCenter, FOVRadius, OutROI, MountType and DewarpTyp.
2. Not support panorama dewarp types.
3. (Xo, Yo) must locate from (OutROI.Left, OutROI.Top) to (OutROI.Right, OutROI.Bottom)
4. The following figure shows the mapping relationship bewteen OutRoi points and the InVPicture points. The dewaped image border points (P1~P8) and red center point are in the coordinate of OutRoi region. Their mapping points on fisheye source are P1’ to P8’ that are retrieved by this function.



Requirements

fisheyesw.h, fisheyesw_errordef.h

See Also

3.4 dewarp_OcontrolR

4.3.14 Fisheye_PositionToYawPitchRoll (deprecated)

Call this function to get the Yaw, Pitch and Roll which are mapped from fisheye picture. This API is obsolete and will be removed in one of the future releases. Use API Fisheye_InVPicturePointToPanTilt for the replacement.

Syntax

```
void Fisheye_PositionToYawPitchRoll (
                                        FEPOINT             hObject,
                                        long                Xo,
                                        FEMOUNTTYPE         Yo,
                                        int                 pXi,
                                        int                 pYi
                                        float               *pYaw,
                                        float               *pPitch,
                                        float               *pRoll,
);
```

Parameters

FOVCenter

[in] The center of fisheye circular region.

FOVRadius

[in] The radius of fisheye circular region.

MountType

[in] Specifies the mount type of fisheye camera. You can use one of the values defined in FEMOUNTTYPE.

X

[in] The x-coordinate in fisheye image.

Y

[in] The y-coordinate in fisheye image.

***pYaw**

[out] The yaw angle. This parameter is valid only when the mount type is FE_MOUNT_WALL.

***pPitch**

[out] The pitch angle.

***pRoll**

[out] The roll angle. This parameter is valid only when the mount type is FE_MOUNT_CEILING and FE_MOUNT_FLOOR.

Return Values

N/A

Remarks

1. This function is a standalone utility to retrieve the relative Yaw, Pitch and Roll according to the position(X,Y) of fisheye image.
2. You need to provide the values of FOVCenter, FOVRadius, X and Y based on the target width and height. Yaw, pitch and roll are used for ePTZ control in rectilinear projection. The minimum and maximum values of these parameters are controlled by fisheyesw module. You can directly set the values of yaw, pitch and roll which are get from Fisheye_PositionToYawPitchRoll.

Requirements

fisheyesw.h

See Also

Example: (768x768 fisheye image)

```
FEPOINT center = {384, 384};

int radius = 384;
```

```
FEMOUNTTYPE type = FE_MOUNT_CEILING;
int X = 499, Y = 277;

FEOPTION option = {0};
Fisheye_PositionToYawPitchRoll(center, radius, type, X, Y,
&option.Yaw, &option.Pitch, &option.Roll);

option.Flags = (FE_OPTION_ROLL | FE_OPTION_PITCH);
Fisheye_SetOption(hObject, pOption);
Fisheye_OneFrame(hObject);
```