

# Preface

Thank you very much for choosing our company's products. We are dedicated to providing you with the best and highest quality service.

This manual may contain some errors or inaccuracies, and we apologize for any inconvenience this may cause. We appreciate your valuable feedback, and we will regularly update the manual's content.

If you have any questions about the use of this manual and SDK, please contact our technical support team. To ensure a quick resolution of issues, please provide the following information before communication:

1. The current system platform using this SDK, such as Windows, Linux, or others, and specify the platform's detailed version number and bit (32-bit or 64-bit).
2. The model of our hardware product currently in use.
3. The version number of the SDK currently in use.
4. Detailed description of the issue, including, but not limited to, involved functional interfaces, specific error messages, etc.

Thank you for your support!

## Revision History

Version No.	Release Date	Descriptions
V1.0.0	2023-12-05	Initial release
V1.0.1	2024-03-04	<ol style="list-style-type: none"><li>1. The sdk supports PD4 and FC1;</li><li>2. Add device search interface IRC_NET_SearchDevice;</li><li>3. Add the function of recording SDK running logs and setting the level of saving logs;</li><li>4. Add the query device capability interface IRC_NET_GetDevAbility, which currently only supports querying the specific pan tilt capabilities of the device;</li><li>5. Add "preset id" argument for the "delete all</li></ol>

		<p>temperature measurement area</p> <p>interface(IRC_NET_DeleteAllTempRule);</p> <p>6. Add interface for querying, modifying the entire frame temperature measurement alarm configuration;</p> <p>7. Add interfaces for querying the number of shielding area, querying shielding area detail information, adding\deleting\deleting all shielding area;</p> <p>8. Add “channel id” argument for querying ,setting the time title”interface;</p> <p>9. Add “channel id” argument for querying ,setting the channel title”interface;</p> <p>10. Add interfaces related to pan tilt, including: pan tilt control operation, querying auxiliary command status, preset points (query, management control, etc.), cruise group (query, management control, etc.), patrol (query, management control, etc.), precise positioning, 3D positioning;</p> <p>11. Add interfaces for querying SD card files and downloading files</p>
V1.0.2	2024-7-26	<p>1. 1.Added interfaces related to IP configuration.</p> <p>2. 2.Added Temperature Stream V2 interface.</p> <p>3. 3.Added Video Stream V2 interface.</p> <p>4. 4.Added Pan-Tilt interface.</p> <p>5. 5.PC5, FC2, TN220, PD2, SI4.</p>
V1.0.3	2024-8-14	<p>1. 1.Revised Supported Devices Description</p>
V1.0.4	2024-8-17	<p>1. Added new ATR device interfaces: Get detailed temperature measurement rule information_G1, Add temperature measurement rule_G1, Modify a specific temperature measurement rule_G1, Get</p>

		<p>full-frame temperature measurement alarm configuration_G1, Modify full-frame temperature measurement alarm configuration_G1;</p> <p>2. Added new interface to get the number of pattern configurations;</p> <p>3. Added new TN400 environmental parameter correction temperature flow interface;</p> <p>4. Compatible with PC2, PC4, ATR31, ATR61P.</p>
V1.0.5	2024-10-17	<p>1. Compatible with PC6, PC8, FC4, ATR300, ATR600P</p>
V1.0.6	2024-12-06	<p>1. 1.Added PTZ command table</p> <p>2. 2.Added SDK basic parameter table</p>
V1.0.7	2024-12-17	<p>1. Added "Region Focus IRC_NET_PtzRegionFocus" interface;</p> <p>2. Added "Manual Tracking IRC_NET_PtzManualTrack" interface;</p> <p>3. Added "Lens Initialization IRC_NET_PtzLensInit" interface;</p> <p>4. Added "Query Current Preset ID IRC_NET_QueryPtzPresetId" interface;</p> <p>5. Added "Query Random Point Temperature IRC_NET_QueryRandomTemp" interface.</p>
V1.0.8.1	2024-12-25	<p>1. Added "Get Channel Target Recognition Configuration IRC_NET_GetTargetRecognitionConfig" interface;</p> <p>2. Added "Set Channel Target Recognition Configuration IRC_NET_SetTargetRecognitionConfig" interface.</p>
V1.0.9	2025-2-24	<p>1. Added “Reboot</p>

		<p>IRC_NET_SystemReboot”interface;</p> <ol style="list-style-type: none"> <li>2. Added “Data Passthrough”related interfaces;</li> <li>3. Added “Get Gimbal Position</li> </ol> <p>IRC_NET_GetCurrentPtz”interface;</p> <ol style="list-style-type: none"> <li>4. Added “Area Scanning” related interfaces;</li> <li>5. Added “Boot-up\Idle Actions”related interfaces;</li> <li>6. Modify”Alarm Type</li> </ol> <p>IRC_NET_ALARM_TYPE”;</p> <ol style="list-style-type: none"> <li>7. Added Chapter 6 “Demo Source Code &amp; Operation Guide”.</li> </ol>
V1.0.10	2025-3-20	<ol style="list-style-type: none"> <li>1. Compatible with AT20, TN460U;</li> <li>2. Added ranging related interfaces for PC5;</li> <li>3. Added interfaces related to wiper and fill light mode settings;</li> <li>4. Added “IRC_NET SystemRestart” interface;</li> <li>5. Fix IRC_NET_GetCurrentPtz Zoom value acquisition anomaly.</li> </ol>
V1.0.11	2025-7-04	<ol style="list-style-type: none"> <li>1. Compatible with new PC4, PC6;</li> <li>2. Support capability-set-based login for TN220, PC4, and PC6;</li> <li>3. Live view: “Start live view_2” supports pushing target detection bounding box information;</li> <li>4. Device configuration: added “day/night”, “IR brightness”, “IR contrast”, “IR flip”, “IR enhance”, “Web logo”related interfaces;</li> <li>5. PTZ control: added “linkage tracking” , “precise tracking positioning”, “multiplier”, “acceleration” , “get PTZ position_V1” related interfaces;</li> <li>6. Temperature Measurement Configuration:</li> </ol>

		<p>added “temperature unit” related interfaces, “Start pulling real-time temperature stream_V2” added environmental and distance compensation parameters to the returned data, “Get rule/full-frame high-low temperature information” added coordinate to the returned data;</p> <p>7. Alarm Data Retrieval: added alarm image to the returned data, for fire(pulse) type alarm, added fire coordinate to the returned data.</p>
--	--	--

# **Chapter 1 Brief Introduction**

## **Overview**

Welcome to the SDK\_NET Development Manual. This document provides a detailed description of the various functional interfaces, structures, constants, and the calling process of the SDK interface functions included in our company's Ethernet port infrared device development kit. The main features of the SDK include: device connection, live view & image/video capture, temperature measurement configuration & retrieval of temperature data, alarm data retrieval, device configuration, and other functions.

Note: Some interfaces have multiple versions. It is recommended to use the latest version.

# Chapter 2 Development Guide

## 1. Supported Systems for the SDK

### **Windows 32/64bit Version:**

Windows 11/Windows 10/Windows 8/Windows 7

### **Linux 32/64bit Version:**

Ubuntu 18.04 and above

### **Arm Linux 32/64bit Version:**

Linux\_x86\_64, GCC 7.5.0 and above

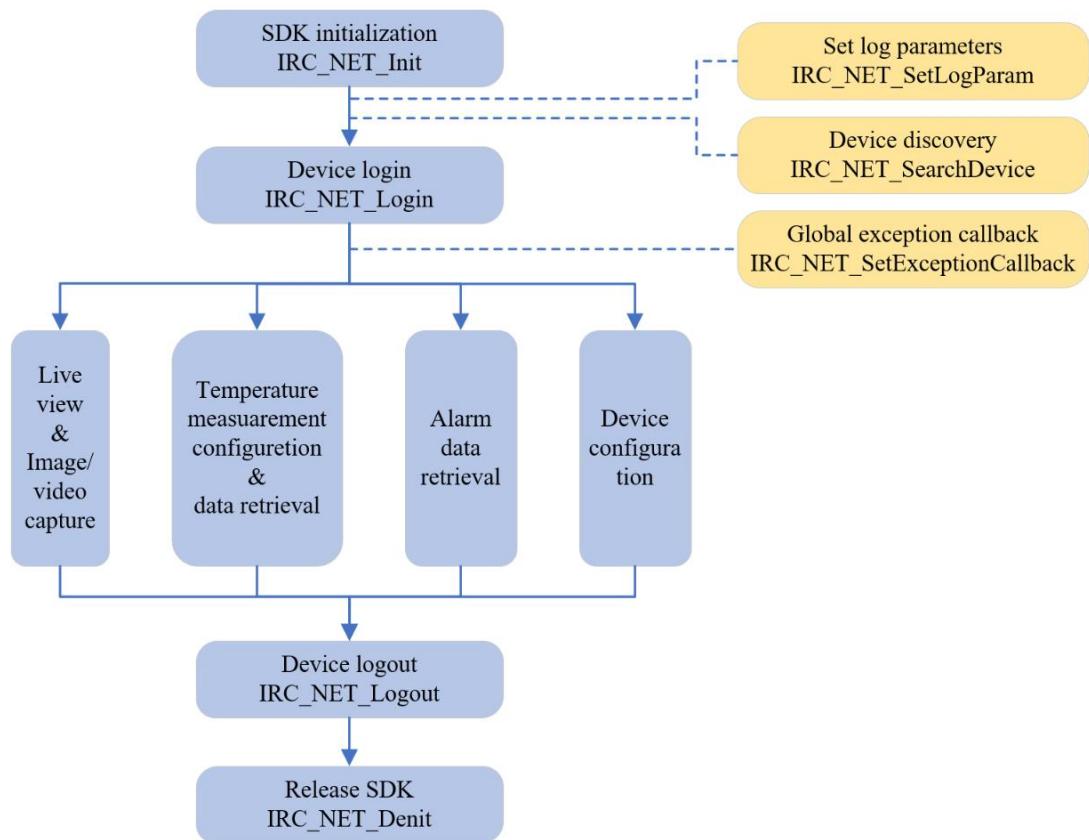
Arrch64\_linux\_gnu\_7.3.1 and above, GCC 7.3.1 and above

## 2. Device Compatibility Information

This SDK is compatible with the following types of devices from our company:

Device Models
AT20
TN Series(TN220 TN430 TN460 TN460U)
ATR Series(ATR31 ATR61P ATR300 ATR600P)
FC Series(FC1 FC2 FC4)
PD Series(PD2 PD4)
PC Series(PC2 PC4 PC5 PC6 PC8)
SI4

### 3. Programming Process



Note: The dashed-line interfaces in the diagram represent non-essential calls that may be bypassed.

# Chapter 3 Interface List

## 1. Device Connection

Interface Name	Interface Description
<a href="#">IRC_NET_Init</a>	SDK initialization
<a href="#">IRC_NET_SetLogParam</a>	Set log parameters
<a href="#">IRC_NET_Deinit</a>	Release SDK
<a href="#">IRC_NET_SearchDevice</a>	Device discovery
<a href="#">IRC_NET_Login</a>	Device login
<a href="#">IRC_NET_Logout</a>	Device logout
<a href="#">IRC_NET_GetDevInfo</a>	Get basic device information
<a href="#">IRC_NET_GetDevAbility</a>	Get device capability information

### 1) SDK Initialization [IRC\\_NET\\_Init](#)

Options	Introduction
Description	SDK initialization
Function	Int IRC_NET_Init()
Parameter	
Return Value	(Refer to the <a href="#">status code</a> table for details.)
Note	

## 2) Set log parameters    **IRC\_NET\_SetLogParam**

Options	Introduction
Description	Set log parameters
Function	<pre>int IRC_NET_SetLogParam(     int level,     const char* logDir,     int upperLimit)</pre>
Parameter	<p>Param [in] level log level, refer to <a href="#"><u>IRC_NET_LOG_LEVEL</u></a></p> <p>Param [in] logDir log file path must be an absolute path and end with "\\", It is recommended that the user create it manually first</p> <p>Param[in] upperLimit upper limit of the number of log files, 0-no upper limit</p>
Return Value	<a href="#"><u>status code</u></a>
Note	After setting the log limit, log files will be cyclically overwritten when the number reaches the upper limit.

## 3) Release SDK    **IRC\_NET\_Deinit**

Options	Introduction
---------	--------------

Description	Release SDK
Function	void IRC_NET_Deinit()
Parameter	
Return Value	
Note	

#### 4) Device discovery    IRC\_NET\_SearchDevice

Options	Introduction
Description	Device discovery
Function	<pre>int IRC_NET_SearchDevice(     int timeout,     <a href="#">IRC_NET_DEV_SEARCH_INFO</a> searchInfos[],     int inSize,     int* outSize);</pre>
Parameter	<p>Param[in] timeout    query timeout, unit: ms</p> <p>Param[out] infos[]    device search information</p> <p>Param[in] inSize    device search information input size</p> <p>Param[out] outSize    Device search information output size</p>
Return Value	<a href="#">status code</a>

Note	
------	--

## 5) Device login IRC\_NET\_Login

Options	Introduction
Description	Device login
Function	<pre>int IRC_NET_Login(     Const <a href="#">IRC_NET_LOGIN_INFO</a>* loginInfo,     IRC_NET_HANDLE* handle);</pre>
Parameter	param[in] loginInfo login information param[out] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 6) Device logout IRC\_NET\_Logout

Options	Introduction
Description	Device logout
Function	<pre>int CALL_METHOD IRC_NET_Logout(     IRC_NET_HANDLE handle);</pre>
Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 7) Get basic device information **IRC\_NET\_GetDeviceInfo**

Options	Introduction
Description	Get basic device information
Function	int IRC_NET_GetDeviceInfo( IRC_NET_HANDLE handle, <a href="#"><u>IRC_NET_DEV_INFO</u></a> * devInfo);
Parameter	param[in] handle operation handle param[out] devInfo device information
Return Value	<a href="#"><u>status code</u></a>
Note	

## 8) Get device capability information

### **IRC\_NET\_GetDevAbility**

Options	Introduction
Description	Get device capability information
Function	int IRC_NET_GetDevAbility( IRC_NET_HANDLE handle, const <a href="#"><u>IRC_NET_DEV_ABILITY_QUERY_CONDITION</u></a> N* condition, void* info,

	int inSize)
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] condition The device capability type to be queried</p> <p>Param [out] info Specific capability information under the device capability type to be queried</p> <p>Param [in] size The specific size of the capability structure under the device capability type to be queried</p>
Return Value	
Note	

## 2. Live View&Image/Video Capture

Interface Name	Interface Description
<a href="#"><u>IRC_NET_StartPreview</u></a>	Start live view
<a href="#"><u>IRC_NET_VIDEO_CALLBACK</u></a>	Video Callback Function
<a href="#"><u>IRC_NET_StartPreview_V2</u></a>	Start live view_V2
<a href="#"><u>IRC_NET_VIDEO_CALLBACK_V2</u></a>	Video Callback Function_V2
<a href="#"><u>IRC_NET_StopPreview</u></a>	Stop live view
<a href="#"><u>IRC_NET_PreviewSnapshot</u></a>	Live view snapshot
<a href="#"><u>IRC_NET_StartPreviewRecord</u></a>	Start live view recording
<a href="#"><u>IRC_NET_StopPreviewRecord</u></a>	Stop live view recording

## 1) Start live view    **IRC\_NET\_StartPreview**

Options	Introduction
Description	Start live view
Function	<pre>int IRC_NET_StartPreview(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_PREVIEW_INFO</a>* previewInfo,     <a href="#">IRC_NET_VIDEO_CALLBACK</a> videoCb,     void* userData);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] previewInfo live view information</p> <p>param[in] videoCb video callback function</p> <p>param[in] userData User-defined data, will be passed through videoCb</p>
Return Value	<a href="#">status code</a>
Note	

## 2) Video callback function

### **IRC\_NET\_VIDEO\_CALLBACK**

Options	Introduction
Description	Video Callback Function
Function	<pre>void* IRC_NET_VIDEO_CALLBACK(     IRC_NET_HANDLE handle,     char* frame,     int width,     int height,     void* userData);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[out] frame video frame data</p> <p>param[out] width video frame width</p> <p>param[out] height video frame height</p> <p>param[out] userData User-defined data, thrown out as is through videoCb</p>
Return Value	
Note	

### 3) Start live view\_V2 IRC\_NET\_StartPreview\_V2

Options	Introduction
Description	Start live view
Function	<pre>int IRC_NET_StartPreview_V2(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_PREVIEW_INFO</a>* previewInfo,     <a href="#">IRC_NET_VIDEO_CALLBACK_V2</a> videoCb,     void* userData);</pre>
Parameter	param[in] handle operation handle param[in] previewInfo preview info param[in] videoCb video callback function param[in] userData User-defined data, will be passed through videoCb
Return Value	<a href="#">status code</a>
Note	

#### 4) Video callback function\_V2

##### **IRC\_NET\_VIDEO\_CALLBACK\_V2**

Options	Introduction
Description	Video callback function_V2
Function	<pre>void* IRC_NET_VIDEO_CALLBACK_V2(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_VIDEO_INFO_CB</a>* videoInfo,     <a href="#">IRC_NET_IVS_INFO_CB</a>* ivsInfo,     void* userData);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[out] videoInfo video callback</p> <p>param[out] ivsInfo video callback intelligent info</p> <p>param[out] userData User-defined data</p>
Return Value	
Note	Added video callback intelligent information

#### 5) Stop live view **IRC\_NET\_StopPreview**

Options	Introduction
Description	Stop live view
Function	Int IRC_NET_StopPreview(

	<code>IRC_NET_HANDLE handle);</code>
Parameter	<code>param[in]</code> handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 6) Live view snapshot **IRC\_NET\_PreviewSnapshot**

Options	Introduction
Description	Live view snapshot
Function	<code>Int IRC_NET_PreviewSnapshot(</code> <code>    IRC_NET_HANDLE handle,</code> <code>    const char* filePath);</code>
Parameter	<code>param[in]</code> handle operation handle <code>param[in]</code> filePath File storage path: [Save file path] + [File name] + .jpg
Return Value	<a href="#">status code</a>
Note	

## 7) Start live view recording

### **IRC\_NET\_StartPreviewRecord**

Options	Introduction
Description	Start live view recording , MP4 format , without temperature data

Function	<code>int IRC_NET_StartPreviewRecord(     IRC_NET_HANDLE handle,     const char* filePath);</code>
Parameter	param[in] handle operation handle param[in] filePath File storage path: [Save file path] + [File name] + .mp4
Return Value	<a href="#">status code</a>
Note	

## 8) Stop live view recording

### **IRC\_NET\_StopPreviewRecord**

Options	Introduction
Description	Stop live view recording
Function	<code>Int IRC_NET_StopPreviewRecord(     IRC_NET_HANDLE handle);</code>
Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 3. Temperature Measurement Configuration & Retrieval of Temperature Data

Interface Name	Interface Description
----------------	-----------------------

<a href="#"><u>IRC_NET_StartPullTemp</u></a>	Start pulling real-time temperature stream
<a href="#"><u>IRC_NET_TEMP_CALLBACK</u></a>	Temperature callback function
<a href="#"><u>IRC_NET_StartPullTemp_V2</u></a>	Start pulling real-time temperature stream_V2
<a href="#"><u>IRC_NET_TEMP_CALLBACK_V2</u></a>	Temperature callback function_V2
<a href="#"><u>IRC_NET_StopPullTemp</u></a>	Stop pulling real-time temperature stream
<a href="#"><u>IRC_NET_QueryTempRuleSize</u></a>	Get the number and summary information of drawn temperature measurement rules
<a href="#"><u>IRC_NET_QueryTempRule</u></a>	Get detailed information for a specific temperature measurement rule
<a href="#"><u>IRC_NET_QueryTempRule_G1</u></a>	Get detailed information for a specific temperature measurement rule_G1
<a href="#"><u>IRC_NET_AddTempRule</u></a>	Add a temperature measurement rule (draw a point, line or box)
<a href="#"><u>IRC_NET_AddTempRule_G1</u></a>	Add a temperature measurement rule_G1 (draw a point, line or

	box)
<a href="#"><u>IRC_NET_UpdateTempRule</u></a>	Modify a specific temperature measurement rule
<a href="#"><u>IRC_NET_UpdateTempRule_G1</u></a>	Modify a specific temperature measurement rule_G1
<a href="#"><u>IRC_NET_DeleteTempRule</u></a>	Delete a single temperature measurement rule
<a href="#"><u>IRC_NET_DeleteAllTempRule</u></a>	Delete all temperature measurement rules
<a href="#"><u>IRC_NET_QueryRuleTempSize</u></a>	Get the number of temperatures that a rule can output
<a href="#"><u>IRC_NET_QueryRuleTemp</u></a>	Get information on the high and low temperatures of a rule
<a href="#"><u>IRC_NET_QueryFrameTemp</u></a>	Get information on the full frame's high and low temperatures
<a href="#"><u>IRC_NET_GetIRGImage</u></a>	Get thermal images in IRG format
<a href="#"><u>IRC_NET_GetDlt664Image</u></a>	Get thermal images in the National Grid 664 format
<a href="#"><u>IRC_NET_QueryTempMaskSize</u></a>	Get the number of temperature measurement shielding areas

<a href="#"><u>IRC_NET_QueryTempMask</u></a>	Get detailed information on the temperature measurement shielding area
<a href="#"><u>IRC_NET_AddTempMask</u></a>	Add temperature measurement shielding area
<a href="#"><u>IRC_NET_DeleteTempMask</u></a>	Delete temperature measurement shielding area
<a href="#"><u>IRC_NET_DeleteAllTempMask</u></a>	Delete all temperature measurement shielding areas
<a href="#"><u>IRC_NET_QueryRandomTemp</u></a>	Get random point temperature
<a href="#"><u>IRC_NET_GetTempUnit</u></a>	Get temperature unit
<a href="#"><u>IRC_NET_SetTempUnit</u></a>	Set temperature unit

## 1) Start pulling real-time temperature stream

### **IRC\_NET\_StartPullTemp**

Options	Introduction
Description	Start pulling real-time temperature stream
Function	<pre>int IRC_NET_StartPullTemp(     IRC_NET_HANDLE handle,     <a href="#"><u>IRC_NET_TEMP_CALLBACK</u></a> tempCb,     void* userData);</pre>
Parameter	param[in] handle operation handle

	param[in] tempCb temperature callback function param[in] userData user self-defined data
Return Value	<a href="#">status code</a>
Note	

## 2) Temperature callback function

### IRC\_NET\_TEMP\_CALLBACK

Options	Introduction
Description	Temperature callback function
Function	<pre>void* IRC_NET_TEMP_CALLBACK(     IRC_NET_HANDLE handle,     char* temp,     int width,     int height,     void* userData);</pre>
Parameter	param[in] handle operation handle param[out] temp temperature frame data param[out] width temperature frame width param[out] height temperature frame height param[out] userData user defined data
Return Value	

Note	The returned temperature frame data format is: Kelvin temperature * 10
------	---

### 3) Start pulling real-time temperature stream\_V2

#### **IRC\_NET\_StartPullTemp\_V2**

Options	Introduction
Description	Start pulling real-time temperature stream
Function	<pre>int IRC_NET_StartPullTemp_V2(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_TEMP_CALLBACK_V2</a> tempCb,     void* userData);</pre>
Parameter	param[in] handle operation handle param[in] tempCb temperature callback function param[in] userData user-defined data
Return Value	<a href="#">Status code</a>
Note	

### 4) Temperature callback function\_V2

#### **IRC\_NET\_TEMP\_CALLBACK\_V2**

Options	Introduction
Description	Temperature callback function_V2
Function	void* <a href="#">IRC_NET_TEMP_CALLBACK_V2</a> (

	<pre>IRC_NET_HANDLE handle, <a href="#">IRC_NET_TEMP_INFO_CB</a>* tempInfo, <a href="#">IRC_NET_TEMP_EXT_INFO_CB</a>* extInfo, void* userData);</pre>
Parameter	param[in] handle operation handle param[out] tempInfo temperature callback temperature info param[out] extInfo temperature callback extended information param[out] userData user-defined data
Return Value	
Note	Added temperature callback extended information The returned temperature data format is: Kelvin temperature * 10

## 5) Stop pulling real-time temperature stream

### **IRC\_NET\_StopPullTemp**

Options	Introduction
Description	Stop pulling real-time temperature stream
Function	int IRC_NET_StopPullTemp( IRC_NET_HANDLE handle);

Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 6) Get the number of interested temperature

### **measurement rules    IRC\_NET\_QueryTempRuleSize**

Options	Introduction
Description	Get the number of drawn temperature measurement rules
Function	<pre>int IRC_NET_QueryTempRuleSize(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>*     tempRuleIndex,     int* size);</pre>
Parameter	<p>param[in] handle operation handle      param[in] tempRuleIndex rule index</p> <p>Note:</p> <p>①In the parameter structure, presetId must be greater than or equal to 0. This is used to Get temperature measurement rule information for a specific preset.If the device is PTZ, set this field to 0 to obtain the</p>

	<p>number of entire frame. If the device is non-PTZ, set this field value to 0.</p> <p>② Set either both type and id to -1, or set them to specific type and id values.</p> <p>param[out] size Query the count</p>
Return Value	<u>status code</u>
Note	

## 7) Get detailed information for an interested temperature measurement rule **IRC\_NET\_QueryTempRule**

Options	Introduction
Description	Get detailed information for an interested temperature measurement rule
Function	<pre>int IRC_NET_QueryTempRule(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>*     tempRuleIndex,     <a href="#">IRC_NET_TEMP_RULE_INFO</a> tempRuleInfos[],     int inSize,     int* outSize);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] tempRuleIndex Temperature measurement</p>

rule index

Note:

① In the parameter structure, presetId must be greater than or equal to 0. This is used to Get temperature measurement rule information for a specific preset. If the device is PTZ, set this field to 0 to obtain the number of entire frame. If the device is non-PTZ, set this field value to 0.

② Set either both type and id to -1, or set them to specific type and id values.

param[out] tempRuleInfos[] temperature measurement rule information

When retrieving rule information for a specific ID, the size of the constructed array should be 1.

Otherwise, the array size should be the return value of the [IRC\\_NET\\_QueryTempRuleSize](#) function.

param[in] inSize Input size of the temperature measurement rule

When retrieving rule information for a specific ID, this field is set to 1. Otherwise, its value should be the return value of the

[IRC\\_NET\\_QueryTempRuleSize](#) function.

	param[out] outSize Output size of the temperature measurement rule
Return Value	<a href="#">status code</a>
Note	When you want to Get all temperature measurement rules or all temperature measurement rules of a specific type, this function needs to be used in conjunction with <a href="#">IRC_NET_QueryTempRuleSize</a>

## 8) Get detailed information for an interested

### temperature measurement rule\_G1

#### **IRC\_NET\_QueryTempRule\_G1**

Options	Introduction
Description	Get detailed information for a specific temperature measurement rule.
Function	<pre>int IRC_NET_QueryTempRule_G1(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>*     tempRuleIndex,     <a href="#">IRC_NET_TEMP_RULE_INFO_G1</a> tempRuleInfos[],     int inSize,     int* outSize);</pre>
Parameter	param[in] handle operation handle

	<p>param[in] tempRuleIndex Temperature measurement rule index, where presetId must be greater than or equal to 0. If other parameters are set to -1, all records will be queried.</p> <p>param[out] tempRuleInfos[] Temperature measurement rule info</p> <p>param[in] inSize temperature measurement rule input size</p> <p>param[out] outSize temperature measurement rule output size</p>
Return Value	<a href="#">status code</a>
Note	Compatible with ATR series only

## 9) Add temperature measurment rule(draw point/line/box)

### IRC\_NET\_AddTempRule

Options	Introduction
Description	Add temperature measurment rule(Draw point/line/box)
Function	int IRC_NET_AddTempRule( IRC_NET_HANDLE handle,

	const <a href="#">IRC_NET_TEMP_RULE_INFO</a> * tempRegionRuleInfo);
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo temperature measurement rule information
Return Value	<a href="#">status code</a>
Note	

## 10) Add temperature measurment rule\_G1

### **IRC\_NET\_AddTempRule\_G1**

Options	Introduction
Description	Add temperature measurement rule_G1
Function	int IRC_NET_AddTempRule_G1(IRC_NET_HANDLE handle, const <a href="#">IRC_NET_TEMP_RULE_INFO_G1</a> * tempRegionRuleInfo);
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo temperature measurement rule information
Return Value	<a href="#">status code</a>

Note	Compatible with ATR series only
------	---------------------------------

## 11)Modify an interested temperature measurement rule

### **IRC\_NET\_UpdateTempRule**

Options	Introduction
Description	Modify a specific temperature measurement rule.
Function	<pre>int IRC_NET_UpdateTempRule(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INFO</a>*     tempRegionRuleInfo);</pre>
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo temperature measurement rule information
Return Value	<a href="#">status code</a>
Note	

## 12)Modify an interested temperature measurement

### **rule\_g1 irc\_net\_updatetemprule\_G1**

Options	Introduction
Description	Modify a specific temperature measurement rule_G1

Function	<pre>int IRC_NET_UpdateTempRule_G1(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INFO_G1</a>*     tempRegionRuleInfo);</pre>
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo Temperature measurement rule info
Return Value	<a href="#">status code</a>
Note	Compatible with ATR series only

### 13) Delete a single temperature measurement rule

#### **IRC\_NET\_DeleteTempRule**

Options	Introduction
Description	Delete a single temperature measurement rule
Function	<pre>int IRC_NET_DeleteTempRule(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>*     tempRuleIndex);</pre>
Parameter	param[in] handle operation handle param[in] tempRuleIndex temperature measurement rule index

Return Value	<a href="#">status code</a>
Note	

## 14) Delete all temperature measurement rules

### IRC\_NET\_DeleteAllTempRule

Options	Introduction
Description	Delete all temperature measurement rules
Function	<pre>int IRC_NET_DeleteAllTempRule(     IRC_NET_HANDLE handle);</pre>
Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 15) Get the number of temperatures that a rule can output

### IRC\_NET\_QueryRuleTempSize

Options	Introduction
Description	Get the number of temperatures that a rule can output
Function	<pre>int IRC_NET_QueryRuleTempSize(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>* tempRuleIndex,</pre>

	int* size);
Parameter	<p>param[in] handle operation handle</p> <p>param[in] tempRuleIndex rule index</p> <p>Note:①In the parameter structure, presetId must be greater than or equal to 0. This is used to Get temperature measurement rule information for a specific preset. If the device is non-PTZ, set this field value to 0.②Please set the values of type and id to -1.</p> <p>param[out] size Query the count</p>
Return Value	<a href="#">status code</a>
Note	

## 16)Get rules high-low temperature information

### IRC\_NET\_QueryRuleTemp

Options	Introduction
Description	Get rules high-low temperature information
Function	<pre>int IRC_NET_QueryRuleTemp(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_RULE_INDEX</a>*     tempRuleIndex,     <a href="#">IRC_NET_RULE_TEMP_INFO</a> ruleTempInfos[],</pre>

	<pre>int inSize, int* outSize);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] tempRuleIndex temperature measurement rule index</p> <p>Note:①In the parameter structure, presetId must be greater than or equal to 0. This is used to Get temperature measurement rule information for a specific preset. If the device is non-PTZ, set this field value to 0.②Set either both type and id to -1, or set them to specific type and id values.</p> <p>param[out] ruleTempInfos[] rule temperature information</p> <p>When retrieving temperature information for a specific ID, the size of the constructed array should be 1. Otherwise, the array size should be the return value of the IRC_NET_QueryRuleTempSize function.</p> <p>param[in] inSize Rule temperature input size</p> <p>When retrieving temperature information for a specific ID, this field is set to 1. Otherwise, its value should be the return value of the</p>

	IRC_NET_QueryRuleTempSize function.  param[out] outSize Rule temperature output size
Return Value	<a href="#">status code</a>
Note	When you want to Get temperature information for all temperature measurement rules or all temperature measurement rules of a specific type, this function needs to be used in conjunction with "IRC_NET_QueryRuleTempSize"

## 17)Get full-frame high-low temperature information

### **IRC\_NET\_QueryFrameTemp**

Options	Introduction
Description	Get full-frame high-low temperature information
Function	int IRC_NET_QueryFrameTemp(  IRC_NET_HANDLE handle,  <a href="#">IRC_NET_TEMP_INFO</a> * tempInfo);
Parameter	param[in] handle operation handle  param[out] ruleTempInfos[] rule temperature information
Return Value	<a href="#">status code</a>
Note	

## 18) Get thermal images in irg format

### **IRC\_NET\_GetIRGImage**

Options	Introduction
Description	Get thermal images in IRG format
Function	<pre>int IRC_NET_GetIRGImage(     IRC_NET_HANDLE handle,     const char* irgFilePath,     const char* jpgFilePath);</pre>
Parameter	<pre>param[in] handle    operation handle param[in] irgFilePath    file saving path+file name+.irg param[in] jpgFilePath file saving path+file name+.jpg</pre>
Return Value	<a href="#">status code</a>
Note	

## 19) Get thermal images in DLT664 format

### **IRC\_NET\_GetDlt664Image**

Options	Introduction
Description	Get thermal images in DLT664 format
Function	<pre>int IRC_NET_GetDLT664Image(</pre>

	<code>IRC_NET_HANDLE handle, const char* filePath);</code>
Parameter	<code>param[in] handle operatio handle param[in] filePath file saving path+file name+.jpg</code>
Return Value	<a href="#">status code</a>
Note	

## 20) Get the number of temperature measurement shielding areas

**areas** `IRC_NET_QueryTempMaskSize`

Options	Introduction
Description	Get the number of temperature measurement shielding areas
Function	<pre>int IRC_NET_QueryTempMaskSize(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_MASK_INDEX</a>*     tempMaskIndex,     int* size);</pre>
Parameter	Param [in] handle operation handle Param [in] tempMaskIndex masks the area index. The presetId in the structure must be greater than or equal to 0. If 0, it means there are no preset points. When id is -1, query all

	Param [out] size query quantity
Return Value	<a href="#">status code</a>
Note	

## 21)Get detailed information on the temperature

### measurement shielding area

#### **IRC\_NET\_QueryTempMask**

Options	Introduction
Description	Get detailed information on the temperature measurement shielding area
Function	<pre>int IRC_NET_QueryTempMask(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_MASK_INDEX</a>*     tempMaskIndex,     <a href="#">IRC_NET_TEMP_MASK_INFO</a> tempMaskInfos[],     int inSize,     int* outSize);</pre>
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] tempMaskIndex Mask area index,</p> <p>presetId must be greater than or equal to 0</p> <p>Param tempMaskInfos temperature measurement shielding area information</p>

	<p>Param [in] inSize Input size of temperature measurement shielding area</p> <p>Param [out] outSize Output size of temperature measurement shielding area</p>
Return Value	<a href="#">status code</a>
Note	

## 22) Add temperature measurement shielding area

### **IRC\_NET\_AddTempMask**

Options	Introduction
Description	Add temperature measurement shielding area
Function	<pre>int IRC_NET_AddTempMask(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_MASK_INFO</a>*     tempMaskInfo);</pre>
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] tempMaskInfo Temperature measurement shielding area information</p>
Return Value	<a href="#">status code</a>
Note	

## 23)Delete temperature measurement shielding area

### **IRC\_NET\_DeleteTempMask**

Options	Introduction
Description	Delete temperature measurement shielding area.
Function	<pre>int IRC_NET_DeleteTempMask(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_MASK_INDEX</a>*     tempMaskIndex);</pre>
Parameter	Param [in] handle operation handle Param [in] tempMaskIndex Mask Area Index
Return Value	<a href="#">status code</a>
Note	

## 24)Delete all temperature measurement shielding areas

### **IRC\_NET\_DeleteAllTempMask**

Options	Introduction
Description	Delete all temperature measurement shielding areas.
Function	<pre>int IRC_NET_DeleteAllTempMask(     IRC_NET_HANDLE handle,     int presetId);</pre>
Parameter	Param [in] handle operation handle

	Param [in] presetId
Return Value	<a href="#">status code</a>
Note	

## 25) Get random point temperature

### IRC\_NET\_QueryRandomTemp

Options	Introduction
Description	Get random point temperature
Function	<pre>int IRC_NET_QueryRandomTemp(     IRC_NET_HANDLE handle,     const IRC_NET_POINT* point, float* temp);</pre>
Parameter	param[in] handle operation handle Point[in] Query point, using the 8192 coordinate system temp[out] temperature value
Return Value	<a href="#">Status code</a>
Note	

## 26) Get temperature unit IRC\_NET\_GetTempUnit

Options	Introduction
Description	Get temperature unit
Function	<pre>int IRC_NET_GetTempUnit(</pre>

	<code>IRC_NET_HANDLE handle, int* unit);</code>
Parameter	<code>param[in] handle operation handle param[out] unit temp unit, 0: Celsius, 1: Fahrenheit, 2: Kelvin</code>
Return Value	<a href="#">Status code</a>
Note	

## 27) Set temperature unit `IRC_NET_SetTempUnit`

Options	Introduction
Description	Set temperature unit
Function	<code>int IRC_NET_SetTempUnit( IRC_NET_HANDLE handle, int unit);</code>
Parameter	<code>param[in] handle operation handle param[in] unit temp unit, 0: Celsius, 1: Fahrenheit, 2: Kelvin</code>
Return Value	<a href="#">Status code</a>
Note	

## 4. Alarm Data Retrieval

Interface Name	Interface Description
----------------	-----------------------

<a href="#"><u>IRC_NET_SetExceptionCallback</u></a>	Global exception callback
<a href="#"><u>IRC_NET_EXCEPTION_CALL_BACK</u></a>	exception callback function
<a href="#"><u>IRC_NET_SubscribeAlarm</u></a>	Subscribe to alarms
<a href="#"><u>IRC_NET_ALARM_CALLBACK</u></a>	Alarm callback function
<a href="#"><u>IRC_NET_UnsubscribeAlarm</u></a>	Unsubscribe from alarms
<a href="#"><u>IRC_NET_QueryFrameTempAlarmConfig</u></a>	Get the entire frame temperature measurement alarm configuration
<a href="#"><u>IRC_NET_QueryFrameTempAlarmConfig_G1</u></a>	Get the entire frame temperature measurement alarm configuration_G1
<a href="#"><u>IRC_NET_UpdateFrameTempAlarmConfig</u></a>	Modify the entire frame temperature measurement alarm configuration
<a href="#"><u>IRC_NET_UpdateFrameTempAlarmConfig_G1</u></a>	Modify the entire frame temperature measurement alarm configuration_G1

## 1) Global exception callback

### **IRC\_NET\_SetExceptionCallback**

Options	Introduction
Description	Exception callback registration
Function	<pre>int IRC_NET_SetExceptionCallback(     IRC_NET_HANDLE handle,     <a href="#"><u>IRC_NET_EXCEPTION_CALLBACK</u></a>     exceptionCb,     void* userData);</pre>
Parameter	<p>param[in] handle operatio handle</p> <p>param[in] exceptionCb exception callback function</p> <p>param[in] userData user self-defined data</p>
Return Value	<a href="#"><u>status code</u></a>
Note	

## 2) exception callback function

### **IRC\_NET\_EXCEPTION\_CALLBACK**

Options	Introduction
Description	exception callback function, push exception information after registration
Function	<pre>void IRC_NET_EXCEPTION_CALLBACK(</pre>

	<pre>IRC_NET_HANDLE handle, int exceptionType, void* userData);</pre>
Parameter	param[in] handle operation handle param[out] exceptionType exception type, refer to <a href="#"><u>IRC_NET_EXCEPTION_TYPE</u></a> param[out] userData self-defined data
Return Value	
Note	With the "Global Exception Callback" function. For handling methods of exception types, please refer to the demo.

### 3) Subscribe to Alarms    IRC\_NET\_SubscribeAlarm

Options	Introduction
Description	Subscribe to alarms and register the alarm callback.
Function	<pre>int IRC_NET_SubscribeAlarm( IRC_NET_HANDLE handle, <a href="#"><u>IRC_NET_ALARM_CALLBACK</u></a> alarmCb, void* userData);</pre>
Parameter	param[in] handle operation handle param[in] alarmCb alarm callback function

	param[in] userData user self-defined data
Return Value	<a href="#">status code</a>
Note	

#### 4) Alarm callback function

##### **IRC\_NET\_ALARM\_CALLBACK**

Options	Introduction
Description	Alarm callback function,push alarm information after registration
Function	<pre>void IRC_NET_ALARM_CALLBACK(     IRC_NET_HANDLE handle,     int alarmType,     void* alarmInfo,     void* userData);</pre>
Parameter	param[in] handle operation handle param[out] alarmType alarm type, refer to <a href="#">IRC_NET_ALARM_TYPE</a> param[out] alarmInfo alarm information param[out] userData self-defined data
Return Value	
Note	With the "Subscribe to alarms" function. For handling methods of alarm types, please refer to the

	demo.
--	-------

## 5) Unsubscribe from Alarms

### **IRC\_NET\_UnsubscribeAlarm**

Options	Introduction
Description	Unsubscribe from alarms
Function	int IRC_NET_UnsubscribeAlarm( IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 6) Get the entire frame temperature measurement alarm

### **configuration**

### **IRC\_NET\_QueryFrameTempAlarmConfig**

Options	Introduction
Description	Get the entire frame temperature measurement alarm configuration
Function	int IRC_NET_QueryFrameTempAlarmConfig( IRC_NET_HANDLE handle, <a href="#">IRC_NET_TEMP_FRAME_ALARM_CONFIG</a> * alarmConfig);

Parameter	Param [in] handle operation handle Param [out] alarm configuration for whole frame temperature measurement alarm
Return Value	<a href="#">status code</a>
Note	

## 7) Get the entire frame temperature measurement alarm configuration\_G1

### IRC\_NET\_QueryFrameTempAlarmConfig\_G1

Options	Introduction
Description	Get the entire frame temperature measurement alarm configuration_G1
Function	int  IRC_NET_QueryFrameTempAlarmConfig_G1(IRC_ NET_HANDLE handle,  <a href="#">IRC_NET_FRAME_TEMP_ALARM_CONFIG_G1</a>  * alarmConfig);
Parameter	param[in] handle operation handle param[out] alarmConfig full-frame temperature measurement alarm configuration
Return Value	<a href="#">status code</a>
Note	

## 8) Modify the entire frame temperature measurement

alarm **IRC\_NET\_UpdateFrameTempAlarmConfig**

Options	Introduction
Description	Modify the entire frame temperature measurement alarm
Function	<pre>Int IRC_NET_UpdateFrameTempAlarmConfig(     IRC_NET_HANDLE handle,     const     <a href="#"><u>IRC_NET_TEMP_FRAME_ALARM_CONFIG</u></a>*     alarmConfig);</pre>
Parameter	Param [in] handle operation handle Param [out] alarm configuration for whole frame temperature measurement alarm
Return Value	<a href="#"><u>status code</u></a>
Note	

## 9) Modify full-frame temperature measurement alarm

**configuration\_G1**

**IRC\_NET\_UpdateFrameTempAlarmConfig\_G1**

Options	Introduction
---------	--------------

Description	Modify full-frame temperature measurement alarm configuration
Function	<pre>Int IRC_NET_UpdateFrameTempAlarmConfig(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_FRAME_ALARM_CONFIG</a>*     alarmConfig);</pre>
Parameter	param[in] handle operation handle param[out] alarmConfig full-frame temperature measurement alarm configuration
Return Value	<a href="#">status code</a>
Note	

## 5.Device Configuration

Interface Name	Interface Description
System configuration	
<a href="#">IRC_NET_SyncSystemTime</a>	Synchronize system time
<a href="#">IRC_NET_CorrectShutter</a>	Shutter/background correction
<a href="#">IRC_NET_GetTempBarState</a>	Get temperature bar status
<a href="#">IRC_NET_SetTempBarState</a>	Set temperature bar status
<a href="#">IRC_NET_GetPalletteType</a>	Get color pallette index
<a href="#">IRC_NET_SetPalletteType</a>	Set color palette index
<a href="#">IRC_NET_GetTempLevel</a>	Get temperature measurement

	range
<a href="#"><u>IRC_NET_SetTempLevel</u></a>	Set temperature measurement range
<a href="#"><u>IRC_NET_GetOSDState</u></a>	Get OSD (On-Screen Display) overall status
<a href="#"><u>IRC_NET_SetOSDState</u></a>	Set OSD overall status
<a href="#"><u>IRC_NET_GetOSDTimeTitleInfo</u></a>	Get time title
<a href="#"><u>IRC_NET_SetOSDTimeTitleInfo</u></a>	Set time title
<a href="#"><u>IRC_NET_GetOSDChannelTitleInfo</u></a>	Get channel title
<a href="#"><u>IRC_NET_SetOSDChannelTitleInfo</u></a>	Set channel title
<a href="#"><u>IRC_NET_GetLaserDistanceOsdParam</u></a>	Get Laser Distance OSD parameters
<a href="#"><u>IRC_NET_SetLaserDistanceOsdParam</u></a>	Set Laser Distance OSD parameters
<a href="#"><u>IRC_NET_GetEnvParam</u></a>	Get environmental parameters
<a href="#"><u>IRC_NET_SetEnvParam</u></a>	Set environmental parameters
<a href="#"><u>IRC_NET_GetFrameRate</u></a>	Get temperature measurement frame rate
<a href="#"><u>IRC_NET_SetFrameRate</u></a>	Set temperature measurement frame rate

<a href="#"><u>IRC_NET_GetTempSpanInfo</u></a>	Get temperature span information
<a href="#"><u>IRC_NET_SetTempSpanInfo</u></a>	Set temperature span information
<a href="#"><u>IRC_NET_GetIpConfig</u></a>	Get IP Configuration
<a href="#"><u>IRC_NET_SetIpConfig</u></a>	Set IP Configuration
<a href="#"><u>IRC_NET_QueryFileSize</u></a>	Get the number of SD card files
<a href="#"><u>IRC_NET_QueryFile</u></a>	Get SD card files
<a href="#"><u>IRC_NET_StartDownloadFile</u></a>	Download SD card files
<a href="#"><u>IRC_NET_GetDownloadProgress</u></a>	Check download progress
<a href="#"><u>IRC_NET_StopDownloadFile</u></a>	Stop downloading
<a href="#"><u>IRC_NET_SetTargetRecognitionConfig</u></a>	Set Channel Target Recognition Configuration
<a href="#"><u>IRC_NET_GetTargetRecognitionConfig</u></a>	Get Channel Target Recognition Configuration
<a href="#"><u>IRC_NET_SystemReboot</u></a>	Reboot
<a href="#"><u>IRC_NET_SetTransparentState</u></a>	Set Passthrough State
<a href="#"><u>IRC_NET_TransparentData</u></a>	Data Passthrough
<a href="#"><u>IRC_NET_GetWiperConfigInfo</u></a>	Get wiper configuration
<a href="#"><u>IRC_NET_SetWiperConfigInfo</u></a>	Set wiper configuration
<a href="#"><u>IRC_NET_GetFillLightConfigInfo</u></a>	Get fill light configuration
<a href="#"><u>IRC_NET_SetFillLightConfigInfo</u></a>	Set fill light configuration

<a href="#"><u>IRC_NET_SystemRestart</u></a>	Power Restart
PTZ Control	
<a href="#"><u>IRC_NET_PtzControl</u></a>	Pan tilt control operation
<a href="#"><u>IRC_NET_GetPtzAuxFuncState</u></a>	Get the status of PTZ auxiliary command
<a href="#"><u>IRC_NET_GetLaserDistance</u></a>	Execute laser ranging
<a href="#"><u>IRC_NET_QueryPtzPresetSize</u></a>	Get the number of preset points
<a href="#"><u>IRC_NET_QueryPtzPreset</u></a>	Get preset point information
<a href="#"><u>IRC_NET_PtzPresetControl</u></a>	Preset point control
<a href="#"><u>IRC_NET_QueryPtzPresetId</u></a>	Get Current Preset ID
<a href="#"><u>IRC_NET_QueryPtzTourSize</u></a>	Get the number of cruise group configurations
<a href="#"><u>IRC_NET_QueryPtzTour</u></a>	Get cruise group configuration
<a href="#"><u>IRC_NET_PtzTourControl</u></a>	Cruise control
<a href="#"><u>IRC_NET_QueryPtzPatternSize</u></a>	Get Pattern Configuration Quantity
<a href="#"><u>IRC_NET_QueryPtzPattern</u></a>	Get patrol configuration
<a href="#"><u>IRC_NET_PtzPatternControl</u></a>	Patrol control
<a href="#"><u>IRC_NET_ResetPtzConfig</u></a>	Cloud tilt restoration default configuration

<a href="#"><u>IRC_NET_PtzPrecisePosition</u></a>	Pan tilt precise positioning
<a href="#"><u>IRC_NET_Ptz3DPosition</u></a>	3D positioning
<a href="#"><u>IRC_NET_SwivelControl</u></a>	Pan/Tilt Control
<a href="#"><u>IRC_NET_PtzRegionFocus</u></a>	Region Focus
<a href="#"><u>IRC_NET_PtzManualTrack</u></a>	Manual Tracking
<a href="#"><u>IRC_NET_PtzLensInit</u></a>	Lens Initialization
<a href="#"><u>IRC_NET_GetCurrentPtz</u></a>	Get Gimbal Position
<a href="#"><u>IRC_NET_QueryPtzRegionScanInfo</u></a>	
<a href="#"><u>nfo</u></a>	Get Area Scanning Information
<a href="#"><u>IRC_NET_SetPtzRegionScanInfo</u></a>	Set Area Scanning Information
<a href="#"><u>IRC_NET_DeletePtzRegionScanInfo</u></a>	Delete Area Scanning
<a href="#"><u>nfo</u></a>	Information
<a href="#"><u>IRC_NET_PtzRegionScanControl</u></a>	Area Scanning Control
<a href="#"><u>IRC_NET_QueryPtzBootActionInfo</u></a>	
<a href="#"><u>fo</u></a>	Get Boot-up Actions
<a href="#"><u>IRC_NET_SetPtzBootActionInfo</u></a>	Set Boot-up Actions
<a href="#"><u>IRC_NET_QueryPtzParkActionInfo</u></a>	
<a href="#"><u>fo</u></a>	Get Idle Actions
<a href="#"><u>IRC_NET_SetPtzParkActionInfo</u></a>	Set Idle Actions

## 5.1 System Configuration

### 1) Synchronize system time IRC\_NET\_SyncSystemTime

Options	Introduction
Description	Synchronize system time
Function	<pre>int IRC_NET_SyncSystemTime(     IRC_NET_HANDLE handle,     const char* datetime);</pre>
Parameter	<p>param[in] handle operation handle param[in] datetime Synchronize time, the format is “2020-05-21 12:22:33”</p>
Return Value	<a href="#">status code</a>
Note	

### 2) Shutter correction IRC\_NET\_CorrectShutter

Options	Introduction
Description	Implementing shutter calibration function
Function	<pre>int IRC_NET_CorrectShutter(     IRC_NET_HANDLE handle);</pre>
Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

### 3) Get temperature color palette bar display status

#### **IRC\_NET\_GetTempBarState**

Options	Introduction
Description	Get the display status of the temperature color palette bar on the real-time preview page sidebar.
Function	<pre>int IRC_NET_GetTempBarState(     IRC_NET_HANDLE handle,     int* state);</pre>
Parameter	param[in] handle operation handle param[out] state temperature bar status, 0: off 1:on
Return Value	<a href="#">status code</a>
Note	

### 4) Set the temperature color palette bar display status

#### **IRC\_NET\_SetTempBarState**

Options	Introduction
Description	Set the display status of the temperature color palette bar on the real-time preview page sidebar.
Function	<pre>int IRC_NET_SetTempBarState(     IRC_NET_HANDLE handle,     int state);</pre>

Parameter	param[in] handle operation handle param[in] state temperature bar status, 0: off 1:on
Return Value	<a href="#">status code</a>
Note	

## 5) Get current color palette index

### **IRC\_NET\_GetPalleteType**

Options	Introduction
Description	Get the current pseudocolor index for the live preview image
Function	int IRC_NET_GetPalleteType( IRC_NET_HANDLE handle, int* palleteType);
Parameter	param[in] handle operation handle param[out] palleteType Color Palette type, refer to <a href="#">IRC_NET_PALETTE_TYPE</a>
Return Value	<a href="#">status code</a>
Note	

## 6) Set color palette index **IRC\_NET\_SetPalleteType**

Options	Introduction
Description	Set color palette index

Function	<pre>int IRC_NET_SetPalletType(     IRC_NET_HANDLE handle,     int palletType);</pre>
Parameter	param[in] handle operation handle param[in] palletType color palette type, refer to <a href="#"><u>IRC_NET_PALETTE_TYPE</u></a>
Return Value	<a href="#"><u>status code</u></a>
Note	

## 7) Get temperature measurement range

### **IRC\_NET\_GetTempLevel**

Options	Introduction
Description	Get temperature measurement range
Function	<pre>int IRC_NET_GetTempLevel(     IRC_NET_HANDLE handle,     int* level);</pre>
Parameter	param[in] handle operation handle param[out] level temperature measurement range, refer to <a href="#"><u>IRC_NET_TEMP_LEVEL_TYPE</u></a>
Return Value	<a href="#"><u>status code</u></a>
Note	

## 8) Set temperature measurement range

### **IRC\_NET\_SetTempLevel**

Options	Introduction
Description	Set temperature measurement range
Function	<pre>int IRC_NET_SetTempLevel(     IRC_NET_HANDLE handle,     int level);</pre>
Parameter	param[in] handle operation handle param[in] level temperature measurement range, refer to <a href="#">IRC_NET_TEMP_LEVEL_TYPE</a>
Return Value	<a href="#">status code</a>
Note	

## 9) Get OSD (on-screen display) overall status

### **IRC\_NET\_GetOSDState**

Options	Introduction
Description	Get OSD overall status
Function	<pre>int IRC_NET_GetOSDState(     IRC_NET_HANDLE handle,     int* osdMode);</pre>
Parameter	param[in] handle operation handle

	param[out] osdMode temperature measurement information OSD mode; 0 on, 2 off
Return Value	<a href="#">status code</a>
Note	

## 10) Set OSD (on-screen display) overall status

### **IRC\_NET\_SetOSDState**

Options	Introduction
Description	Set OSD overall status
Function	int IRC_NET_SetOSDState( IRC_NET_HANDLE handle, int osdMode);
Parameter	param[in] handle operation handle param[in] osdMode temperature measurement information OSD mode; 0 on, 2 off
Return Value	<a href="#">status code</a>
Note	

## 11) Get time title **IRC\_NET\_GetOSDTimeTitleInfo**

Options	Introduction
---------	--------------

Description	Get time title
Function	<pre>int IRC_NET_GetOSDTimeTitleInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET OSD TIME TITLE INFO</a>*     timeTitleInfo);</pre>
Parameter	param[in] handle operation handle param[out] timeTitleInfo time title information
Return Value	<a href="#">status code</a>
Note	

## 12) Set time title [IRC\\_NET\\_SetOSDTimeTitleInfo](#)

Options	Introduction
Description	Set time title
Function	<pre>int IRC_NET_SetOSDTimeTitleInfo(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET OSD TIME TITLE INFO</a>*     timeTitleInfo);</pre>
Parameter	param[in] handle operation handle param[in] timeTitleInfo time title information
Return Value	<a href="#">status code</a>
Note	

## 13) Get channel title

### **IRC\_NET\_GetOSDChannelTitleInfo**

Options	Introduction
Description	Get channel title
Function	<pre>int IRC_NET_GetOSDChannelTitleInfo(     IRC_NET_HANDLE handle,     <a href="#"><u>IRC_NET OSD CHANNEL TITLE INFO</u></a>*     channelTitleInfo);</pre>
Parameter	param[in] handle operation handle param[out] channelTitleInfo channel title information
Return Value	<a href="#"><u>status code</u></a>
Note	

## 14) Set channel title **IRC\_NET\_SetOSDChannelTitleInfo**

Options	Introduction
Description	Set channel title
Function	<pre>int IRC_NET_SetOSDChannelTitleInfo(     IRC_NET_HANDLE handle,     const <a href="#"><u>IRC_NET OSD CHANNEL TITLE INFO</u></a>*     channelTitleInfo);</pre>
Parameter	param[in] handle operation handle

	param[in] channelTitleInfo channel title information
Return Value	<a href="#">status code</a>
Note	

## 15) Get laser distance OSD parameters

### **IRC\_NET\_GetLaserDistanceOsdParam**

Options	Introduction
Description	Get Laser Distance OSD parameters
Function	<pre>int IRC_NET_GetLaserDistanceOsdParam(     IRC_NET_HANDLE handle,     int channel,     <a href="#">IRC_NET_LASER_DISTANCE OSD PARAM</a>*     laserDistanceOsdParam);</pre>
Parameter	param[in] handle operation handle param[in] channel channel param[out] laserDistanceOsdParam Laser Distance OSD parameters
Return Value	<a href="#">status code</a>
Note	

## 16) Set laser distance OSD parameters

### **IRC\_NET\_SetLaserDistanceOsdParam**

Options	Introduction
Description	Set Laser Distance OSD parameters
Function	<pre>int IRC_NET_SetLaserDistanceOsdParam(     IRC_NET_HANDLE handle,     int channel,     const     <a href="#">IRC_NET_LASER_DISTANCE OSD PARAM*</a>     laserDistanceOsdParam);</pre>
Parameter	param[in] handle operation handle param[in] channel channel param[in] laserDistanceOsdParam Laser Distance OSD parameters
Return Value	<a href="#">status code</a>
Note	

## 17) Get environmental parameters of the full frame

### **IRC\_NET\_GetEnvParam**

Options	Introduction
Description	Get environmental parameters of the full frame

Function	<pre>int IRC_NET_GetEnvParam(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_ENV_PARAM</a>* envParam);</pre>
Parameter	param[in] handle operation handle param[out] envParam environmental parameters
Return Value	<a href="#">status code</a>
Note	

## 18) Set environmental parameters of the full frame

### **IRC\_NET\_SetEnvParam**

Options	Introduction
Description	Set environmental parameters of the full frame
Function	<pre>int IRC_NET_SetEnvParam(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_ENV_PARAM</a>* envParam);</pre>
Parameter	param[in] handle operation handle param[in] envParam environmental parameters
Return Value	<a href="#">status code</a>
Note	

## 19) Get temperature measurement frame rate

### **IRC\_NET\_GetFrameRate**

Options	Introduction
Description	Get temperature measurement frame rate
Function	<pre>int IRC_NET_GetFrameRate(     IRC_NET_HANDLE handle,     int* rate);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[out] rate temperature measurement frame rate, range[1, frame rate], 12 by default, support 25 frames at most</p>
Return Value	<a href="#">status code</a>
Note	

## 20) Set temperature measurement frame rate

### **IRC\_NET\_SetFrameRate**

Options	Introduction
Description	Set temperature measurement frame rate
Function	<pre>int IRC_NET_SetFrameRate(     IRC_NET_HANDLE handle,     int rate);</pre>

Parameter	<p>param[in] handle operation handle</p> <p>param[in] rate temperature measurement frame rate, range[1, frame rate], 12 by default, support 25 frames at most</p>
Return Value	<a href="#">status code</a>
Note	

## 21) Get temperature span information

### IRC\_NET\_GetTempSpanInfo

Options	Introduction
Description	Get temperature span information
Function	<pre>int IRC_NET_GetTempSpanInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_TEMP_SPAN_INFO</a>* tempSpanInfo);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[out] tempSpanInfo temperature span information</p>
Return Value	<a href="#">status code</a>
Note	

## 22) Set temperature span information

### IRC\_NET\_SetTempSpanInfo

Options	Introduction
Description	Set temperature span information
Function	<pre>int IRC_NET_SetTempSpanInfo(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_TEMP_SPAN_INFO</a>*     tempSpanInfo);</pre>
Parameter	param[in] handle operation handle param[in] tempSpanInfo temperature span information
Return Value	<a href="#">status code</a>
Note	

## 23) Get IP configuration IRC\_NET\_GetIpConfig

Options	Introduction
Description	Get IP configuration
Function	<pre>IRC_NET_GetIpConfig(IRC_NET_HANDLE handle, <a href="#">IRC_NET_IP_CONFIG</a>* ipConfig);</pre>
Parameter	param[in] handle operation handle param[out] ipConfig ip configuration

Return Value	<a href="#">status code</a>
Note	

## 24) Set IP configurations    **IRC\_NET\_SetIpConfig**

Options	Introduction
Description	Set IP configurations
Function	IRC_NET_SetIpConfig(IRC_NET_HANDLE handle, const <a href="#">IRC_NET_IP_CONFIG</a> * ipConfig)
Parameter	param[in] handle operation handle param[in] ipConfig ip configurations
Return Value	<a href="#">status code</a>
Note	

## 25) Get the number of files on the SD card

### **IRC\_NET\_QueryFileSize**

Options	Introduction
Description	Get the number of files on the SD card
Function	int IRC_NET_QueryFileSize( IRC_NET_HANDLE handle, const <a href="#">IRC_NET_FILE_QUERY_PARAM</a> * queryParam, int* size);
Parameter	param[in] handle operation handle

	param[in] condition Query conditions param[out] size Query quantity
Return Value	<a href="#">status code</a>
Note	

## 26)Get SD card files    **IRC\_NET\_QueryFile**

Options	Introduction
Description	Get SD card files
Function	<pre>int IRC_NET_QueryFile(     IRC_NET_HANDLE handle,     const IRC_NET_FILE_QUERY_PARAM*     queryParam, <a href="#">IRC_NET_FILE_INFO</a> fileInfo[],     int inSize,     int* outSize);</pre>
Parameter	param[in] handle operation handle param[in] condition query conditions param[out] fileInfo[] file info param[in] inSize File information input size param[out] outSize File information output size
Return Value	<a href="#">status code</a>
Note	

## 27) Download SD card files

### **IRC\_NET\_StartDownloadFile**

Options	Introduction
Description	Download files on SD card
Function	<pre>int IRC_NET_StartDownloadFile(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_FILE_DOWNLOAD_INFO</a>*     downloadInfo);</pre>
Parameter	param[in] handle operation handle param[in] downloadInfo download info
Return Value	<a href="#">status code</a>
Note	

## 28) Get download progress

### **IRC\_NET\_GetDownloadProgress**

Options	Introduction
Description	Get download progress
Function	<pre>int IRC_NET_GetDownloadProgress(     IRC_NET_HANDLE handle,     int fileHandle,     <a href="#">IRC_NET_FILE_DOWNLOAD_PROGRESS</a>/*</pre>

	downloadProgress);
Parameter	param[in] handle operation handle param[in] fileHandle download handle, IRC_NET_DownloadSDFile return value param[out] downloadProgress download process
Return Value	-1 indicates failure, 0-100 represents download progress, 100 means download completion. The normal range is 0-100; if 200 is returned, it indicates a network exception.
Note	

## 29) Stop download IRC\_NET\_StopDownloadFile

Options	Introduction
Description	Stop download
Function	int IRC_NET_StopDownloadFile( IRC_NET_HANDLE handle, int fileHandle);
Parameter	param[in] handle operation handle param[in] fileHandle download handle, IRC_NET_StartDownloadFile return value
Return Value	<a href="#">status code</a>
Note	

### 30) Set channel target recognition configuration

#### **IRC\_NET\_SetTargetRecognitionConfig**

Options	Introduction
Description	Set channel target recognition configuration
Function	IRC_NET_SetTargetRecognitionConfig(IRC_NET_ HANDLE handle, int channel, const <a href="#"><u>IRC_NET_IP_TARGET_RECOGNITION_CONFIG</u></a> * targetRecognitionConfig)
Parameter	param[in] handle operation handle param[in] channel channle No. param[in] targetRecognitionConfig channel target recognition configuration
Return Value	<a href="#"><u>status code</u></a>
Note	

### 31) Get channel target recognition configuration

#### **IRC\_NET\_GetTargetRecognitionConfig**

Options	Introduction
Description	Get channel target recognition configuration
Function	IRC_NET_GetTargetRecognitionConfig(IRC_NET_ HANDLE handle, int channel,

	<u><a href="#">IRC_NET_IP_TARGET_RECOGNITION_CONFIG</a></u> * targetRecognitionConfig)
Parameter	param[in] handle operation handle param[in] channel channel No. param[out] targetRecognitionConfig channel target recognition configuration
Return Value	<u><a href="#">status code</a></u>
Note	

### 32) Reboot IRC\_NET\_SystemReboot

Options	Introduction
Description	Reboot
Function	int IRC_NET_SystemReboot( IRC_NET_HANDLE handle)
Parameter	param[in] handle operation handle
Return Value	<u><a href="#">status code</a></u>
Note	

### 33) Set serial port passthrough state

#### **IRC\_NET\_SetTransparentState**

Options	Introduction
Description	Set serial port passthrough state

Function	<pre>int IRC_NET_SetTransparentState(     IRC_NET_HANDLE handle,     int state)</pre>
Parameter	param[in] handle operation handle param[in] state Passthrough state, 0: close 1: open
Return Value	<a href="#">status code</a>
Note	

### 34) Data serial port passthrough

#### **IRC\_NET\_TransparentData**

Options	Introduction
Description	Data serial port passthrough
Function	<pre>int IRC_NET_TransparentData(     IRC_NET_HANDLE handle,     const char* sendBuf,     int sendBufSize,     char* recvBuf,     int recvBufInSize,     int* recvBufOutSize,     int timeout)</pre>
Parameter	param[in] handle operation handle param[in] sendBuf Send command Buffer

	param[in] sendBufSize Send command size param[out] recvBuf Receive command Buffer param[in] recvBufInSize Receive command size, 1024 max param[out] recvBufOutSize True receive command size param[in] timeout timeout, unit: ms
Return Value	<a href="#">status code</a>
Note	need to use the 'IRC_NET_SetTransparentState' function to enable transparent transmission

### 35) Get wiper configuration

#### **IRC\_NET\_GetWiperConfigInfo**

Options	Introduction
Description	Get wiper configuration information
Function	int IRC_NET_GetWiperConfigInfo( IRC_NET_HANDLE handle, int* wiperMode);
Parameter	param[in] handle operation handle param[out] wiperMode wiper mode 0 Auto, 1 Manual
Return Value	<a href="#">status code</a>
Note	

### 36) Set wiper configuration

#### **IRC\_NET\_SetWiperConfigInfo**

Options	Introduction
Description	Set wiper configuration information
Function	<pre>int IRC_NET_SetWiperConfigInfo(     IRC_NET_HANDLE handle,     int wiperMode);</pre>
Parameter	<p>param[in] handle operation handle param[in] wiperMode wiper mode 0 Auto, 1 Manual</p>
Return Value	<a href="#">status code</a>
Note	

### 37) Get fill light configuration

#### **IRC\_NET\_GetFillLightConfigInfo**

Options	Introduction
Description	Get fill light configuration
Function	<pre>int IRC_NET_GetFillLightConfigInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_FILL_LIGHT_CONFIG_INFO</a>*     fillLightConfigInfo);</pre>
Parameter	param[in] handle operation handle

	param[out] fillLightConfigInfo fill light configuration
Return Value	<a href="#">status code</a>
Note	

### 38) Set fill light configuration

#### IRC\_NET\_SetFillLightConfigInfo

Options	Introduction
Description	Data Passthrough
Function	<pre>int IRC_NET_SetFillLightConfigInfo(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_FILL_LIGHT_CONFIG_INFO</a>*     fillLightConfigInfo);</pre>
Parameter	param[in] handle operation handle param[in] fillLightConfigInfo fill light configuration
Return Value	<a href="#">status code</a>
Note	

### 39) Power restart IRC\_NET\_SystemRestart

Options	Introduction
Description	Power Restart
Function	<pre>int IRC_NET_SystemRestart(     IRC_NET_HANDLE handle);</pre>

Parameter	param[in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

#### 40)Get Web logo image IRC\_NET\_GetLogoPicture

Options	Introduction
Description	Get Web Logo image
Function	<pre>int IRC_NET_GetLogoPicture(     IRC_NET_HANDLE handle,     int id,     const char* filePath);</pre>
Parameter	param[in] handle operation handle param[in] id 0-login interface logo,1-main interface logo param[in] filePath file path
Return Value	<a href="#">status code</a>
Note	示例？

#### 41)Set Web logo image IRC\_NET\_SetLogoPicture

Options	Introduction
Description	Set Web Logo image
Function	<pre>int IRC_NET_SetLogoPicture(</pre>

	<code>IRC_NET_HANDLE handle, int id, const char* filePath);</code>
Parameter	<code>param[in] handle operation handle param[in] id 0-login interface logo,1-main interface logo param[in] filePath file path</code>
Return Value	<a href="#">status code</a>
Note	示例?

## 42)Get IR contrast `IRC_NET_GetThermalImageContrast`

Options	Introduction
Description	Get IR contrast
Function	<code>int IRC_NET_GetThermalImageContrast( IRC_NET_HANDLE handle, int* contrast);</code>
Parameter	<code>param[in] handle operation handle param[out] contrast contrast, 0-100</code>
Return Value	<a href="#">status code</a>
Note	

## 43)Set IR contrast `IRC_NET_SetThermalImageContrast`

Options	Introduction
Description	Set IR contrast
Function	<pre>int IRC_NET_SetThermalImageContrast(     IRC_NET_HANDLE handle,     int contrast);</pre>
Parameter	param[in] handle operation handle param[in] contrast contrast, 0-100
Return Value	<a href="#">status code</a>
Note	

#### 44) Get IR brightness

##### **IRC\_NET\_GetThermalImageLuminance**

Options	Introduction
Description	Get IR brightness
Function	<pre>int IRC_NET_GetThermalImageLuminance(     IRC_NET_HANDLE handle,     int* luminance);</pre>
Parameter	param[in] handle operation handle param[out] luminance brightness, 0-100
Return Value	<a href="#">status code</a>
Note	

## 45) Set IR brightness IRC\_NET\_SetThermalImageLuminance

Options	Introduction
Description	Set IR brightness
Function	<pre>int IRC_NET_SetThermalImageLuminance(     IRC_NET_HANDLE handle,     int luminance);</pre>
Parameter	<pre>param[in] handle operation handle param[in] luminance brightness, 0-100</pre>
Return Value	<a href="#">status code</a>
Note	

## 46) Get IR image flip IRC\_NET\_GetThermalImageFlipMode

Options	Introduction
Description	Get IR image flip
Function	<pre>int IRC_NET_GetThermalImageFlipMode(     IRC_NET_HANDLE handle,     int* flipMode);</pre>
Parameter	<pre>param[in] handle operation handle param[out] flipMode flip mode (0:normal,                                 1:horizontal flip, 2:vertical flip, 3:180° flip)</pre>
Return Value	<a href="#">status code</a>

Note	
------	--

#### 47) Set IR image flip **IRC\_NET\_SetThermalImageFlipMode**

Options	Introduction
Description	Set IR image flip
Function	<pre>int IRC_NET_SetThermalImageFlipMode(     IRC_NET_HANDLE handle,     int flipMode);</pre>
Parameter	param[in] handle operation handle param[in] flipMode flip mode (0:normal, 1:horizontal flip, 2:vertical flip, 3:180° flip)
Return Value	<a href="#">status code</a>
Note	

#### 48) Get IR image enhance param

##### **IRC\_NET\_GetThermalImageEnhanceInfo**

Options	Introduction
Description	Get IR image enhance param
Function	<pre>int IRC_NET_GetThermalImageEnhanceInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_THERMAL_IMAGE_MODE_ENHANCE_INFO</a>* thermalImageEnhanceInfo);</pre>

Parameter	param[in] handle operation handle param[out] thermalImageEnhanceInfo IR image enhance param
Return Value	<a href="#">status code</a>
Note	

#### 49) Set IR image enhance param

##### **IRC\_NET\_SetThermalImageEnhanceInfo**

Options	Introduction
Description	Set IR image enhance param
Function	int IRC_NET_SetThermalImageEnhanceInfo( IRC_NET_HANDLE handle, <a href="#">IRC_NET_THERMAL_IMAGE_MODE_ENHANCE_INFO</a> * thermalImageEnhanceInfo);
Parameter	param[in] handle operation handle param[in] thermalImageEnhanceInfo IR image enhance param
Return Value	<a href="#">status code</a>
Note	

#### 50) Get visible day night mode param

##### **IRC\_NET\_GetDayNightModeParam**

Options	Introduction
Description	Get visible day night mode param
Function	<pre>int IRC_NET_GetDayNightModeParam(     IRC_NET_HANDLE handle,     <a href="#"><u>IRC_NET_DAY_NIGHT_MODE_PARAM</u></a>*     dayNightModeParam);</pre>
Parameter	param[in] handle operation handle param[out] dayNightModeParam visible day night mode param
Return Value	<a href="#"><u>status code</u></a>
Note	

## 51) Set visible day night mode param

### **IRC\_NET\_SetDayNightModeParam**

Options	Introduction
Description	Set visible day night mode param
Function	<pre>int IRC_NET_SetDayNightModeParam(     IRC_NET_HANDLE handle,     const <a href="#"><u>IRC_NET_DAY_NIGHT_MODE_PARAM</u></a>*     dayNightModeParam);</pre>
Parameter	param[in] handle operation handle param[in] dayNightModeParam visible day night

	mode param
Return Value	<a href="#">status code</a>
Note	

## 5.2 PTZ Control

### 1) Pan tilt control operation **IRC\_NET\_PtzControl**

Options	Introduction
Description	Pan tilt control operation
Function	<pre>int IRC_NET_PtzControl(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_PTZ_CONTROL</a>* ptzControl);</pre>
Parameter	Param [in] handle operation handle Param [in] ptzControl pan tilt control
Return Value	<a href="#">status code</a>
Note	

### 2) Get the status of PTZ auxiliary command

#### **IRC\_NET\_GetPtzAuxFuncState**

Options	Introduction
Description	Get the status of PTZ auxiliary command
Function	<pre>int IRC_NET_GetPtzAuxFuncState(     IRC_NET_HANDLE handle,</pre>

	<u><a href="#">IRC_NET_PTZ_AUX_FUNC_STATE</a></u> *
	ptzAuxFuncState);
Parameter	Param [in] handle operation handle Param [out] ptzAuxFuncState PTZ auxiliary function status
Return Value	<a href="#">status code</a>
Note	

### 3) Execute laser ranging [IRC\\_NET\\_GetLaserDistance](#)

Options	Introduction
Description	Execute laser ranging
Function	int IRC_NET_GetLaserDistance( IRC_NET_HANDLE handle, int* distance);
Parameter	Param [in] handle operation handle Param [out] distance Laser Ranging Distance
Return Value	<a href="#">status code</a>
Note	need to enable laser ranging first (enable it on the Web)

#### 4) Get the number of presets

##### **IRC\_NET\_QueryPtzPresetSize**

Options	Introduction
Description	Get the number of presets
Function	<pre>int IRC_NET_QueryPtzPresetSize(     IRC_NET_HANDLE handle,     int* size);</pre>
Parameter	Param [in] handle operation handle Param [out] size query quantity
Return Value	<a href="#">status code</a>
Note	

#### 5) Get preset information    **IRC\_NET\_QueryPtzPreset**

Options	Introduction
Description	Get preset information
Function	<pre>int IRC_NET_QueryPtzPreset(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_PTZ_PRESET_INFO</a> ptzPresetInfo[],     int inSize,     int* outSize);</pre>
Parameter	Param [in] handle operation handle

	Param ptzPresetInfo Param [in] inSize preset input size Param [out] outSize preset output size
Return Value	<a href="#">status code</a>
Note	

## 6) Preset control    **IRC\_NET\_PtzPresetControl**

Options	Introduction
Description	Preset control
Function	<pre>int IRC_NET_PtzPresetControl(     IRC_NET_HANDLE handle,     int ptzPresetCmd,     const <a href="#">IRC_NET_PTZ_PRESET_INFO</a>*     ptzPresetInfo);</pre>
Parameter	Param [in] handle operation handle Param [in] ptzPrepatCmd preset control command, refer to IRC-NET-PTZ-PRESET-CMD-TYPE Param [in] ptzPresetInfo preset information
Return Value	<a href="#">status code</a>
Note	

## 7) Get current preset ID    **IRC\_NET\_QueryPtzPresetId**

Options	Introduction
Description	Get current preset ID
Function	IRC_NET_QueryPtzPresetId(IRC_NET_HANDLE handle, int* ptzPresetId)
Parameter	param[in] handle operation handle param[out] ptzPresetId current preset ID
Return Value	<a href="#">status code</a>
Note	

## 8) Get the number of tour configurations

### **IRC\_NET\_QueryPtzTourSize**

Options	Introduction
Description	Get the number of tour configurations
Function	int IRC_NET_QueryPtzTourSize( IRC_NET_HANDLE handle, int* size);
Parameter	Param [in] handle operation handle Param [out] size query quantity
Return Value	<a href="#">status code</a>
Note	

## 9) Get tour configuration information

### **IRC\_NET\_QueryPtzTour**

Options	Introduction
Description	Get cruise group configuration information
Function	<pre>int IRC_NET_QueryPtzTour(     IRC_NET_HANDLE handle,     int id,     <a href="#">IRC_NET_PTZ_TOUR_INFO</a> ptzTourInfo[],     int inSize,     int*outSize) ;</pre>
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] id Cruise group id, range [1255], query all when the value is -1</p> <p>Param ptzTourInfo cruise group configuration</p> <p>Param [in] inSize Cruise group input size</p> <p>Param [out] outSize Cruise group output size</p>
Return Value	<a href="#">status code</a>
Note	

## 10) Tour control    **IRC\_NET\_PtzTourControl**

Options	Introduction
---------	--------------

Description	Tour control
Function	<pre>int IRC_NET_PtzTourControl(     IRC_NET_HANDLE handle,     int ptzTourCmd,     const <a href="#">IRC_NET_PTZ TOUR INFO</a>* ptzTourInfo);</pre>
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] ptzTourCmd tour control command, refer to <a href="#">IRC_NET_PTZ TOUR CMD TYPE</a></p> <p>Param [in] ptzTourInfo tour information.</p>
Return Value	<a href="#">status code</a>
Note	

## 11)Get pattern configuration quantity

### **IRC\_NET\_QueryPtzPatternSize**

Options	Introduction
Description	Get pattern configuration quantity
Function	<pre>int IRC_NET_QueryPtzPatternSize(IRC_NET_HANDLE handle, int* size);</pre>
Parameter	param[in] handle operation handle param[out] size query quantity
Return	<a href="#">status code</a>

Value	
Note	

## 12) Get pattern configuration

### IRC\_NET\_QueryPtzPattern

Options	Introduction
Description	Get pattern configuration
Function	<pre>int IRC_NET_QueryPtzPattern(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_PTZ_PATTERN_CONFIG</a>* ptzPatternConfig);</pre>
Parameter	Param [in] handle operation handle Param [out] ptzPatternConfig patrol configuration
Return Value	<a href="#">status code</a>
Note	

## 13) Patrol control    IRC\_NET\_PtzPatternControl

Options	Introduction
Description	Patrol control
Function	<pre>int IRC_NET_PtzPatternControl(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_PTZ_PATTERN_CONFIG</a>*</pre>

	<code>ptzPatternControl);</code>
Parameter	<p>Param [in] handle operation handle</p> <p>Param [in] ptzPatternControl Patrol control</p>
Return Value	<a href="#">status code</a>
Note	

## 14) PTZ restore default configuration

### **IRC\_NET\_ResetPtzConfig**

Options	Introduction
Description	Cloud tilt restoration default configuration
Function	<pre>int IRC_NET_ResetPtzConfig(     IRC_NET_HANDLE handle);</pre>
Parameter	Param [in] handle operation handle
Return Value	<a href="#">status code</a>
Note	

## 15) PTZ precise positioning

### **IRC\_NET\_PtzPrecisePosition**

Options	Introduction
Description	Pan tilt precise positioning
Function	<pre>int IRC_NET_PtzPrecisePosition(     IRC_NET_HANDLE handle,</pre>

	const <a href="#">IRC_NET_PTZ_POSITION_PARAM</a> * positionParam);
Parameter	Param [in] handle operation handle Param [in] positionConfigure precise positioning configuration
Return Value	<a href="#">status code</a>
Note	

## 16)3D positioning [IRC\\_NET\\_Ptz3DPosition](#)

Options	Introduction
Description	3D positioning
Function	int IRC_NET_Ptz3DPosition( IRC_NET_HANDLE handle, const <a href="#">IRC_NET_PTZ_3D_POSITION_PARAM</a> * positionParam);
Parameter	Param [in] handle operation handle Param [in] positionParam 3D positioning configuration
Return Value	<a href="#">status code</a>
Note	

## 17) Pan/Tilt control IRC\_NET\_SwivelControl

Options	Introduction
Description	Pan/Tilt Control
Function	<pre>int IRC_NET_SwivelControl(     IRC_NET_HANDLE handle,     int swivelCmd);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] swivelCmd Pan/Tilt control command, refer to <a href="#">IRC_NET_SWIVEL_CMD_TYPE</a></p>
Return Value	<a href="#">status code</a>
Note	

## 18) Region focus IRC\_NET\_PtzRegionFocus

Options	Introduction
Description	Region focus
Function	<pre>int IRC_NET_PtzRegionFocus(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_PTZ_REGION_FOCUS_PARAM</a>*     focusParam)</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] focusParam region focus parameters</p>

Return Value	<a href="#">status code</a>
Note	

## 19) Manual tracking IRC\_NET\_PtzManualTrack

Options	Introduction
Description	Manual tracking
Function	IRC_NET_PtzManualTrack( IRC_NET_HANDLE handle, const <a href="#">IRC_NET_PTZ_MANUAL_TRACK_PARAM</a> * trackParam)
Parameter	param[in] handle operation handle param[in] trackParam manual tracking parameters
Return Value	<a href="#">status code</a>
Note	

## 20) Lens initialization IRC\_NET\_PtzLensInit

Options	Introduction
Description	Lens initialization
Function	IRC_NET_PtzLensInit( IRC_NET_HANDLE handle, int channel)

Parameter	param[in] handle operation handle param[in] channel channel No.
Return Value	<a href="#">status code</a>
Note	

## 21) Get PTZ position    **IRC\_NET\_GetCurrentPtz**

Options	Introduction
Description	Get Gimbal Position
Function	int IRC_NET_GetCurrentPtz(  IRC_NET_HANDLE handle,  <a href="#">IRC_NET_PTZ_POSITION_PARAM</a> *  positionParam)
Parameter	param[in] handle operation handle param[out] positionParam Gimbal position parameter
Return Value	<a href="#">status code</a>
Note	

## 22)    Get PTZ position\_V1

### **IRC\_NET\_GetCurrentPtz\_V1**

Options	Introduction
Description	Get Gimbal Position_V1
Function	int IRC_NET_GetCurrentPtz_V1(

	<pre>IRC_NET_HANDLE handle, <a href="#"><u>IRC_NET_PTZ_POSITION_PARAM_V1</u></a>*  positionParam);</pre>
Parameter	param[in] handle operation handle param[out] positionParam Gimbal position parameter
Return Value	<a href="#"><u>status code</u></a>
Note	

## 23) Get area scanning information

### **IRC\_NET\_QueryPtzRegionScanInfo**

Options	Introduction
Description	Get area scanning information
Function	<pre>int IRC_NET_QueryPtzRegionScanInfo(     IRC_NET_HANDLE handle, <a href="#"><u>IRC_NET_REGION_SCAN_INFO</u></a>     regionScanInfos[],     int inSize,     int* outSize)</pre>
Parameter	param[in] handle operation handle param[out] regionScanInfos[] area scanning information param[in] inSize Area Scanning insize

	param[out] outSize Area Scanning outsize
Return Value	<a href="#">status code</a>
Note	

## 24) Set area scanning information

### **IRC\_NET\_SetPtzRegionScanInfo**

Options	Introduction
Description	Set Area Scanning Information
Function	<pre>int IRC_NET_SetPtzRegionScanInfo(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_REGION_SCAN_INFO</a>*     regionScanInfo)</pre>
Parameter	param[in] handle operation handle param[in] regionScanInfo area scanning information
Return Value	<a href="#">status code</a>
Note	

## 25) Delete area scanning information

### **IRC\_NET\_DeletePtzRegionScanInfo**

Options	Introduction
Description	Delete Area Scanning Information
Function	<pre>int IRC_NET_DeletePtzRegionScanInfo(</pre>

	<code>IRC_NET_HANDLE handle,</code> <code>int id)</code>
Parameter	<code>param[in] handle operation handle</code> <code>param[in] id Area Scanning id, refer to</code> <a href="#"><u>IRC_NET_REGION_SCAN_INFO</u></a> , <code>id ≥ 1, id = -1:</code> <code>delete all</code>
Return Value	<a href="#"><u>status code</u></a>
Note	

## 26)Area scanning control

### **IRC\_NET\_PtzRegionScanControl**

Options	Introduction
Description	Area Scanning Control
Function	<code>int IRC_NET_PtzRegionScanControl(</code> <code>IRC_NET_HANDLE handle,</code> <code>int id,</code> <code>bool state)</code>
Parameter	<code>param[in] handle operation handle</code> <code>param[in] id Area Scanning id, refer to</code> <a href="#"><u>IRC_NET_REGION_SCAN_INFO</u></a> <code>param[in] state Area scanning state, true-start,</code> <code>false-stop</code>

Return Value	<a href="#">status code</a>
Note	

## 27) Get boot-up actions

### **IRC\_NET\_QueryPtzBootActionInfo**

Options	Introduction
Description	Get Boot-up Actions
Function	<pre>int IRC_NET_QueryPtzBootActionInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_BOOT_ACTION_INFO</a>* bootActionInfo)</pre>
Parameter	param[in] handle operation handle param[out] bootActionInfo Boot-up Actions
Return Value	<a href="#">status code</a>
Note	

## 28) Set boot-up actions    **IRC\_NET\_SetPtzBootActionInfo**

Options	Introduction
Description	Set Boot-up Actions
Function	<pre>int IRC_NET_SetPtzBootActionInfo(     IRC_NET_HANDLE handle,     const <a href="#">IRC_NET_BOOT_ACTION_INFO</a>* bootActionInfo)</pre>

	bootActionInfo)
Parameter	param[in] handle operation handle param[in] bootActionInfo Boot-up Actions
Return Value	<a href="#">status code</a>
Note	

## 29) Get idle actions `IRC_NET_QueryPtzParkActionInfo`

Options	Introduction
Description	Get Idle Actions
Function	<pre>int IRC_NET_QueryPtzParkActionInfo(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_PARK_ACTION_INFO</a>* parkActionInfo)</pre>
Parameter	param[in] handle operation handle param[out] parkActionInfo Idle Actions
Return Value	<a href="#">status code</a>
Note	

## 30) Set idle actions `IRC_NET_SetPtzParkActionInfo`

Options	Introduction
Description	Set Idle Actions
Function	<code>int IRC_NET_SetPtzParkActionInfo(</code>

	<pre>IRC_NET_HANDLE handle, const <a href="#">IRC_NET_PARK_ACTION_INFO</a>* parkActionInfo)</pre>
Parameter	param[in] handle operation handle param[in] parkActionInfo Idle Actions
Return Value	<a href="#">status code</a>
Note	

### 31) Set precise tracking positioning

#### **IRC\_NET\_PtzTrackingPosition**

Options	Introduction
Description	Set precise tracking positioning
Function	<pre>int IRC_NET_PtzTrackingPosition(     IRC_NET_HANDLE handle,     const     <a href="#">IRC_NET_PTZ_TRACKING_POSITION_PARAM</a>     * positionParam);</pre>
Parameter	param[in] handle operation handle param[in] positionParam precise tracking positioning param
Return Value	<a href="#">status code</a>
Note	

## 32) Get zoom IRC\_NET\_GetPtzZoomMultiplier

Options	Introduction
Description	Get zoom
Function	<pre>int IRC_NET_GetPtzZoomMultiplier(     IRC_NET_HANDLE handle,     int channel,     float* zoomMultiplier);</pre>
Parameter	<p>param[in] handle operation handle</p> <p>param[in] channel channel</p> <p>param[out] zoomMultiplier zoom value</p>
Return Value	<a href="#">status code</a>
Note	

## 33) Set zoom IRC\_NET\_SetPtzZoomMultiplier

Options	Introduction
Description	Set zoom
Function	<pre>int IRC_NET_SetPtzZoomMultiplier(     IRC_NET_HANDLE handle,     int channel,     float zoomMultiplier);</pre>
Parameter	param[in] handle operation handle

	param[in] channel channel param[in] zoomMultiplier zoom value
Return Value	<a href="#">status code</a>
Note	

### 34) Get acceleration related data

#### **IRC\_NET\_GetAccelerationData**

Options	Introduction
Description	Get acceleration related data
Function	<pre>int IRC_NET_GetAccelerationData(     IRC_NET_HANDLE handle,     <a href="#">IRC_NET_ACCELERATION_DATA</a>*     accelerationData);</pre>
Parameter	param[in] handle operation handle param[out] accelerationData acceleration related data
Return Value	<a href="#">status code</a>
Note	

### 35) Get linkage tracking config

#### **IRC\_NET\_GetTargetTrackConfig**

Options	Introduction
Description	Get Linkage tracking config

Function	<pre>int IRC_NET_GetTargetTrackConfig(     IRC_NET_HANDLE handle,     int channel,     <a href="#">IRC_NET_TARGET_TRACK_DATA</a>* targetTrackData)</pre>
Parameter	param[in] handle operation handle param[in] channel channel param[out] targetTrackData Linkage tracking config
Return Value	<a href="#">status code</a>
Note	

### 36) Set linkage tracking config

#### **IRC\_NET\_SetTargetTrackConfig**

Options	Introduction
Description	Set Linkage tracking config
Function	<pre>int IRC_NET_SetTargetTrackConfig(     IRC_NET_HANDLE handle,     int channel,     const <a href="#">IRC_NET_TARGET_TRACK_DATA</a>* targetTrackData);</pre>
Parameter	param[in] handle operation handle param[in] channel channel

	param[in] targetTrackData Linkage tracking config
Return Value	<a href="#">status code</a>
Note	

# Chapter 4 Data Introduction

## 1. IRC\_NET\_DEV\_SEARCH\_INFO

### Structure Definition

```
typedef struct

{
    char ip[IRC_NET_IP_LEN_MAX]; ///
```

## 2. IRC\_NET\_LOGIN\_INFO

### Structure Definition

```
typedef struct

{
    char ip[16]; ///
```

### **3. IRC\_NET\_DEV\_INFO**

#### Structure Definition

```
typedef struct

{
    int channelNum; ///< number of channels

    int optChannel; ///< visible light channel

    int irChannel; ///< infrared channel

    int productGeneration; ///<product generation

}
```

### **4. IRC\_NET\_PREVIEW\_INFO**

#### Structure Definition

```
typedef struct

{
    int channel; ///< channel

    int streamType; ///< stream type, refer to

IRC_NET_STREAM_TYPE

    int frameFmt; ///< frame format, refer to

IRC_NET_FRAME_FMT

}
```

## 5. IRC\_NET\_POINT

### Structure Definition

```
typedef struct  
{  
    int x; ///  
    int y; ///  
}
```

## 6. IRC\_NET\_RECT

### Structure Definition

```
typedef struct  
{  
    int top; ///  
    int left; ///  
    int right; ///  
    int bottom; ///  
}
```

## 7. IRC\_NET\_ENV\_PARAM

### Structure Definition

```
typedef struct  
{
```

```

float atmosphereTemp; ///< atmospheric temperature

float distance; ///< target distance

float emissivity; ///< emissivity 0.01-1

float reflectedTemp; ///< reflected temperature

float transmittance; ///< atmospheric transmissivity 0.01-1

}

```

## **8. IRC\_NET\_TEMP\_ALARM\_RULE\_INFO**

### Structure Definition

```

typedef struct

{
    int type; ///< type of alarm rule, refer to

IRC_NET_TEMP_ALARM_RULE_TYPE

    int debounce; ///< time of de-jitter, 5s by default

    float thresholdTemp; ///< temperature threshold

    float toleranceTemp; ///< tolerance temperature

}

```

## **9. IRC\_NET\_TEMP\_RULE\_INFO**

### Structure Definition

```

typedef struct

{

```

```

bool enable; ///< enable switch

int presetId; ///< preset Id, start from 0

int id; ///< temperature rule ID, start from 1, invalid when adding
rules

char name[IRC_NET_NAME_LEN_MAX]; ///< name of
temperature measurement rule

int type; ///< type of temperature measurement rule, refer to
IRC_NET_TEMP_RULE_TYPE

IRC_NET_POINT points[IRC_NET_POINT_MAX]; ///<
coordinates of temperature measurement rule

int pointNum; ///< Number of coordinates in temperature
measurement rules

bool envParamEnable; ///< enable switch of environmental
parameters

IRC_NET_ENV_PARAM envParam; ///< environmental
parameters of the rule

IRC_NET_TEMP_ALARM_RULE_INFO alarmRuleInfo;
///< alarm rule info

}

```

## **10.IRC\_NET\_TEMP\_RULE\_INFO\_G1**

Structure Definition
----------------------

```
typedef struct

{

    int id; ///
```

```
}
```

## 11.IRC\_NET\_TEMP\_RULE\_INDEX

### Structure Definition

```
typedef struct

{
    int presetId; ///</i>preset Id, start from 0

    int type; ///</i>type of temperature measurement rule, refer to

IRC_NET_TEMP_RULE_TYPE

    int id; ///</i>ID or temperature measurement rule, start from 1

}
```

## 12.IRC\_NET\_TEMP\_INFO

### Structure Definition

```
typedef struct

{
    float avgTemp; ///</i>average temperature

    float centerTemp; ///</i>center temperature

    float maxTemp; ///</i>highest temperature

    float minTemp; ///</i>lowest temperature

    IRC\_NET\_POINT maxTempPoint; ///</i>highest temperature
```

coordinate

[IRC\\_NET\\_POINT](#) minTempPoint;///< lowest temperature

coordinate

}

## 13.IRC\_NET\_RULE\_TEMP\_INFO

### Structure Definition

```
typedef struct

{
    IRC_NET_TEMP_RULE_INDEX ruleIndex;

    IRC_NET_TEMP_INFO tempInfo;

}
```

## 14.IRC\_NET OSD\_TIME\_TITLE\_INFO

### Structure Definition

```
typedef struct

{
    bool enable; ///< enable

    IRC_NET_RECT rect; ///< title position, 8192 coordinate system

}
```

## **15.IRC\_NET OSD CHANNEL TITLE INFO**

### Structure Definition

```
typedef struct

{
    bool enable; ///< enable

    char name[IRC_NET_NAME_LEN_MAX]; ///< channel name,
    UTF-8 string, maximum of 64 characters or Chinese characters.

    IRC_NET_RECT rect; ///< title position, 8192 coordinate system

}
```

## **16.IRC\_NET TEMP SPAN INFO**

### Structure Definition

```
typedef struct

{
    bool enable; ///< enable

    float lowTemp; ///< low tempeature threshold

    float highTemp; ///< high temperature thersholt

}
```

## **17.IRC\_NET ALARM BASE INFO**

### Structure Definition

```
typedef struct
```

```
{  
  
    int channel; ///< channel  
  
    int64_t timestamp; ///< time stamp  
  
    int alarmAction; ///< alarm action, refer to  
  
IRC_NET_ALARM_ACTION  
  
}
```

## 18.IRC\_NET\_ALARM\_TEMP\_INFO

### Structure Definition

```
typedef struct  
  
{  
  
    IRC_NET_ALARM_BASE_INFO alarmBaseInfo; ///< basic  
    alram information  
  
    IRC_NET_RULE_TEMP_INFO ruleTempInfo; ///< rule  
    temperature information  
  
    int tempUnit; ///< temperature unit, refer to  
  
IRC_NET_TEMP_UNIT  
  
    int alarmRuleType; ///< type of alarm rule, refer to  
  
IRC_NET_TEMP_ALARM_RULE_TYPE  
  
    float thresholdTemp; ///< threshold temperature  
  
}
```

## **19.IRC\_NET\_DEV\_ABILITY\_QUERY\_CONDITION**

### Structure Definition

```
typedef struct

{
    int channel; ///< thoroughfare

    int type; ///< equipment capability type, refer to

IRC_NET_DEV_ABILITY_TYPE

}
```

## **20.IRC\_NET\_PTZ\_ABILITY**

### Structure Definition

Pan tilt detailed capability

Typedef struct

```
{

    Int Zoom///< zoom

    Int focus///< focusing

    Int iris///< aperture

    Int ptz///< gimbal

    Int wiper///< wiper blade

    Int light///< fill light

    Int defrost///< defrosting
```

```
Int defog///< translucent fog  
Int fan///< fan  
Int heater///< heater  
Int automatic///< auto focus  
Int synchronousView///< bidirectional follow-up  
Int ptzPosition///< accurate positioning  
Int threeDPosition///< 3D positioning  
}
```

## 21.IRC\_NET\_PTZ\_CONTROL

Structure Definition
<pre>Pan tilt direction control Typedef struct {     Int channel///&lt; passageway     Int type///&lt; pan tilt control function type, refer to IRC-NET-PTZ-CMD-TYPE     Int param1///&lt; parameter 1, please refer to the document for details on gimbal command parameters     Int param2///&lt; parameter 2, please refer to the document for the PTZ command parameters     Int param3///&lt; parameter 3, please refer to the document for the</pre>

PTZ command parameters

Int stop///< whether to stop, 0-start, 1-stop

}

## 22.IRC\_NET\_PTZ\_AUX\_FUNC\_STATE

Structure Definition

Enabling status of pan tilt assist function

Typedef struct

{

Int wiperState///< wiper status

Int lightState///< fill light status

Int derostState///< defrosting status

Int defogState///< foggy state

Int fanState///< fan status

Int heaterState///< heater status

Int automaticState///< auto focus state

Int synchronousViewState///< bidirectional follow-up state

}

## 23.IRC\_NET\_PTZ\_PRESET\_INFO

Structure Definition

Pre set point information

```
Typedef struct

{
    Bool enable;

    Int id///< preset point ID, starting from 0

    Char name [IRC-NET-NAME-LEN-MAX]///< name of preset
point
}
```

## 24.IRC\_NET\_PTZ TOUR\_INFO

### Structure Definition

Cruise group configuration information

```
Typedef struct

{
    Bool enable;

    Int id///< cruise group ID, starting from 1

    Char name [IRC-NET-NAME-LEN-MAX]///< cruise group point
name

    Bool running///< running state

    IRC-NET-PTZ-TOUR-PRESET-INFO presetInfos

    [IRC-NET-TOUR-PRESET-NUM-MAX]///< cruise group preset point
information

    Int presetNum///< number of preset cruise control points
```

```
}
```

## 25.IRC\_NET\_PTZ\_TOUR\_PRESET\_INFO

### Structure Definition

Cruise group preset point information

Typedef struct

```
{
```

    IRC-NET-PTZ-PRESET-INFO presetInfo///< cruise group preset  
    point information

    Int residenceTime///< residence time

```
}
```

## 26.IRC\_NET\_PTZ\_PATTERN\_CONFIG

### Structure Definition

Patrol configuration

Typedef struct

```
{
```

    IRC-NET-PTZ-PATTERN\_INFO patternInfo  
    [IRC-NET-PATTERN\_NUM-MAX]///< patrol information  
}

## **27.IRC\_NET\_PTZ\_PATTERN\_INFO**

Structure Definition

Patrol information

Typedef struct

{

    Bool enable///< enable

    Int id///< patrol ID, range [1, 5]

    Bool running///< running state

}

## **28.IRC\_NET\_PTZ\_PATTERN\_CONTROL**

Structure Definition

Patrol control

Typedef struct

{

    Int type///< patrol control function type, refer to

    IRC-NET-PTZ-PATTERN\_CMD-TYPE

    Int id///< patrol ID, range [1, 5]

}

## **29.IRC\_NET\_PTZ\_POSITION\_PARAM**

Structure Definition

Pan tilt precise positioning configuration

Typedef struct

{

    Float pan///< horizontal angle, 0-36000, =P\*100

    Float tilt///< vertical angle, 0-9000, =T\*100

    Float Zoom///< doubling, 0-16384. For precise zoom adjustment,  
    use the "Get zoom" and "Set zoom" functions.

}

## **30.IRC\_NET\_PTZ\_POSITION\_PARAM\_V1**

Structure Definition

Pan tilt precise positioning configuration

Typedef struct

{

    float pan; ///< Horizontal position, -1-360

    float tilt; ///< Vertical position, -1-90

    float visZoom; ///< VIS zoom value

    float irZoom; ///< IR zoom value

    float visFovH; ///< VIS Horizontal FOV

        float visFovV; ///< VIS vertical FOV

        float irFovH; ///< IR Horizontal FOV

        float irFovV; ///< IR vertical FOV

```
}
```

## 31.IRC\_NET\_PTZ\_3D\_POSITION\_PARAM

### Structure Definition

3D positioning configuration

Typedef struct

```
{
```

Int channel///< passageway

[IRC\\_NET\\_POINT](#) startPoint///< starting point

[IRC\\_NET\\_POINT](#) endPoint///< end point

```
}
```

## 32.IRC\_NET\_FRAME\_TEMP\_ALARM\_CONFIG

### Structure Definition

Typedef struct

```
{
```

Bool enable///< enable switch

[IRC\\_NET\\_TEMP\\_ALARM\\_RULE\\_INFO](#)

alarmRuleInfo[IRC\_NET\_FRAME\_TEMP\_RULE\_NUM\_MAX]///<

alarm rule information

[IRC\\_NET\\_ALARM\\_LINKAGE\\_SET](#) alarmLinkage///< alarm linkage

```
}
```

### 33.IRC\_NET\_FRAME\_TEMP\_ALARM\_CONFIG\_G1

#### Structure Definition

```
typedef struct

{
    int alarmType; ///alarm rule type, refer to
IRC\_NET\_TEMP\_ALARM\_RULE\_TYPE\_G1

    IRC\_NET\_TEMP\_ALARM\_RULE\_INFO\_G1
    lowTempAlarmRuleInfo; ///Low temperature alarm rule information

    IRC\_NET\_TEMP\_ALARM\_RULE\_INFO\_G1
    highTempAlarmRuleInfo; ///High temperature alarm rule information

    int debounce; ///time of de_jitter, 5s by default
    bool alarmLinkageSnapshotEnable; ///Alarm-triggered snapshot
enable
}
```

### 34.IRC\_NET\_TEMP\_ALARM\_RULE\_INFO

#### Structure Definition

Temperature measurement alarm rules

Typedef struct

```
{
```

Int type///< alarm rule type, refer to  
**IRC\_NET\_TEMP\_ALARM\_RULE\_TYPE**  
 Int debounce///< dejitter time, default to 5 seconds  
 Float thresholdTemps///< temperature threshold  
 Float tolerance Temps///< tolerance temperature  
 }

## 35.IRC\_NET\_TEMP\_ALARM\_RULE\_INFO\_G1

Structure Definition
Temperature measurement alarm rules <b>typedef struct</b> { float thresholdTemp; ///< temperature threshold bool enable; ///< Level enable switch float tempLevel1; ///< Level 1 float tempLevel2; ///< Level 2 float tempLevel3; ///< Level 3 } 

## 36.IRC\_NET\_ALARM\_LINKAGE\_SET

Structure Definition
Temperature measurement alarm linkage 

```

Typedef struct

{

    IRC\_NET\_ALARM\_LINKAGE\_INFO snapLinkageChannel///<
snap linkage channel

IRC\_NET\_ALARM\_LINKAGE\_INFO

recordLinkageChannel///< video linkage channel

}

```

### **37.[IRC\\_NET\\_ALARM\\_LINKAGE\\_INFO](#)**

Structure Definition
Temperature measurement alarm linkage channel
Typedef struct
{
Bool enable///< enable switch
Bool channel[IRC_NET_CHANNEL_NUM_MAX]///< linkage
channel
Int64_t delay///< linkage delay
}

### **38.IRC\_NET\_TEMP\_MASK\_INDEX**

Structure Definition
Index of temperature measurement shielding area

```
Typedef struct  
{  
    Int presetId///< preset point ID, starting from 0  
    Int id///< block area ID, starting from 1  
}
```

## 39.IRC\_NET\_TEMP\_MASK\_INFO

Structure Definition
Temperature measurement shielding area information  Typedef struct { Bool enable///< enable switch IRC_NET_TEMP_MASK_INDEX index///< block area index Char name [IRC_NET_NAME_LEN_MAX]///< block area name IRC NET POINT points [IRC NET MASK POINT NUM MAX]///< shielded area coordinate points }

## 40.IRC\_NET\_FILE\_QUERY\_PARAM

Structure Definition
SD card query criteria  Typedef struct

{

Int channel///< passageway

Int type///< file type, refer to IRC\_NET\_SD\_FILE\_TYPE

Char startTime [IRC\_NET\_TIME\_LEN\_MAX]///< start time

Char endTime [IRC\_NET\_TIME\_LEN\_MAX]///< end time

Int count///< expected number of returns, default to 50

Int offset///< query position offset, fill in 0 for the first time

}

## 41.IRC\_NET\_FILE\_INFO

### Structure Definition

SD card file information

Typedef struct

{

Int id///< ID number

Int channel///< passageway

Char startTime [IRC\_NET\_TIME\_LEN\_MAX]///< start time

Char endTime [IRC\_NET\_TIME\_LEN\_MAX]///< end time

Char path [IRC\_NET\_FILE\_PATH\_LEN\_MAX]///< file path

}

## **42.IRC\_NET\_FILE\_DOWNLOAD\_INFO**

Structure Definition

SD card file download information

Typedef struct

{

    Int type///< download file type, refer to

    IRC\_NET\_SD\_FILE\_TYPE

    Int channel///< passageway

    Char startTime [IRC\_NET\_TIME\_LEN\_MAX]///< start time

    Char endTime [IRC\_NET\_TIME\_LEN\_MAX]///< end time

    Char filePath [IRC\_NET\_FILE\_PATH\_LEN\_MAX]///< file path

    Char downloadPath [IRC\_NET\_FILE\_PATH\_LEN\_MAX]///<

    download path

}

## **43.IRC\_NET\_FILE\_DOWNLOAD\_PROGRESS**

Structure Definition

SD card file download progress

Typedef struct

{

    Int64\_t totalSize///< total download length, in bytes

    Int64\_t downloadSize///< downloaded length, in bytes

```
}
```

## 44.IRC\_NET\_TEMP\_EXT\_INFO\_CB

### Structure Definition

Extended information for temperature callback

```
typedef struct
```

```
{
```

```
    uint64_t utcTime; ///    uint32_t emiss; //Emissivity  
    uint32_t humidity; //Atmospheric  
    uint32_t reflectTempK10; //Reflected apparent temperature, K*10  
    uint32_t envTempK10; //Ambient temperature, K*10  
    uint32_t distance; //Target distance, m  
    uint32_t sensorTemp; //Sensor temperature, °C  
    int A0; //Distance compensation coefficient  
    int B0; //Distance compensation coefficient  
    int C0; //Distance compensation coefficient  
    int D0; //Distance compensation coefficient  
    int A1; //Distance compensation coefficient  
    int B1; //Distance compensation coefficient  
    int C1; //Distance compensation coefficient  
    int D1; //Distance compensation coefficient
```

```
}
```

## 45.IRC\_NET\_TEMP\_INFO\_CB

### Structure Definition

Temperature information in temperature callback

```
typedef struct
```

```
{
```

```
    char* temp; ///< frame data
```

```
    int width; ///< width
```

```
    int height; ///< height
```

```
}
```

## 46.IRC\_NET\_VIDEO\_INFO\_CB

### Structure Definition

Video callback

```
typedef struct
```

```
{
```

```
    char* frame; ///< frame data
```

```
    int width; ///< width
```

```
    int height; ///< height
```

```
    int validWidth; ///< active width
```

```
    int validHeight; ///< active height
```

```
}
```

## 47.IRC\_NET\_IVS\_INFO\_CB

### Structure Definition

IVS info in video callback

```
typedef struct
```

```
{
```

```
    int baseIvsInfoLen; ///< length of basic IVS info
```

```
    char* baseIvsInfo; ///< basic IVS info
```

```
    int tempIvsInfoLen; ///<length of temperature IVS info
```

```
    char* tempIvsInfo; ///< temperature IVS info
```

```
}
```

## 48.IRC\_NET\_IP\_CONFIG

### Structure Definition

IP configuration

```
typedef struct
```

```
{
```

```
    char name[IRC_NET_NAME_LEN_MAX]; ///< Network card
```

name, read\_only

```
    bool dhcpEnable; ///< dhcp enable
```

```
    char mac[IRC_NET_MAC_LEN_MAX]; ///<MAC address,
```

```

read_only

    char ip[IRC_NET_IP_LEN_MAX]; ///< ip address

    char subnetMask[IRC_NET_IP_LEN_MAX]; ///< Subnet mask

    char gateway[IRC_NET_IP_LEN_MAX]; ///< default gateway

    char defaultDns[IRC_NET_IP_LEN_MAX]; ///< primary DNS

    char standbyDns[IRC_NET_IP_LEN_MAX]; ///< secondary DNS

}

```

## **49.IRC\_NET\_SWIVEL\_CMD\_TYPE**

### Structure Definition

Basic control function types for the pan\_tilt unit

```

typedef enum

{
    IRC_NET_SWIVEL_CMD_UP = 0, ///< Tilt up

    IRC_NET_SWIVEL_CMD_DOWN ///< Tilt down

}

```

## **50.IRC\_NET\_IP\_TARGET\_RECOGNITION\_CONFIG**

### Structure Definition

Channel target recognition configuration

```

typedef struct
{

```

```
bool enable; ///< enable  
  
int sensitivity; ///< sensitivity, range: 0-100, 70 by default  
  
IRC_NET_LINKAGE_SET linkageSet; ///< linkage action  
  
}
```

## 51.IRC\_NET\_PTZ\_REGION\_FOCUS\_PARAM

### Structure Definition

region focus parameters

```
typedef struct  
  
{  
  
    int channel; ///< channel  
  
    IRC_NET_POINT startPoint; ///< Start point coordinates  
  
    IRC_NET_POINT endPoint; ///< End point coordinates  
  
    int stop; ///< whether to stop, 0-start, 1-stop  
  
}
```

## 52.IRC\_NET\_PTZ\_MANUAL\_TRACK\_PARAM

### Structure Definition

Manual track parameters

```
typedef struct  
  
{  
  
    int channel; ///< channel
```

```

IRC_NET_POINT startPoint; ///< Start point coordinates

IRC_NET_POINT endPoint; ///< End point coordinates

int stop; ///< whether to stop, 0-start, 1-stop

{
}

```

## **53.IRC\_NET\_REGION\_SCAN\_INFO**

### Structure Definition

```

area scanning information

typedef struct

{
    int id; ///< area scanning ID

    bool enable; ///< area scanning enable

    int derection; ///< Scan direction, 1-clockwise, -1-Anticlocksize

    int speed; ///< speed

    int startPresetId; ///< start preset ID

    int stopPresetId; ///< end presed ID

    float tiltStepAngle; ///< gradient value
}

```

## **54.IRC\_NET\_BOOT\_ACTION\_INFO**

### Structure Definition

### Boot-up Actions

```

typedef struct

{
    bool enable; ///IRC_NET_ACTION_TYPE
    int lineScanId; ///

```

## **55.Idle Actions IRC\_NET\_PARK\_ACTION\_INFO**

### Structure Definition

```

typedef struct

{
    bool enable; ///IRC_NET_ACTION_TYPE 0: None
    int lineScanId; ///

```

```

int patternId; ///< Pattern id

int tourId; ///< Tour id

int runningFunction; ///< refer to IRC\_NET\_ACTION\_TYPE, 0:
None

int second; ///< Idle time

bool running; ///< 运行状态

}

```

## 56.Laser Distance OSD parameters

### **IRC\_NET\_LASER\_DISTANCE OSD\_PARAM**

#### Structure Definition

```

typedef struct

{
    bool enable; ///< enable

    int rangingDuration; ///< duration of a single ranging

    IRC_NET_LASER_DISTANCE OSD_TITLE_FORMAT

    titleFormat; ///< laser ranging OSD title format

}

```

## 57.Title Format For Laser Distance OSD

### **IRC\_NET\_LASER\_DISTANCE OSD\_TITLE\_FORM AT**

## Structure Definition

```
typedef struct

{
    int alignType; ///< Alignment: 0 Left-aligned, 1 Right-aligned
    int fontSize; ///< Font size: 0 Small, 1 medium, 2 large
    int autoTurn; ///< Invert color treatment: 0 no treatment, 1 automatic
    invert color, 2 hook edge
    int bgColor[IRC_NET_COLOR_TYPE_MAX]; ///< Background
    color
    int fgColor[IRC_NET_COLOR_TYPE_MAX]; ///< Foreground
    color
    IRC_NET_RECT titlePosition; ///< Title position
}
```

## 58.Fill Light Configuration

### **IRC\_NET\_FILL\_LIGHT\_CONFIG\_INFO**

## Structure Definition

```
typedef struct

{
    int fillLightMode; ///< Filllight mode 0 Auto, 1 Manual
    int infraredLight; ///< Infrared Light Control 0-100
    int whiteLight; ///< White Light Control 0-100
```

```

int hFov; ///< Horizontal Field of View of Fill Light 0-100

float hFovFactor; ///< Angle Factor of the horizontal field of view of
fill light

}

```

## **59.IRC\_NET\_THERMAL\_IMAGE\_MODE\_ENHANCE\_INFO**

### Structure Definition

```

IR image enhance param

typedef struct

{
    bool brightMutationSuppression; ///< Brightness mutation
suppression

    IRC\_NET\_VIDEOIN\_LEVEL denoise2D; ///< denoise2D

    IRC\_NET\_VIDEOIN\_LEVEL denoise3D; ///< denoise3D

    IRC\_NET\_VIDEOIN\_LEVEL detailEnhance;///< Detail
enhancement

    IRC\_NET\_REGIONAL\_VIDEO\_ENHANCE

    regionalVideoEnhance;///< Local video enhancement

    int flipMode;///< flipMode flip mode (0:normal, 1:horizontal flip,
2:vertical flip, 3:180° flip)

}

```

## **60.IRC\_NET\_VIDEOIN\_LEVEL**

### Structure Definition

Global video enhancement parameters

```
typedef struct
```

```
{
```

```
    bool enable; ///< enable;
```

```
    int level; ///< level, 0-100;
```

```
}
```

## **61.IRC\_NET\_REGIONAL\_VIDEO\_ENHANCE**

### Structure Definition

Local video enhancement info

```
typedef struct
```

```
{
```

```
    bool enable; ///< enable;
```

```
    IRC\_NET\_REGIONAL\_VIDEO\_ENHANCE\_RANGE range;///<
```

```
    Local video enhancement range;
```

```
}
```

## **62.IRC\_NET\_REGIONAL\_VIDEO\_ENHANCE\_RANGE**

### Structure Definition

Local video enhancement range

```

typedef struct

{
    int level; ///IRC_NET_POINT region[2]; ///

```

## **63.IRC\_NET\_PTZ\_TRACKING\_POSITION\_PARAM**

### Structure Definition

Precise tracking positioning param

```

typedef struct

{
    float panSpeed; ///

```

## **64.IRC\_NET\_ACCELERATION\_DATA**

### Structure Definition

Acceleration-related data

```
typedef struct
```

```

{

    IRC\_NET\_ANGLE\_DATA angleData; ///< Angular acceleration
data

    IRC\_NET\_COORDINATE\_AXIS\_DATA accelerationData; ///<
Linear acceleration per axis, unit:g

    IRC\_NET\_COORDINATE\_AXIS\_DATA rotationData; ///<
Angular acceleration per axis, unit: $^{\circ}$  /s

}

```

## **65.IRC\_NET\_COORDINATE\_AXIS\_DATA**

### Structure Definition

```

Linear acceleration data

typedef struct

{
    float x; ///< x-Axis Motion Parameters

    float y; ///< y-Axis Motion Parameters

    float z; ///< z-Axis Motion Parameters
}
```

## **66.IRC\_NET\_ANGLE\_DATA**

### Structure Definition

Angular acceleration data

```

typedef struct

{
    IRC\_NET\_COORDINATE\_AXIS\_DATA coordinateData; ///<

Acceleration Vector-Axis Angles, unit:°

    float pan; ///< Pan

    float tilt; ///< Tilt

    float rt; ///< Tilt after accelerometer calibration

}

```

## **67.IRC\_NET\_TARGET\_TRACK\_DATA**

### Structure Definition

Linkage tracking config

typedef struct

{

bool enable; ///< enable;

[IRC\\_NET\\_ALARM\\_LINKAGE\\_INFO](#) snapshotLinkageInfo; ///<

Snapshot linkage

[IRC\\_NET\\_ALARM\\_LINKAGE\\_INFO](#) recordLinkageInfo; ///<

Record linkage

}

## **68.IRC\_NET\_DAY\_NIGHT\_MODE\_PARAM**

## Structure Definition

VIS day/night mode param

typedef struct

{

    int dayNightMode; ///    3-time period;

    int delay; ///

    int start; ///

    int end; ///

    int sensitivity; ///    only

}

# Chapter 5 Constant

## 1. status code list IRC\_NET\_ERROR

Definition	Description
typedef enum	
{	
IRC_NET_ERROR_OK = 0, ///< < succeed	
IRC_NET_ERROR_FAILED = 1, ///< < fail	
IRC_NET_ERROR_NOT_SUPPORTED = 2, ///< < unavailable	
IRC_NET_ERROR_PARAM_WRONG = 3, ///< < parameter error	
IRC_NET_ERROR_TEMP_CALLBACK_WRONG = 4,	
///< < temperature callback disabled	
IRC_NET_ERROR_BLACK_LIST = 1001, ///< < user is not on the	
whitelist	
IRC_NET_ERROR_NONE_USER = 1002, ///< < username does not	
exist	
IRC_NET_ERROR_PWD_WRONG = 1003, ///< < password error	
IRC_NET_ERROR_DEV_NOT_SUPPORTED = 1004, ///< <	
capability set has no device model	
IRC_NET_ERROR_ACCOUNT_LOCK = 1005, ///< < account	
locked	

```

IRC_NET_ERROR_USER_LIMIT = 1006, ///< user count exceeds
the limit

IRC_NET_ERROR_SYSTEM_EXCEPTION = 1007, ///< operation
failed

IRC_NET_ERROR_TEMP_RULE_LIMIT = 1101, ///< temperature
measurement rule limit

} IRC_NET_ERROR;

```

## 2. Type of Global Exception

### **IRC\_NET\_EXCEPTION\_TYPE**

Definition Description
<pre> typedef enum {     IRC_NET_EXCEPTION_PREVIEW_OFFLINE = 1001, ///&lt; preview offline      IRC_NET_EXCEPTION_ALARM_OFFLINE = 1002, ///&lt; alarm offline      IRC_NET_EXCEPTION_DEV_OFFLINE = 1003 ///&lt;device offline  } IRC_NET_EXCEPTION_TYPE; </pre>

### **3. Alarm Type    IRC\_NET\_ALARM\_TYPE**

Definition Description
<pre>typedef enum  {     IRC_NET_ALARM_TEMP = 10001, ///&lt; temperature     IRC_NET_ALARM_FIRE   = 10002, ///&lt; fire     IRC_NET_ALARM_TEMP_RISE = 10003, ///&lt; temperature rise     IRC_NET_ALARM_TEMP_DIFF  = 10004, ///&lt; temperature difference     IRC_NET_ALARM_FIRE_PULSE = 10005, ///&lt; fire pulse     IRC_NET_ALARM_REGION_INTRUSION = 20001, ///&lt; area intrusion     IRC_NET_ALARM_LINE_INTRUSION = 20002, ///&lt; tripwire intrusion     IRC_NET_ALARM_SMOKE_DETECT = 30001, ///&lt; smog     IRC_NET_ALARM_LOCAL = 40001 ///&lt; local } IRC_NET_ALARM_TYPE;</pre>

### **4. Frame Forma    IRC\_NET\_FRAME\_FMT**

Definition Description
<pre>typedef enum  {</pre>

```

IRC_NET_FRAME_FMT_YUV420P = 0, ///<> YUV420P

IRC_NET_FRAME_FMT_RGB24, ///<> RGB24

IRC_NET_FRAME_FMT_RGBA, ///<> RGBA

IRC_NET_FRAME_FMT_BGRA ///<> BGRA

} IRC_NET_FRAME_FMT;

```

## 5. Stream Type **IRC\_NET\_STREAM\_TYPE**

Definition Description
<pre> typedef enum  {     IRC_NET_STREAM_MAIN = 0, ///&lt;&gt; main stream      IRC_NET_STREAM_SUB ///&lt;&gt; sub stream  } IRC_NET_STREAM_TYPE; </pre>

## 6. Equipment capability type

### **IRC\_NET\_DEV\_ABILITY\_TYPE**

Definition Description
<pre> typedef enum  {     IRC_NET_DEV_ABILITY_PTZ = 0, ///&lt;&gt; pan tilt capability,     please refer to <u>IRC_NET_PTZ_ABILITY</u>  } </pre>

## 7. Log level **IRC\_NET\_LOG\_LEVEL**

Definition	Description
<pre>typedef enum  {     IRC_NET_LOG_LEVEL_TRACE = 0,     IRC_NET_LOG_LEVEL_DEBUG,     IRC_NET_LOG_LEVEL_INFO,     IRC_NET_LOG_LEVEL_WARN,     IRC_NET_LOG_LEVEL_ERROR }</pre>	

## 8. Temperature measurement alarm rule type

### **IRC\_NET\_TEMP\_ALARM\_RULE\_TYPE**

Definition	Description
<pre>typedef enum  {     IRC_NET_TEMP_ALARM_RULE_NONE=0,///<no <high<="" <the="" average="" greater="" irc_net_temp_alarm_rule_avg_temp_gt,="" irc_net_temp_alarm_rule_avg_temp_lt,="" irc_net_temp_alarm_rule_hight_temp_gt,="" is="" less="" pre="" rule="" temperature="" than=""></no></pre>	

temperature greater than

    IRC\_NET\_TEMP\_ALARM\_RULE\_HIGHT\_TEMP\_LT,///<high

temperature less than

    IRC\_NET\_TEMP\_ALARM\_RULE\_LOW\_TEMP\_GT,///<low

temperature greater than

    IRC\_NET\_TEMP\_ALARM\_RULE\_LOW\_TEMP\_LT//<low

temperature less than

}

## 9. Temperature measurement alarm rule type\_G1

### IRC\_NET\_TEMP\_ALARM\_RULE\_TYPE\_G1

Definition Description
typedef enum
{
IRC_NET_TEMP_ALARM_RULE_G1_NONE = 0, ///< no rules
IRC_NET_TEMP_ALARM_RULE_G1_HIGH_TEMP,   ///< high temp
IRC_NET_TEMP_ALARM_RULE_G1_LOW_TEMP,   ///< low temp
IRC_NET_TEMP_ALARM_RULE_G1_HIGH_LOW_TEMP   ///< high temp & low temp
}

## 10. Basic control function types of pan tilt

### IRC\_NET\_PTZ\_CMD\_TYPE

Definition	Description
Typedef enum	{  IRC_NET_PTZ_CMDUP_UP,///<upper  IRC_NET_PTZ_CMD_DOWN,///<below  IRC_NET_PTZ_CMD_LEFT,///<left  IRC_NET_PTZ_CMD_RIGHT,///<right  IRC_NET_PTZ_CMD_LEFT_TOP,///<top Left  IRC_NET_PTZ_CMD_RIGHT_TOP,///<top Right  IRC_NET_PTZ_CMD_LIFETDOWN,///<bottom left  IRC_NET_PTZ_CMD_RIGHT_DOWN,///<bottom right  IRC_NET_PTZ_CMD_ZOOM_OUT,///<shortened focal length  IRC_NET_PTZ_CMD_ZOOM_IN,///<lengthened focal length  IRC_NET_PTZ_CMD_FOCUS_NEAR,///<focus closer  IRC_NET_PTZ_CMD_FOCUS_FAR,///<focus becomes farther away  IRC_NET_PTZ_CMD_IRIS_CLOSE,///<aperture reduction  IRC_NET_PTZ_CMD_IRIS_OPEN//<aperture increases  IRC_NET_PTZ_CMD_WIPER=1,///<wipers

```

IRC_NET_PTZ_CMD_LIGHT,///<fill light

IRC_NET_PTZ_CMD_DEFROST,///<defrost

IRC_NET_PTZ_CMD_DEFOG,///<clear fog

IRC_NET_PTZ_CMD_FAN,///<fan

IRC_NET_PTZ_CMD_HEATER,///<heater

IRC_NET_PTZ_CMD_AUTOMATIC,///<auto focus

IRC_NET_PTZ_CMD_SYNCHRONOUS_VIEW,///<bidirectiona
l follow_up

}

```

## 11. Type of preset point control function

### **IRC\_NET\_PTZ\_PRESET\_CMD\_TYPE**

Definition	Description
<pre> typedef enum  {     IRC_NET_PTZ_PRESET_CMD_ADD=0,///&lt;add preset point      IRC_NET_PTZ_PRESET_CMD_UPDATE_NAME,///&lt;update preset point names      IRC_NET_PTZ_PRESET_CMD_GOTO,///&lt;move to preset point      IRC_NET_PTZ_PRESET_CMD_DELETE,///&lt;delete a single preset point      IRC_NET_PTZ_PRESET_CMD_DELETE_ALL,///&lt;delete all </pre>	

```
preset points  
}
```

## 12.Cruise control function type

### **IRC\_NET\_PTZ\_TOUR\_CMD\_TYPE**

Definition	Description
<pre>typedef enum {      IRC_NET_PTZ_TOUR_CMD_SAVE=0,/// &lt;save Cruise Group      IRC_NET_PTZ_TOUR_CMD_START,/// &lt;start cruising      IRC_NET_PTZ_TOUR_CMD_STOP,/// &lt;stop Cruise      IRC_NET_PTZ_TOUR_CMD_DELETE,/// &lt;delete a single cruise group      IRC_NET_PTZ_TOUR_CMD_DELETE_ALL//&lt;delete all cruise control groups }</pre>	

## 13.Type of patrol control function

### **IRC\_NET\_PTZ\_PATTERN\_CMD\_TYPE**

Definition	Description
<pre>typedef enum {</pre>	

```

IRC_NET_PTZ_PATTERN_CMD_START_RECOED=0,///<start
recording

IRC_NET_PTZ_PATTERN_CMD_STOP_RECOED,///<end
Record

IRC_NET_PTZ_PATTERN_CMD_START_REPLY,///<start
replay

IRC_NET_PTZ_PATTERN_CMD_STOP_REPLY,///<end
replay

IRC_NET_PTZ_PATTERN_CMD_ELETE_ALL//<delete      all
patrols

}

```

## **14.SD card file type    IRC\_NET\_SD\_FILE\_TYPE**

Definition	Description
<pre> typedef enum {     IRC_NET_SD0SNAP_ALL=0,///&lt;capture all images     IRC.NET SD SNAP AlaRM,///&lt;alarm capture image     IRC_NET_SD0SNAP_COMMON,///&lt;normal captured images     IRC.NET SD VIDEO_ALL,///&lt;all recordings     IRC.NET SD VIDEO_ALARM,///&lt;alarm recording     IRC.NET SD_VIDEO_COMMON//&lt;regular Recording } </pre>	

```
}
```

## 15. Temperature measurement rule type

### **IRC\_NET\_TEMP\_RULE\_TYPE**

Definition Description
Temperature measurement rule type typedef struct { IRC_NET_TEMP_RULE_POINT = 0, /// IRC_NET_TEMP_RULE_LINE, /// IRC_NET_TEMP_RULE_RECT, /// IRC_NET_TEMP_RULE_CIRCLE, /// IRC_NET_TEMP_RULE_POLYGON /// };

## 16. Alarm action type **IRC\_NET\_ALARM\_ACTION**

Definition Description
Alarm action type typedef struct { IRC_NET_ALARM_ACTION_SINGLE = 0, /// IRC_NET_ALARM_ACTION_START, /// };

```
    IRC_NET_ALARM_ACTION_END ///< alarm end  
}  
}
```

## 17. Temperature unit **IRC\_NET\_TEMP\_UNIT**

Definition Description
Temperature unit  typedef struct  {  IRC_NET_TEMP_CENTIGRADE = 0, ///< Celsius  IRC_NET_TEMP_FAHRENHEIT, ///< Fahrenheit  IRC_NET_TEMP_KELVIN ///< Kelvin  }

## 18. Level temperature measurement range

### **IRC\_NET\_TEMP\_LEVEL\_TYPE**

Definition Description
level temperature measurement range  typedef struct  {  IRC_NET_TEMP_LEVEL_HG = 0, ///< High gain  IRC_NET_TEMP_LEVEL_LG, ///< Low gain  IRC_NET_TEMP_LEVEL_AUTO ///< Auto gain

```
}
```

## 19. Color Palette type IRC\_NET\_PALETTE\_TYPE

Definition Description
<pre>Color Palette type  typedef struct  {      IRC_NET_PALETTE_WHITE_HOT = 0, ///&lt;&lt; WhiteHot      IRC_NET_PALETTE_BLACK_HOT, ///&lt;&lt; BlackHot      IRC_NET_PALETTE_RAINBOW, ///&lt;&lt; Rainbow      IRC_NET_PALETTE_RAINBOW_HC, ///&lt;&lt; RainbowHC      IRC_NET_PALETTE_IRON, ///&lt;&lt; Iron      IRC_NET_PALETTE_LAVA, ///&lt;&lt; Lava      IRC_NET_PALETTE_SKY, ///&lt;&lt; Sky      IRC_NET_PALETTE_MID_GREY, ///&lt;&lt; MidGrey      IRC_NET_PALETTE_RDGY, ///&lt;&lt; RdGy      IRC_NET_PALETTE_PUOR, ///&lt;&lt; PuOr      IRC_NET_PALETTE_SPECIAL, ///&lt;&lt; Special      IRC_NET_PALETTE_RED, ///&lt;&lt; Red      IRC_NET_PALETTE_ICE_FIRE, ///&lt;&lt; IceFire      IRC_NET_PALETTE_GREE_RED, ///&lt;&lt; GreenRed      IRC_NET_PALETTE_SPECIAL2, ///&lt;&lt; Special2</pre>

```
IRC_NET_PALETTE_RED_HOT, ///< RedHot  
IRC_NET_PALETTE_GREEN_HOT, ///< GreenHot  
IRC_NET_PALETTE_BLUE_HOT, ///< BlueHot  
IRC_NET_PALETTE_GREEN, ///< Green  
IRC_NET_PALETTE_BLUE, ///Blue  
}
```

## 20. Action Type **IRC\_NET\_ACTION\_TYPE**

Definition Description
<pre>Action Type typedef enum {     IRC_NET_ACTION_AUTO_RESET = 0, ///&lt; Auto Homing     IRC_NET_ACTION_PRESET, ///&lt; Presets     IRC_NET_ACTION_TOUR, ///&lt; Tour     IRC_NET_ACTION_PATTERN, ///&lt; Pattern     IRC_NET_ACTION_LINE_SCAN, ///&lt; Line scan     IRC_NET_ACTION_AREA_SCAN, ///&lt; Area scan }</pre>

# Chapter 6 Demo Source Code & Operation Guide

## 1. Demo Source Code

### 1) Compiling Environment

Windows: win10, VS2019 + Qt5.15.2

Linux: ubuntu18.04, Qt5.15.2

### 2) QT Configuration Instructions

Reference: [Windows系统安装Qt 5.15.2完全版\\_qt5.15.2-CSDN博客](#)

### 3) Windows C++ Basic Runtime Library Description

The Microsoft runtime libraries are used during the Windows SDK process.

Download path:

### 4) Folder Content Description

	Source Files Location	Library Files Location	Header Files Location
Windows/Linux	../Demo/C++/src	.. /SDK/x64 .. /SDK/x86	.. /SDK/include

### 5) SDK Update Description

To update the SDK, you only need to update the infrared libraries and header files.

Windows:

Infrared libraries are IRCNetSDK.dll, IRCNetSDK.lib; other libraries are the dependencies used by the infrared library.

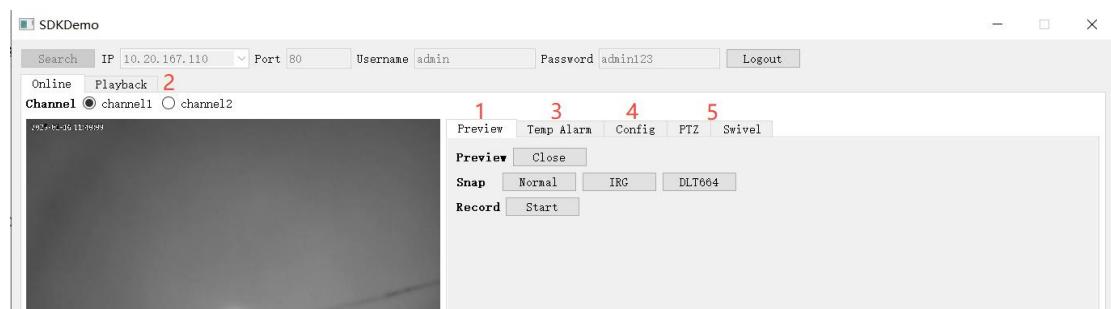
Header files are IRCNetSDK.h, IRCNetSDKDef.h.

Linux:

Infrared libraries are LibIRCNetSDK.so, LibIRCNetSDK.so.1, LibIRCNetSDK.so.1.0, LibIRCNetSDK.so.1.0.0; header files are IRCNetSDK.h, IRCNetSDKDef.h.

## 2. Demo Operation

- 1) Login username and password are the same as the device's web login credentials; after logging in to the device, further operations can be performed.
- 2) For devices that support both visual and infrared dual-channel images, You can switch the channel number to change the display and configure the corresponding channel.
- 3) The demo is divided into five modules: Preview, Playback, Temp Alarm, Configuration , and PTZ Control.



1 : Preview corresponds to the demonstration of some interfaces in the "Real-time Preview & Snapshot Recording" section of this manual, such as preview and snapshot interfaces.

2 : Playback corresponds to the SD card-related interfaces in the "Device Configuration" section of this manual.

3 : Temp Alarm corresponds to the demonstration of some interfaces in the "Temperature Measurement Configuration & Temperature Data Acquisition" section of this manual, such as full-frame temperature acquisition and full-frame/region high and low-temperature acquisition interfaces.

4 : Config corresponds to the demonstration of some interfaces in the "Device Configuration" section of this manual, such as OSD settings and IP settings interfaces.

5 : PTZ Control is for devices that support PTZ control and corresponds to the demonstration of some PTZ-related interfaces in the "Device Configuration" section of this manual, such as PTZ rotation and preset interfaces.

# Appendix

**Table 1: PTZ Commands**

Note: speed range:1-8; Blank table parameters do not need to be filled in.

CMD	param 1	param 2	param 3
IRC_NET_PTZ_CMD_WIPER			
IRC_NET_PTZ_CMD_LIGHT			
IRC_NET_PTZ_CMD_DEFROST			
IRC_NET_PTZ_CMD_DEFOG			
IRC_NET_PTZ_CMD_FAN			
IRC_NET_PTZ_CMD_HEATER			
IRC_NET_PTZ_CMD_AUTOMATIC			
IRC_NET_PTZ_CMD_SYNCHRONOUS_VIEW			
IRC_NET_PTZ_CMD_UP	speed		
IRC_NET_PTZ_CMD_DOWN	speed		
IRC_NET_PTZ_CMD_LEFT	speed		
IRC_NET_PTZ_CMD_RIGHT	speed		
IRC_NET_PTZ_CMD_LEFT_TOP	speed		
IRC_NET_PTZ_CMD_RIGHT_TOP	speed		
IRC_NET_PTZ_CMD_LEFT_DOWN	speed		
IRC_NET_PTZ_CMD_RIGHT_DOWN	speed		
IRC_NET_PTZ_CMD_ZOOM_OUT			
IRC_NET_PTZ_CMD_ZOOM_IN			
IRC_NET_PTZ_CMD_FOCUS_NEAR			
IRC_NET_PTZ_CMD_FOCUS_FAR			
IRC_NET_PTZ_CMD_IRIS_CLOSE			
IRC_NET_PTZ_CMD_IRIS_OPEN			

