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	1. The sdk supports PD4 and FC1; 2. Add device search interface IRC_NET_SearchDevice; 3. Add the function of recording SDK running logs and setting the level of saving logs; 4. Add the query device capability interface IRC_NET_GetDevAbility, which currently only supports querying the specific pan tilt capabilities of the device; 5. Add "preset id" argument for the “delete all

		temperature measurement area interface(IRC_NET_DeleteAllTempRule); 6. Add interface for querying, modifying the entire frame temperature measurement alarm configuration; 7. Add interfaces for querying the number of shielding area, querying shielding area detail information, adding\deleting\deleting all shielding area; 8. Add “channel id” argument for querying ,setting the time title”interface; 9. Add “channel id” argument for querying ,setting the channel title”interface; 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; 11. Add interfaces for querying SD card files and downloading files
V1.0.2	2024-7-26	1. 1.Added interfaces related to IP configuration. 2. 2.Added Temperature Stream V2 interface. 3. 3.Added Video Stream V2 interface. 4. 4.Added Pan-Tilt interface. 5. 5.PC5, FC2, TN220, PD2, SI4.
V1.0.3	2024-8-14	1. 1.Revised Supported Devices Description
V1.0.4	2024-8-17	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

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

		IRC_NET_SystemReboot”interface; 2. Added “Data Passthrough”related interfaces; 3. Added “Get Gimbal Position IRC_NET_GetCurrentPtz”interface; 4. Added “Area Scanning” related interfaces; 5. Added “Boot-up\Idle Actions”related interfaces; 6. Modify”Alarm Type IRC_NET_ALARM_TYPE”; 7. Added Chapter 6 “Demo Source Code & Operation Guide”.
V1.0.10	2025-3-20	1. Compatible with AT20, TN460U； 2. Added ranging related interfaces for PC5； 3. Added interfaces related to wiper and fill light mode settings; 4. Added “IRC_NET SystemRestart” interface; 5. Fix IRC_NET_GetCurrentPtz Zoom value acquisition anomaly.
V1.0.11	2025-7-04	1. Compatible with new PC4, PC6; 2. Support capability-set-based login for TN220, PC4, and PC6; 3. Live view：“Start live view_2” supports pushing target detection bounding box information; 4. Device configuration：added “day/night”, “IR brightness”，“IR contrast”，“IR flip”, “IR enhance”,“Web logo”related interfaces； 5. PTZ control：added “linkage tracking”，“precise tracking positioning”,“multiplier”, “acceleration”， “get PTZ position_V1” related interfaces； 6. Temperature Measurement Configuration：

		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; 7. Alarm Data Retrieval: added alarm image to the returned data, for fire(pulse) type alarm, added fire coordinate to the returned data.
V1.0.12	2025-12-01	1. Device connection Adjustment of the "Get device capability information" interface;

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

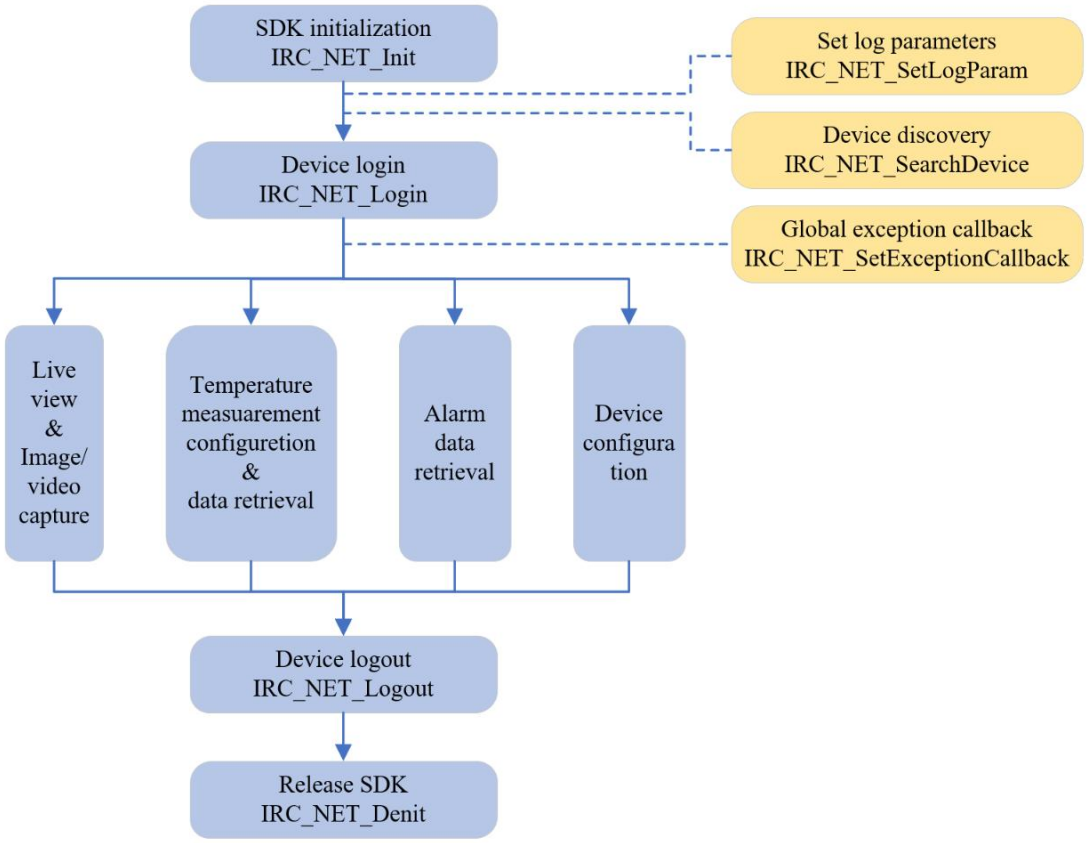
Windows 32/64bit Version: Windows 11/Windows 10/Windows 8/Windows 7
Linux 32/64bit Version: Ubuntu 18.04 and above
Arm Linux 32/64bit Version: Linux_x86_64，GCC 7.5.0 and above Arrch64_linux_gnu_7.3.1 and above，GCC 7.3.1 and above

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

2. If multiple devices need to be accessed, separate logins are required.

Chapter 3 Interface List

1. Device Connection

1) SDK Initialization IRC_NET_Init

Options	Introduction
Description	SDK initialization
Function	Int IRC_NET_Init()
Parameter	
Return Value	(Refer to the status code table for details.)
Note	

2) Release SDK IRC_NET_Deinit

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

3) Set log parameters IRC_NET_SetLogParam

Options	Introduction

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

4) Device discovery IRC_NET_SearchDevice

Options	Introduction
Description	Device discovery

Function	int IRC_NET_SearchDevice(int timeout, IRC_NET_DEV_SEARCH_INFO searchInfos[], int inSize, int* outSize);
Parameter	Param[in] timeout query timeout, unit: ms Param[out] infos[] device search information Param[in] inSize device search information input size Param[out] outSize Device search information output size
Return Value	status code
Note	

5) Device login IRC_NET_Login

Options	Introduction
Description	Device login
Function	int IRC_NET_Login(Const IRC_NET_LOGIN_INFO* loginInfo, IRC_NET_HANDLE* handle);
Parameter	param[in] loginInfo login information

	param[out] handle operation handle
Return Value	status code
Note	

6) Device logout IRC_NET_Logout

Options	Introduction
Description	Device logout
Function	int CALL_METHOD IRC_NET_Logout(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

7) Get basic device information IRC_NET_GetDevInfo

Options	Introduction
Description	Get basic device information
Function	int IRC_NET_GetDevInfo(IRC_NET_HANDLE handle, IRC_NET_DEV_INFO* devInfo);
Parameter	param[in] handle operation handle param[out] devInfo device information
Return Value	status code

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, int channel, int type, void* ability);
Parameter	param[in] handle operation handle param[in] channel channel No. param[in] type capability type, refer to IRC_NET_DEV_ABILITY_TYPE param[out] ability Capability information shall be parsed according to the capability type; refer to the demo for details.
Return Value	
Note	

2. Live View&Image/Video Capture

1) Start live view IRC_NET_StartPreview

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

2) Video callback function IRC_NET_VIDEO_CALLBACK

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

3) Start live view_V2 IRC_NET_StartPreview_V2

Options	Introduction

Description	Start live view
Function	int IRC_NET_StartPreview_V2(IRC_NET_HANDLE handle, const IRC_NET_PREVIEW_INFO* previewInfo, IRC_NET_VIDEO_CALLBACK_V2 videoCb, void* userData);
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	status code
Note	

4) Video callback function_V2

IRC_NET_VIDEO_CALLBACK_V2

Options	Introduction
Description	Video callback function_V2
Function	void* IRC_NET_VIDEO_CALLBACK_V2(IRC_NET_HANDLE handle,

	IRC_NET_VIDEO_INFO_CB* videoInfo, IRC_NET_IVS_INFO_CB* ivsInfo, void* userData);
Parameter	param[in] handle operation handle param[out] videoInfo video callback param[out] ivsInfo video callback intelligent info param[out] userData User-defined data
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(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

6) Live view snapshot IRC_NET_PreviewSnapshot

Options	Introduction

Description	Live view snapshot
Function	Int IRC_NET_PreviewSnapshot(IRC_NET_HANDLE handle, const char* filePath);
Parameter	param[in] handle operation handle param[in] filePath File storage path: [Save file path] + [File name] + .jpg
Return Value	status code
Note	

7) Start live view recording IRC_NET_StartPreviewRecord

Options	Introduction
Description	Start live view recording, MP4 format, without temperature data
Function	int IRC_NET_StartPreviewRecord(IRC_NET_HANDLE handle, const char* filePath);
Parameter	param[in] handle operation handle param[in] filePath File storage path: [Save file path] + [File name] + .mp4
Return Value	status code
Note	

8) Stop live view recording IRC_NET_StopPreviewRecord

Options	Introduction
Description	Stop live view recording
Function	Int IRC_NET_StopPreviewRecord(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

3. Temperature Measurement Configuration & Retrieval of Temperature Data

3.1 Retrieval of Temperature Data

1) Temperature environmental parameter correction IRC_NET_SetPullTempExecParam

Options	Introduction
Description	Temperature environmental parameter correction
Function	int IRC_NET_SetPullTempExecParam(IRC_NET_HANDLE handle, int envCorrectionEnable);
Parameter	param[in] handle operation handle

	param[in] envCorrectionEnable Environmental parameter correction switch，0-close，1-open
Return Value	status code
Note	When enabled, the temperature data returned by the temperature callback function (_V2) is Kelvin × 10; when disabled, the original temperature data is returned; if not executed, Kelvin × 10 is returned.

2) Start pulling real-time temperature stream

IRC_NET_StartPullTemp

Options	Introduction
Description	Start pulling real-time temperature stream
Function	int IRC_NET_StartPullTemp(IRC_NET_HANDLE handle, IRC_NET_TEMP_CALLBACK tempCb, void* userData);
Parameter	param[in] handle operation handle param[in] tempCb temperature callback function param[in] userData user self-defined data
Return Value	status code
Note	

3) Temperature callback function

IRC_NET_TEMP_CALLBACK

Options	Introduction
Description	Temperature callback function
Function	void* IRC_NET_TEMP_CALLBACK(IRC_NET_HANDLE handle, char* temp, int width, int height, void* userData);
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

4) Start pulling real-time temperature stream_V2

IRC_NET_StartPullTemp_V2

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

5) Temperature callback function_V2

IRC_NET_TEMP_CALLBACK_V2

Options	Introduction
Description	Temperature callback function_V2
Function	void* IRC_NET_TEMP_CALLBACK_V2(IRC_NET_HANDLE handle, IRC_NET_TEMP_INFO_CB* tempInfo,

	IRC_NET_TEMP_EXT_INFO_CB* extInfo, void* userData);
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

6) 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	status code

Note	

7) Start pulling real-time raw stream

IRC_NET_StartPullNucData

Options	Introduction
Description	Start pulling real-time raw stream
Function	int IRC_NET_StartPullNucData(IRC_NET_HANDLE handle, IRC_NET_NUC_DATA_CALLBACK dataCb, void* userData);
Parameter	param[in] handle operation handle param[in] dataCb raw callback temperature info param[in] userData user-defined data
Return Value	status code
Note	

8) Raw stream callback function IRC_NET_StopPullTemp

Options	Introduction
Description	Raw stream callback function
Function	void* IRC_NET_NUC_DATA_CALLBACK(IRC_NET_HANDLE handle, char* data,

	int width, int height, void* userData);
Parameter	param[in] handle operation handle param[out] data raw stream param[out] width width of raw stream param[out] height height of raw stream param[out] userData user-defined data
Return Value	status code
Note	Raw data is 14bit

9) Stop pulling real-time raw stream

IRC_NET_StopPullNucData

Options	Introduction
Description	Stop pulling real-time raw stream
Function	int IRC_NET_StopPullNucData(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

3.2 Basic parameters

1) Get temperature measurement enable status

IRC_NET_GetTempEnable

Options	Introduction
Description	Get temperature measurement enable status
Function	int IRC_NET_GetTempEnable(IRC_NET_HANDLE handle, int* enable);
Parameter	param[in] handle operation handle param[out] enable temp switch，0-close，1-open
Return Value	Status code
Note	

2) Set temperature measurement enable status

IRC_NET_SetTempEnable

Options	Introduction
Description	Set temperature measurement enable status
Function	int IRC_NET_SetTempEnable(IRC_NET_HANDLE handle, int enable);
Parameter	param[in] handle operation handle

	param[in] enable temp switch，0-close， 1-open
Return Value	Status code
Note	

3) Get temperature unit IRC_NET_GetTempUnit

Options	Introduction
Description	Get temperature unit
Function	int IRC_NET_GetTempUnit(IRC_NET_HANDLE handle, int* unit);
Parameter	param[in] handle operation handle param[out] unit temp unit, 0: Celsius, 1: Fahrenheit, 2: Kelvin
Return Value	Status code
Note	

4) Set temperature unit IRC_NET_SetTempUnit

Options	Introduction
Description	Set temperature unit
Function	int IRC_NET_SetTempUnit(IRC_NET_HANDLE handle, int unit);

Parameter	param[in] handle operation handle param[in] unit temp unit, 0: Celsius, 1: Fahrenheit, 2: Kelvin
Return Value	Status code
Note	

5) Get temperature measurement frame rate

IRC_NET_GetFrameRate

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

6) Set temperature measurement frame rate

IRC_NET_SetFrameRate

Options	Introduction
Description	Set temperature measurement frame rate
Function	int IRC_NET_SetFrameRate(IRC_NET_HANDLE handle, int rate);
Parameter	param[in] handle operation handle param[in] rate temperature measurement frame rate，range[1，frame rate]，12 by default, support 25 frames at most
Return Value	status code
Note	

7) Get temperature measurement range

IRC_NET_GetTempLevel

Options	Introduction
Description	Get temperature measurement range
Function	int IRC_NET_GetTempLevel(IRC_NET_HANDLE handle, int* level);

Parameter	param[in] handle operation handle param[out] level temperature measurement range, refer to IRC_NET_TEMP_LEVEL_TYPE
Return Value	status code
Note	

8) Set temperature measurement range

IRC_NET_SetTempLevel

Options	Introduction
Description	Set temperature measurement range
Function	int IRC_NET_SetTempLevel(IRC_NET_HANDLE handle, int level);
Parameter	param[in] handle operation handle param[in] level temperature measurement range, refer to IRC_NET_TEMP_LEVEL_TYPE
Return Value	status code
Note	

9) Get environmental parameters of the full frame

IRC_NET_GetEnvParam

Options	Introduction

Description	Get environmental parameters of the full frame
Function	int IRC_NET_GetEnvParam(IRC_NET_HANDLE handle, IRC_NET_ENV_PARAM* envParam);
Parameter	param[in] handle operation handle param[out] envParam environmental parameters
Return Value	status code
Note	

10) Set environmental parameters of the full frame

IRC_NET_SetEnvParam

Options	Introduction
Description	Set environmental parameters of the full frame
Function	int IRC_NET_SetEnvParam(IRC_NET_HANDLE handle, const IRC_NET_ENV_PARAM* envParam);
Parameter	param[in] handle operation handle param[in] envParam environmental parameters
Return Value	status code
Note	

3.3 Temperature measurement rules

1) Get the number of interested temperature measurement rules

IRC_NET_QueryTempRuleSize

Options	Introduction
Description	Get the number of drawn temperature measurement rules
Function	int IRC_NET_QueryTempRuleSize(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INDEX* tempRuleIndex, int* size);
Parameter	param[in] handle operation handle param[in] tempRuleIndex rule index 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 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] size Query the count
Return Value	status code
Note	

2) 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	int IRC_NET_QueryTempRule(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INDEX* tempRuleIndex, IRC_NET_TEMP_RULE_INFO tempRuleInfos[], int inSize, int* outSize);
Parameter	param[in] handle operation handle param[in] tempRuleIndex Temperature measurement rule index 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 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	status code

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 IRC_NET_QueryTempRuleSize

3) Get detailed information for an insterested temperature measurement rule_G1 IRC_NET_QueryTempRule_G1

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

	param[out] tempRuleInfos[] Temperature measurement rule info param[in] inSize temperature measurement rule input size param[out] outSize temperature measurement rule output size
Return Value	status code
Note	Compatible with ATR series only

4) 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 IRC_NET_TEMP_RULE_INFO* tempRegionRuleInfo);
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo temperature measurement rule information

Return Value	status code
Note	

5) 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 IRC_NET_TEMP_RULE_INFO_G1* tempRegionRuleInfo);
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo temperature measurement rule information
Return Value	status code
Note	Compatible with ATR series only

6) Modify an insterested temperature measurement rule

IRC_NET_UpdateTempRule

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

7) Modify an insterested temperature measurement rule_g1

irc_net_updatetemprule_G1

Options	Introduction
Description	Modify a specific temperature measurement rule_G1
Function	int IRC_NET_UpdateTempRule_G1(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INFO_G1*

	tempRegionRuleInfo);
Parameter	param[in] handle operation handle param[in] tempRegionRuleInfo Temperature measurement rule info
Return Value	status code
Note	Compatible with ATR series only

8) Delete a single temperature measurement rule

IRC_NET_DeleteTempRule

Options	Introduction
Description	Delete a single temperature measurement rule
Function	int IRC_NET_DeleteTempRule(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INDEX* tempRuleIndex);
Parameter	param[in] handle operation handle param[in] tempRuleIndex temperature measurement rule index
Return Value	status code

Note	

9) Delete all temperature measurement rules

IRC_NET_DeleteAllTempRule

Options	Introduction
Description	Delete all temperature measurement rules
Function	int IRC_NET_DeleteAllTempRule(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

10) 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	int IRC_NET_QueryRuleTempSize(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INDEX* tempRuleIndex, int* size);

Parameter	param[in] handle operation handle param[in] tempRuleIndex rule index 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. param[out] size Query the count
Return Value	status code
Note	

11) Get rules high-low temperature information

IRC_NET_QueryRuleTemp

Options	Introduction
Description	Get rules high-low temperature information
Function	int IRC_NET_QueryRuleTemp(IRC_NET_HANDLE handle, const IRC_NET_TEMP_RULE_INDEX* tempRuleIndex, IRC_NET_RULE_TEMP_INFO ruleTempInfos[], int inSize,

	int* outSize);
Parameter	param[in] handle operation handle param[in] tempRuleIndex temperature measurement rule index 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. param[out] ruleTempInfos[] rule temperature information 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. param[in] inSize Rule temperature input size When retrieving temperature information for a specific ID, this field is set to 1. Otherwise, its value should be the return value of the IRC_NET_QueryRuleTempSize function.

	param[out] outSize Rule temperature output size
Return Value	status code
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"

12) 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, IRC_NET_TEMP_INFO* tempInfo);
Parameter	param[in] handle operation handle param[out] ruleTempInfos[] rule temperature information
Return Value	status code
Note	

13) Get the number of temperature measurement shielding areas

IRC_NET_QueryTempMaskSize

Options	Introduction
Description	Get the number of temperature measurement shielding areas
Function	int IRC_NET_QueryTempMaskSize(IRC_NET_HANDLE handle, const IRC_NET_TEMP_MASK_INDEX* tempMaskIndex, int* size);
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	status code
Note	

14) 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	int IRC_NET_QueryTempMask(IRC_NET_HANDLE handle, const IRC_NET_TEMP_MASK_INDEX* tempMaskIndex, IRC_NET_TEMP_MASK_INFO tempMaskInfos[], int inSize, int* outSize);
Parameter	Param [in] handle operation handle Param [in] tempMaskIndex Mask area index, presetId must be greater than or equal to 0 Param tempMaskInfos temperature measurement shielding area information Param [in] inSize Input size of temperature measurement shielding area Param [out] outSize Output size of temperature measurement shielding area

Return Value	status code
Note	

15) Add temperature measurement shielding area

IRC_NET_AddTempMask

Options	Introduction
Description	Add temperature measurement shielding area
Function	int IRC_NET_AddTempMask(IRC_NET_HANDLE handle, const IRC_NET_TEMP_MASK_INFO* tempMaskInfo);
Parameter	Param [in] handle operation handle Param [in] tempMaskInfo Temperature measurement shielding area information
Return Value	status code
Note	

16) Delete temperature measurement shielding area

IRC_NET_DeleteTempMask

Options	Introduction
Description	Delete temperature measurement shielding area.
Function	int IRC_NET_DeleteTempMask(

	IRC_NET_HANDLE handle, const IRC_NET_TEMP_MASK_INDEX* tempMaskIndex);
Parameter	Param [in] handle operation handle Param [in] tempMaskIndex Mask Area Index
Return Value	status code
Note	

17) Delete all temperature measurement shielding areas

IRC_NET_DeleteAllTempMask

Options	Introduction
Description	Delete all temperature measurement shielding areas.
Function	int IRC_NET_DeleteAllTempMask(IRC_NET_HANDLE handle, int presetId);
Parameter	Param [in] handle operation handle Param [in] presetId
Return Value	status code
Note	

18) Get random point temperature

IRC_NET_QueryRandomTemp

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

3.4 Temperature measurement tool

1) Get temperature span information

IRC_NET_GetTempSpanInfo

Options	Introduction
Description	Get temperature span information
Function	int IRC_NET_GetTempSpanInfo(IRC_NET_HANDLE handle,

	IRC_NET_TEMP_SPAN_INFO* tempSpanInfo);
Parameter	param[in] handle operation handle param[out] tempSpanInfo temperature span information
Return Value	status code
Note	

2) Set temperature span information

IRC_NET_SetTempSpanInfo

Options	Introduction
Description	Set temperature span information
Function	int IRC_NET_SetTempSpanInfo(IRC_NET_HANDLE handle, const IRC_NET_TEMP_SPAN_INFO* tempSpanInfo);
Parameter	param[in] handle operation handle param[in] tempSpanInfo temperature span information
Return Value	status code
Note	

3) Get current color pallette 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 IRC_NET_PALETTE_TYPE
Return Value	status code
Note	

4) Set color palette index IRC_NET_SetPalleteType

Options	Introduction
Description	Set color palette index
Function	int IRC_NET_SetPalleteType(IRC_NET_HANDLE handle, int palleteType);
Parameter	param[in] handle operation handle param[in] palleteType color palette type，refer to

	IRC_NET_PALETTE_TYPE
Return Value	status code
Note	

5) Get thermal images in DLT664 format

IRC_NET_GetDlt664Image

Options	Introduction
Description	Get thermal images in DLT664 format
Function	int IRC_NET_GetDLT664Image(IRC_NET_HANDLE handle, const char* filePath);
Parameter	param[in] handle operatio handle param[in] filePath file saving path+file name+.jpg
Return Value	status code
Note	

6) Get thermal images in irg format IRC_NET_GetIRGImage

Options	Introduction
Description	Get thermal images in IRG format
Function	int IRC_NET_GetIRGImage(IRC_NET_HANDLE handle, const char* irgFilePath,

	const char* jpgFilePath);
Parameter	param[in] handle operation handle param[in] irgFilePath file saving path+file name+.irg param[in] jpgFilePath file saving path+file name+.jpg
Return Value	status code
Note	

3.5 Template temperature measurement

1) Get temperature measurement template

IRC_NET_GetTempTemplate

Options	Introduction
Description	Get temperature measurement template
Function	int IRC_NET_GetTempTemplate(IRC_NET_HANDLE handle, int tempTemplate[], int inSize, int* outSize);
Parameter	param[in] handle operation handle param[out] tempTemplate[] Temperature

	measurement template array param[in] inSize Input array size of temperature measurement template, must be ≤ device resolution param[out] outSize Output array size of temperature
Return Value	status code
Note	

2) Set temperature measurement template

IRC_NET_SetTempTemplate

Options	Introduction
Description	Set temperature measurement template
Function	int IRC_NET_SetTempTemplate(IRC_NET_HANDLE handle, const int tempTemplate[], int inSize);
Parameter	param[in] handle operation handle param[in] tempTemplate[] Temperature measurement template array, data elements are: 1-set corresponding pixel, 0-do not set corresponding pixel param[in] inSize Temperature measurement template array size, must be ≤ device resolution, remaining bits are automatically filled with 0

Return Value	status code
Note	

3) Get temperature measurement template size

IRC_NET_GetTempTemplateSize

Options	Introduction
Description	Get temperature measurement template size
Function	int IRC_NET_GetTempTemplateSize(IRC_NET_HANDLE handle, int* outSize);
Parameter	param[in] handle operation handle param[out] outSize Temperature measurement template array size
Return Value	status code
Note	

4) Get temperature measurement template temperature

IRC_NET_GetTempTemplateValue

Options	Introduction
Description	Get temperature measurement template temperature
Function	int IRC_NET_GetTempTemplateValue(IRC_NET_HANDLE handle,

	int tempTemplateValue[], int inSize);
Parameter	param[in] handle Operation handle param[out] tempTemplateValue[] Temperature measurement template temperature array param[in] inSize Temperature measurement template temperature array size, must be ≤ device resolution
Return Value	status code
Note	

4.Alarm Data Retrieval

4.1 Alarm configuration

1) 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, IRC_NET_TEMP_FRAME_ALARM_CONFIG* alarmConfig);

Parameter	Param [in] handle operation handle Param [out] alarm configuration for whole frame temperature measurement alarm
Return Value	status code
Note	

2) 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, IRC_NET_FRAME_TEMP_ALARM_CONFIG_G1 * alarmConfig);
Parameter	param[in] handle operation handle param[out] alarmConfig full-frame temperature measurement alarm configuration
Return Value	status code
Note	

3) Modify the entire frame temperature measurement alarm

IRC_NET_UpdateFrameTempAlarmConfig

Options	Introduction
Description	Modify the entire frame temperature measurement alarm
Function	Int IRC_NET_UpdateFrameTempAlarmConfig(IRC_NET_HANDLE handle, const IRC_NET_TEMP_FRAME_ALARM_CONFIG* alarmConfig);
Parameter	Param [in] handle operation handle Param [out] alarm configuration for whole frame temperature measurement alarm
Return Value	status code
Note	

4) Modify full-frame temperature measurement alarm configuration_G1

IRC_NET_UpdateFrameTempAlarmConfig_G1

Options	Introduction
Description	Modify full-frame temperature measurement alarm

	configuration
Function	Int IRC_NET_UpdateFrameTempAlarmConfig(IRC_NET_HANDLE handle, const IRC_NET_TEMP_FRAME_ALARM_CONFIG* alarmConfig);
Parameter	param[in] handle operation handle param[out] alarmConfig full-frame temperature measurement alarm configuration
Return Value	status code

5) 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, IRC_NET_IP_TARGET_RECOGNITION_CONFIG * targetRecognitionConfig)
Parameter	param[in] handle operation handle param[in] channel channel No. param[out] targetRecognitionConfig channel target recognition configuration

Return Value	status code
Note	

6) 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 IRC_NET_IP_TARGET_RECOGNITION_CONFIG * targetRecognitionConfig)
Parameter	param[in] handle operation handle param[in] channel channle No. param[in] targetRecognitionConfig channel target recognition configuration
Return Value	status code
Note	

4.2 Alarm subscription

1) Global exception callback IRC_NET_SetExceptionCallback

Options	Introduction

Description	Exception callback registration
Function	int IRC_NET_SetExceptionCallback(IRC_NET_HANDLE handle, IRC_NET_EXCEPTION_CALLBACK exceptionCb, void* userData);
Parameter	param[in] handle operatio handle param[in] exceptionCb exception callback function param[in] userData user self-defined data
Return Value	status code
Note	

2) Exception callback function

IRC_NET_EXCEPTION_CALLBACK

Options	Introduction
Description	exception callback function, push exception information after registration
Function	void IRC_NET_EXCEPTION_CALLBACK(IRC_NET_HANDLE handle, int exceptionType, void* userData);
Parameter	param[in] handle operatio handle

	param[out] exceptionType exception type, refer to IRC_NET_EXCEPTION_TYPE 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	int IRC_NET_SubscribeAlarm(IRC_NET_HANDLE handle, IRC_NET_ALARM_CALLBACK alarmCb, void* userData);
Parameter	param[in] handle operation handle param[in] alarmCb alarm callback function param[in] userData user self-defined data
Return Value	status code
Note	

4) Subscribe to Alarms_V2 IRC_NET_SubscribeAlarm_V2

Options	Introduction
Description	Subscribe to Alarms_V2
Function	int IRC_NET_SubscribeAlarm_V2(IRC_NET_HANDLE handle, bool snapEnable, IRC_NET_ALARM_CALLBACK alarmCb, void* userData);
Parameter	param[in] handle operation handle param[in] snapEnable Whether to capture alarm images, 0-disabled, 1-enabled param[in] alarmCb Alarm callback function param[in] userData user self-defined data
Return Value	status code
Note	

5) Alarm callback function IRC_NET_ALARM_CALLBACK

Options	Introduction
Description	Alarm callback function,push alarm information after registration
Function	void IRC_NET_ALARM_CALLBACK(

	IRC_NET_HANDLE handle, int alarmType, void* alarmInfo, void* userData);
Parameter	param[in] handle operation handle param[out] alarmType alarm type，refer to IRC_NET_ALARM_TYPE 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.

6) 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	status code
Note	

5.Device Configuration

5.1 Camera Settings

1) Shutter correction IRC_NET_CorrectShutter

Options	Introduction
Description	Implementing shutter calibration function
Function	int IRC_NET_CorrectShutter(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

2) 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	int IRC_NET_GetTempBarState(IRC_NET_HANDLE handle, int* state);
Parameter	param[in] handle operation handle

	param[out] state temperature bar status，0: off　1:on
Return Value	status code
Note	

3) 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	int IRC_NET_SetTempBarState(IRC_NET_HANDLE handle, int state);
Parameter	param[in] handle operation handle param[in] state temperature bar status，0: off　1:on
Return Value	status code
Note	

4) Get IR contrast IRC_NET_GetThermalImageContrast

Options	Introduction
Description	Get IR contrast
Function	int IRC_NET_GetThermalImageContrast(IRC_NET_HANDLE handle,

	int* contrast);
Parameter	param[in] handle operation handle param[out] contrast contrast，0-100
Return Value	status code
Note	

5) Set IR contrast IRC_NET_SetThermalImageContrast

Options	Introduction
Description	Set IR contrast
Function	int IRC_NET_SetThermalImageContrast(IRC_NET_HANDLE handle, int contrast);
Parameter	param[in] handle operation handle param[in] contrast contrast，0-100
Return Value	status code
Note	

6) Get IR brightness IRC_NET_GetThermalImageLuminance

Options	Introduction
Description	Get IR brightness
Function	int IRC_NET_GetThermalImageLuminance(IRC_NET_HANDLE handle,

	int* luminance);
Parameter	param[in] handle operation handle param[out] luminance brightness，0-100
Return Value	status code
Note	

7) Set IR brightness IRC_NET_SetThermalImageLuminance

Options	Introduction
Description	Set IR brightness
Function	int IRC_NET_SetThermalImageLuminance(IRC_NET_HANDLE handle, int luminance);
Parameter	param[in] handle operation handle param[in] luminance brightness，0-100
Return Value	status code
Note	

8) Get IR image flip IRC_NET_GetThermalImageFlipMode

Options	Introduction
Description	Get IR image flip
Function	int IRC_NET_GetThermalImageFlipMode(IRC_NET_HANDLE handle,

	int* flipMode);
Parameter	param[in] handle operation handle param[out] flipMode flip mode (0:normal, 1:horizontal flip, 2:vertical flip, 3:180° flip)
Return Value	status code
Note	

9) Set IR image flipIRC_NET_SetThermalImageFlipMode

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

10) Get IR image enhance param

IRC_NET_GetThermalImageEnhanceInfo

Options	Introduction

Description	Get IR image enhance param
Function	int IRC_NET_GetThermalImageEnhanceInfo(IRC_NET_HANDLE handle, IRC_NET_THERMAL_IMAGE_MODE_ENHANCE_INFO* thermalImageEnhanceInfo);
Parameter	param[in] handle operation handle param[out] thermalImageEnhanceInfo IR image enhance param
Return Value	status code
Note	

11) 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, IRC_NET_THERMAL_IMAGE_MODE_ENHANCE_INFO* thermalImageEnhanceInfo);
Parameter	param[in] handle operation handle param[in] thermalImageEnhanceInfo IR image enhance param

Return Value	status code
Note	

12) Get visible day night mode param

IRC_NET_GetDayNightModeParam

Options	Introduction
Description	Get visible day night mode param
Function	int IRC_NET_GetDayNightModeParam(IRC_NET_HANDLE handle, IRC_NET_DAY_NIGHT_MODE_PARAM* dayNightModeParam);
Parameter	param[in] handle operation handle param[out] dayNightModeParam visible day night mode param
Return Value	status code
Note	

13) Set visible day night mode param

IRC_NET_SetDayNightModeParam

Options	Introduction
Description	Set visible day night mode param
Function	int IRC_NET_SetDayNightModeParam(

	IRC_NET_HANDLE handle, const IRC_NET_DAY_NIGHT_MODE_PARAM* dayNightModeParam);
Parameter	param[in] handle operation handle param[in] dayNightModeParam visible day night mode param
Return Value	status code
Note	

5.2 OSD Settings

1) Get OSD (on-screen display) overall status

IRC_NET_GetOSDState

Options	Introduction
Description	Get OSD overall status
Function	int IRC_NET_GetOSDState(IRC_NET_HANDLE handle, int* osdMode);
Parameter	param[in] handle operation handle param[out] osdMode temperature measurement information OSD mode; 0 on，2 off

Return Value	status code
Note	

2) 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	status code
Note	

3) Get time title IRC_NET_GetOSDTimeTitleInfo

Options	Introduction
Description	Get time title
Function	int IRC_NET_GetOSDTimeTitleInfo(IRC_NET_HANDLE handle,

	IRC_NET_OSD_TIME_TITLE_INFO* timeTitleInfo);
Parameter	param[in] handle operation handle param[out] timeTitleInfo time title information
Return Value	status code
Note	

4) Set time title IRC_NET_SetOSDTimeTitleInfo

Options	Introduction
Description	Set time title
Function	int IRC_NET_SetOSDTimeTitleInfo(IRC_NET_HANDLE handle, const IRC_NET_OSD_TIME_TITLE_INFO* timeTitleInfo);
Parameter	param[in] handle operation handle param[in] timeTitleInfo time title information
Return Value	status code
Note	

5) Get channel title IRC_NET_GetOSDChannelTitleInfo

Options	Introduction
Description	Get channel title

Function	int IRC_NET_GetOSDChannelTitleInfo(IRC_NET_HANDLE handle, IRC_NET_OSD_CHANNEL_TITLE_INFO* channelTitleInfo);
Parameter	param[in] handle operation handle param[out] channelTitleInfo channel title information
Return Value	status code
Note	

6) Set channel title IRC_NET_SetOSDChannelTitleInfo

Options	Introduction
Description	Set channel title
Function	int IRC_NET_SetOSDChannelTitleInfo(IRC_NET_HANDLE handle, const IRC_NET_OSD_CHANNEL_TITLE_INFO* channelTitleInfo);
Parameter	param[in] handle operation handle param[in] channelTitleInfo channel title information
Return Value	status code
Note	

7) Get laser distance OSD parameters

IRC_NET_GetLaserDistanceOsdParam

Options	Introduction
Description	Get Laser Distance OSD parameters
Function	int IRC_NET_GetLaserDistanceOsdParam(IRC_NET_HANDLE handle, int channel, IRC_NET_LASER_DISTANCE_OSD_PARAM* laserDistanceOsdParam);
Parameter	param[in] handle operation handle param[in] channel channel param[out] laserDistanceOsdParam Laser Distance OSD parameters
Return Value	status code
Note	

8) Set laser distance OSD parameters

IRC_NET_SetLaserDistanceOsdParam

Options	Introduction
Description	Set Laser Distance OSD parameters
Function	int IRC_NET_SetLaserDistanceOsdParam(

	IRC_NET_HANDLE handle, int channel, const IRC_NET_LASER_DISTANCE_OSD_PARAM* laserDistanceOsdParam);
Parameter	param[in] handle operation handle param[in] channel channel param[in] laserDistanceOsdParam Laser Distance OSD parameters
Return Value	status code
Note	

5.3 Network transparent transmission

1) Set serial port passthrough state

IRC_NET_SetTransparentState

Options	Introduction
Description	Set serial port passthrough state
Function	int IRC_NET_SetTransparentState(IRC_NET_HANDLE handle, int state)
Parameter	param[in] handle operation handle

	param[in] state Passthrough state，0：close 1：open
Return Value	status code
Note	

2) Data serial port passthrough IRC_NET_TransparentData

Options	Introduction
Description	Data serial port passthrough
Function	int IRC_NET_TransparentData(IRC_NET_HANDLE handle, const char* sendBuf, int sendBufSize, char* recvBuf, int recvBufInSize, int* recvBufOutSize, int timeout)
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	status code
Note	need to use the 'IRC_NET_SetTransparentState' function to enable transparent transmission

5.4 Storage Management

1) 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 IRC_NET_FILE_QUERY_PARAM* queryParam, int* size);
Parameter	param[in] handle operation handle param[in] condition Query conditions param[out] size Query quantity
Return Value	status code
Note	

2) Get SD card files IRC_NET_QueryFile

Options	Introduction
Description	Get SD card files
Function	int IRC_NET_QueryFile(IRC_NET_HANDLE handle, const IRC_NET_FILE_QUERY_PARAM* queryParam, IRC_NET_FILE_INFO fileInfo[], int inSize, int* outSize);
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	status code
Note	

3) Download SD card files IRC_NET_StartDownloadFile

Options	Introduction
Description	Download files on SD card

Function	int IRC_NET_StartDownloadFile(IRC_NET_HANDLE handle, const IRC_NET_FILE_DOWNLOAD_INFO* downloadInfo);
Parameter	param[in] handle operation handle param[in] downloadInfo download info
Return Value	status code
Note	

4) Get download progress IRC_NET_GetDownloadProgress

Options	Introduction
Description	Get download progress
Function	int IRC_NET_GetDownloadProgress(IRC_NET_HANDLE handle, int fileHandle, IRC_NET_FILE_DOWNLOAD_PROGRESS* downloadProgress);
Parameter	param[in] handle operation handle param[in] fileHandel 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	

5) 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] fileHandel down handle, IRC_NET_StartDownloadFile return value
Return Value	status code
Note	

5.5 Peripheral Settings

1) 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	status code
Note	

2) Set wiper configuration IRC_NET_SetWiperConfigInfo

Options	Introduction
Description	Set wiper configuration information
Function	int IRC_NET_SetWiperConfigInfo(IRC_NET_HANDLE handle, int wiperMode);
Parameter	param[in] handle operation handle param[in] wiperMode wiper mode 0 Auto， 1 Manual
Return Value	status code
Note	

3) Get fill light configuration IRC_NET_GetFillLightConfigInfo

Options	Introduction
Description	Get fill light configuration

Function	int IRC_NET_GetFillLightConfigInfo(IRC_NET_HANDLE handle, IRC_NET_FILL_LIGHT_CONFIG_INFO* fillLightConfigInfo);
Parameter	param[in] handle operation handle param[out] fillLightConfigInfo fill light configuration
Return Value	status code
Note	

4) Set fill light configuration IRC_NET_SetFillLightConfigInfo

Options	Introduction
Description	Data Passthrough
Function	int IRC_NET_SetFillLightConfigInfo(IRC_NET_HANDLE handle, const IRC_NET_FILL_LIGHT_CONFIG_INFO* fillLightConfigInfo);
Parameter	param[in] handle operation handle param[in] fillLightConfigInfo fill light configuration
Return Value	status code
Note	

5.6 System Management

1) Synchronize system time IRC_NET_SyncSystemTime

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

2) Get IP configuration IRC_NET_GetIpConfig

Options	Introduction
Description	Get IP configuration
Function	IRC_NET_GetIpConfig(IRC_NET_HANDLE handle, IRC_NET_IP_CONFIG* ipConfig);
Parameter	param[in] handle operation handle param[out] ipConfig ip configuration
Return Value	status code

Note	

3) Set IP configurations IRC_NET_SetIpConfig

Options	Introduction
Description	Set IP configurations
Function	IRC_NET_SetIpConfig(IRC_NET_HANDLE handle, const IRC_NET_IP_CONFIG* ipConfig)
Parameter	param[in] handle operation handle param[in] ipConfig ip configurations
Return Value	status code
Note	

4) Get Web logo image IRC_NET_GetLogoPicture

Options	Introduction
Description	Get Web Logo image
Function	int IRC_NET_GetLogoPicture(IRC_NET_HANDLE handle, int id, const char* filePath);
Parameter	param[in] handle operation handle param[in] id 0-login interface logo,1-main interface logo

	param[in] filePath file path
Return Value	status code
Note	

5) Set Web logo image IRC_NET_SetLogoPicture

Options	Introduction
Description	Set Web Logo image
Function	int IRC_NET_SetLogoPicture(IRC_NET_HANDLE handle, int id, const char* filePath);
Parameter	param[in] handle operation handle param[in] id 0-login interface logo,1-main interface logo param[in] filePath file path
Return Value	status code
Note	

6) 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	status code
Note	

7) Power restart IRC_NET_SystemRestart

Options	Introduction
Description	Power Restart
Function	int IRC_NET_SystemRestart(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code
Note	

8) Restore factory defaults IRC_NET_RestoreFactoryDefaults

Options	Introduction
Description	Restore factory defaults
Function	int IRC_NET_RestoreFactoryDefaults(IRC_NET_HANDLE handle);
Parameter	param[in] handle operation handle
Return Value	status code

Note	

6 PTZ Control

6.1 Basic control

1) Pan tilt control operation IRC_NET_PtzControl

Options	Introduction
Description	Pan tilt control operation
Function	int IRC_NET_PtzControl(IRC_NET_HANDLE handle, const IRC_NET_PTZ_CONTROL* ptzControl);
Parameter	Param [in] handle operation handle Param [in] ptzControl pan tilt control
Return Value	status code
Note	

2) Get zoom IRC_NET_GetPtzZoomMultiplier

Options	Introduction
Description	Get zoom
Function	int IRC_NET_GetPtzZoomMultiplier(IRC_NET_HANDLE handle, int channel,

	float* zoomMultipler);
Parameter	param[in] handle operation handle param[in] channel channel param[out] zoomMultipler zoom value
Return Value	status code
Note	

3) Set zoom IRC_NET_SetPtzZoomMultiplier

Options	Introduction
Description	Set zoom
Function	int IRC_NET_SetPtzZoomMultiplier(IRC_NET_HANDLE handle, int channel, float zoomMultipler);
Parameter	param[in] handle operation handle param[in] channel channel param[in] zoomMultipler zoom value
Return Value	status code
Note	

4) PTZ precise positioning IRC_NET_PtzPrecisePosition

Options	Introduction

Description	Pan tilt precise positioning
Function	int IRC_NET_PtzPrecisePosition(IRC_NET_HANDLE handle, const IRC_NET_PTZ_POSITION_PARAM* positionParam);
Parameter	Param [in] handle operation handle Param [in] positionConfigure precise positioning configuration
Return Value	status code
Note	

5) 3D positioning IRC_NET_Ptz3DPosition

Options	Introduction
Description	3D positioning
Function	int IRC_NET_Ptz3DPosition(IRC_NET_HANDLE handle, const IRC_NET_PTZ_3D_POSITION_PARAM* positionParam);
Parameter	Param [in] handle operation handle Param [in] positionParam 3D positioning configuration
Return Value	status code

Note	

6) Region focus IRC_NET_PtzRegionFocus

Options	Introduction
Description	Region focus
Function	int IRC_NET_PtzRegionFocus(IRC_NET_HANDLE handle, const IRC_NET_PTZ_REGION_FOCUS_PARAM* focusParam)
Parameter	param[in] handle operation handle param[in] focusParam region focus parameters
Return Value	status code
Note	

7) Manual tracking IRC_NET_PtzManualTrack

Options	Introduction
Description	Manual tracking
Function	IRC_NET_PtzManualTrack(IRC_NET_HANDLE handle, const IRC_NET_PTZ_MANUAL_TRACK_PARAM* trackParam)

Parameter	param[in] handle operation handle param[in] trackParam manual tracking parameters
Return Value	status code
Note	

8) 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	status code
Note	

9) Set precise tracking positioning

IRC_NET_PtzTrackingPosition

Options	Introduction
Description	Set precise tracking positioning
Function	int IRC_NET_PtzTrackingPosition(IRC_NET_HANDLE handle,

	const IRC_NET_PTZ_TRACKING_POSITION_PARAM * positionParam);
Parameter	param[in] handle operation handle param[in] positionParam precise tracking positioning param
Return Value	status code
Note	

10) Get PTZ position IRC_NET_GetCurrentPtz

Options	Introduction
Description	Get Gimbal Position
Function	int IRC_NET_GetCurrentPtz(IRC_NET_HANDLE handle, IRC_NET_PTZ_POSITION_PARAM* positionParam)
Parameter	param[in] handle operation handle param[out] positionParam Gimbal position parameter
Return Value	status code
Note	

11) Get PTZ position_V1 IRC_NET_GetCurrentPtz_V1

Options	Introduction
Description	Get Gimbal Position_V1
Function	int IRC_NET_GetCurrentPtz_V1(IRC_NET_HANDLE handle, IRC_NET_PTZ_POSITION_PARAM_V1* positionParam);
Parameter	param[in] handle operation handle param[out] positionParam Gimbal position parameter
Return Value	status code
Note	

12) Get PTZ running status IRC_NET_GetPtzRunningStatus

Options	Introduction
Description	Get PTZ running status
Function	int IRC_NET_GetPtzRunningStatus(IRC_NET_HANDLE handle, IRC_NET_PTZ_RUNNING_STATUS_PARAM* statusParam);
Parameter	Param [in] handle operation handle param[out] statusParam PTZ running status

	parameters
Return Value	status code
Note	

13) Get PTZ power-off memory configuration

IRC_NET_GetPtzPowerOffConfig

Options	Introduction
Description	Get PTZ power-off memory configuration
Function	int IRC_NET_GetPtzPowerOffConfig(IRC_NET_HANDLE handle, IRC_NET_PTZ_POWEROFF_CONFIG* powerOffConfig);
Parameter	Param [in] handle operation handle param[out] powerOffConfig PTZ power-off memory configuration
Return Value	status code
Note	

14) Set PTZ power-off memory configuration

IRC_NET_SetPtzPowerOffConfig

Options	Introduction
Description	Set PTZ power-off memory configuration

Function	int IRC_NET_SetPtzPowerOffConfig(IRC_NET_HANDLE handle, const IRC_NET_PTZ_POWEROFF_CONFIG* powerOffConfig);
Parameter	Param [in] handle operation handle param[in] powerOffConfig PTZ power-off memory configuration
Return Value	status code
Note	

15) PTZ restore default configuration IRC_NET_ResetPtzConfig

Options	Introduction
Description	Cloud tilt restoration default configuration
Function	int IRC_NET_ResetPtzConfig(IRC_NET_HANDLE handle);
Parameter	Param [in] handle operation handle
Return Value	status code
Note	

6.2 Auxiliary function

1) Get the status of PTZ auxiliary command

IRC_NET_GetPtzAuxFuncState

Options	Introduction
Description	Get the status of PTZ auxiliary command
Function	int IRC_NET_GetPtzAuxFuncState(IRC_NET_HANDLE handle, IRC_NET_PTZ_AUX_FUNC_STATE* ptzAuxFuncState);
Parameter	Param [in] handle operation handle Param [out] ptzAuxFuncState PTZ auxiliary function status
Return Value	status code
Note	

2) 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	status code
Note	need to enable laser ranging first (enable it on the Web)

3) Get acceleration related data IRC_NET_GetAccelerationData

Options	Introduction
Description	Get acceleration related data
Function	int IRC_NET_GetAccelerationData(IRC_NET_HANDLE handle, IRC_NET_ACCELERATION_DATA* accelerationData);
Parameter	param[in] handle operation handle param[out] accelerationData acceleration related data
Return Value	status code
Note	

6.3 Preset

1) Get the number of presets IRC_NET_QueryPtzPresetSize

Options	Introduction

Description	Get the number of presets
Function	int IRC_NET_QueryPtzPresetSize(IRC_NET_HANDLE handle, int* size);
Parameter	Param [in] handle operation handle Param [out] size query quantity
Return Value	status code
Note	

2) Get preset information IRC_NET_QueryPtzPreset

Options	Introduction
Description	Get preset information
Function	int IRC_NET_QueryPtzPreset(IRC_NET_HANDLE handle, IRC_NET_PTZ_PRESET_INFO ptzPresetInfo[], int inSize, int* outSize);
Parameter	Param [in] handle operation handle Param ptzPresetInfo Param [in] inSize preset input size Param [out] outSize preset output size
Return Value	status code

Note	

3) Preset control IRC_NET_PtzPresetControl

Options	Introduction
Description	Preset control
Function	int IRC_NET_PtzPresetControl(IRC_NET_HANDLE handle, int ptzPresetCmd, const IRC_NET_PTZ_PRESET_INFO* ptzPresetInfo);
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	status code
Note	

4) 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	status code
Note	

6.4 Cruise

1) 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	status code
Note	

2) Get tour configuration information

IRC_NET_QueryPtzTour

Options	Introduction

Description	Get cruise group configuration information
Function	int IRC_NET_QueryPtzTour(IRC_ NET_ HANDLE handle, int id, IRC_NET_PTZ_TOUR_INFO ptzTourInfo[], int inSize, int*outSize) ;
Parameter	Param [in] handle operation handle Param [in] id Cruise group id, range [1255], query all when the value is -1 Param ptzTourInfo cruise group configuration Param [in] inSize Cruise group input size Param [out] outSize Cruise group output size
Return Value	status code
Note	

3) Tour control IRC_NET_PtzTourControl

Options	Introduction
Description	Tour control
Function	int IRC_NET_PtzTourControl(IRC_NET_HANDLE handle, int ptzTourCmd,

	const IRC_NET_PTZ_TOUR_INFO* ptzTourInfo);
Parameter	Param [in] handle operation handle Param [in] ptzTourCmd tour control command, refer to IRC_NET_PTZ_TOUR_CMD_TYPE Param [in] ptzTourInfo tour information.
Return Value	status code
Note	

6.5 Parttern scan

1) Get pattern configuration quantity

IRC_NET_QueryPtzPatternSize

Options	Introduction
Description	Get pattern configuration quantity
Function	int IRC_NET_QueryPtzPatternSize(IRC_NET_HANDLE handle, int* size);
Parameter	param[in] handle operation handle param[out] size query quantity
Return Value	status code
Note	

2) Get pattern configuration IRC_NET_QueryPtzPattern

Options	Introduction
Description	Get pattern configuration
Function	int IRC_NET_QueryPtzPattern(IRC_NET_HANDLE handle, IRC_NET_PTZ_PATTERN_CONFIG* ptzPatternConfig);
Parameter	Param [in] handle operation handle Param [out] ptzPatternConfig patrol configuration
Return Value	status code
Note	

3) Parttern scan control IRC_NET_PtzPatternControl

Options	Introduction
Description	Parttern sacn control
Function	int IRC_NET_PtzPatternControl(IRC_NET_HANDLE handle, const IRC_NET_PTZ_PATTERN_CONFIG* ptzPatternControl);
Parameter	Param [in] handle operation handle Param [in] ptzPatternControl Patrol control

Return Value	status code
Note	

6.6 Region scan

1) Get area scanning information

IRC_NET_QueryPtzRegionScanInfo

Options	Introduction
Description	Get area scanning information
Function	int IRC_NET_QueryPtzRegionScanInfo(IRC_NET_HANDLE handle, IRC_NET_REGION_SCAN_INFO regionScanInfos[], int inSize, int* outSize)
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	status code
Note	

2) Set area scanning information

IRC_NET_SetPtzRegionScanInfo

Options	Introduction
Description	Set Area Scanning Information
Function	int IRC_NET_SetPtzRegionScanInfo(IRC_NET_HANDLE handle, const IRC_NET_REGION_SCAN_INFO* regionScanInfo)
Parameter	param[in] handle operation handle param[in] regionScanInfo area scanning information
Return Value	status code
Note	

3) Delete area scanning information

IRC_NET_DeletePtzRegionScanInfo

Options	Introduction
Description	Delete Area Scanning Information
Function	int IRC_NET_DeletePtzRegionScanInfo(IRC_NET_HANDLE handle, int id)
Parameter	param[in] handle operation handle

	param[in] id Area Scanning id，refer to IRC_NET_REGION_SCAN_INFO，id≥1，id = -1: delete all
Return Value	status code
Note	

4) Area scanning control IRC_NET_PtzRegionScanControl

Options	Introduction
Description	Area Scanning Control
Function	int IRC_NET_PtzRegionScanControl(IRC_NET_HANDLE handle, int id, bool state)
Parameter	param[in] handle operation handle param[in] id Area Scanning id，refer to IRC_NET_REGION_SCAN_INFO param[in] state Area scanning state，true-start， false-stop
Return Value	status code
Note	

6.7 Action

1) Get boot-up actions IRC_NET_QueryPtzBootActionInfo

Options	Introduction
Description	Get Boot-up Actions
Function	int IRC_NET_QueryPtzBootActionInfo(IRC_NET_HANDLE handle, IRC_NET_BOOT_ACTION_INFO* bootActionInfo)
Parameter	param[in] handle operation handle param[out] bootActionInfo Boot-up Actions
Return Value	status code
Note	

2) Set boot-up actions IRC_NET_SetPtzBootActionInfo

Options	Introduction
Description	Set Boot-up Actions
Function	int IRC_NET_SetPtzBootActionInfo(IRC_NET_HANDLE handle, const IRC_NET_BOOT_ACTION_INFO* bootActionInfo)
Parameter	param[in] handle operation handle

	param[in] bootActionInfo Boot-up Actions
Return Value	status code
Note	

3) Get idle actions IRC_NET_QueryPtzParkActionInfo

Options	Introduction
Description	Get Idle Actions
Function	int IRC_NET_QueryPtzParkActionInfo(IRC_NET_HANDLE handle, IRC_NET_PARK_ACTION_INFO* parkActionInfo)
Parameter	param[in] handle operation handle param[out] parkActionInfo Idle Actions
Return Value	status code
Note	

4) Set idle actions IRC_NET_SetPtzParkActionInfo

Options	Introduction
Description	Set Idle Actions
Function	int IRC_NET_SetPtzParkActionInfo(IRC_NET_HANDLE handle, const IRC_NET_PARK_ACTION_INFO*

	parkActionInfo)
Parameter	param[in] handle operation handle param[in] parkActionInfo Idle Actions
Return Value	status code
Note	

6.8 Linkage tracking

1) Get linkage tracking config IRC_NET_GetTargetTrackConfig

Options	Introduction
Description	Get Linkage tracking config
Function	int IRC_NET_GetTargetTrackConfig(IRC_NET_HANDLE handle, int channel, IRC_NET_TARGET_TRACK_DATA* targetTrackData)
Parameter	param[in] handle operation handle param[in] channel channel param[out] targetTrackData Linkage tracking config
Return Value	status code
Note	

2) Set linkage tracking config IRC_NET_SetTargetTrackConfig

Options	Introduction
Description	Set Linkage tracking config
Function	int IRC_NET_SetTargetTrackConfig(IRC_NET_HANDLE handle, int channel, const IRC_NET_TARGET_TRACK_DATA* targetTrackData);
Parameter	param[in] handle operation handle param[in] channel channel param[in] targetTrackData Linkage tracking config
Return Value	status code
Note	

3) Get Linkage Configuration

IRC_NET_GetGunBallTrackConfigInfo

Options	Introduction
Description	Get Linkage Configuration
Function	int IRC_NET_GetGunBallTrackConfigInfo(IRC_NET_HANDLE handle, IRC_NET_GUN_BALL_TRACK_CONFIG_INFO*

	trackConfigInfo);
Parameter	param[in] handle operation handle param[out] trackConfigInfo Gun - Ball Linkage Tracker Configuration Information
Return Value	status code
Note	

4) Set Linkage Configuration

IRC_NET_SetGunBallTrackConfigInfo

Options	Introduction
Description	Set Linkage Configuration
Function	int IRC_NET_SetGunBallTrackConfigInfo(IRC_NET_HANDLE handle, const IRC_NET_GUN_BALL_TRACK_CONFIG_INFO* trackConfigInfo);
Parameter	param[in] handle operation handle param[in] trackConfigInfo Gun - Ball Linkage Tracker Configuration Information
Return Value	status code
Note	

5) Start Linkage Guidance IRC_NET_StartLinkageTrack

Options	Introduction
Description	Start Linkage Guidance
Function	int IRC_NET_StartLinkageTrack(IRC_NET_HANDLE handle, const IRC_NET_LINKAGE_TRACK_PARAM* linkageTrackParam);
Parameter	param[in] handle operation handle param[in] linkageTrackParam Linkage Guidance Parameters
Return Value	status code
Note	

6) Stop Linkage Guidance IRC_NET_StopLinkageTrack

Options	Introduction
Description	Stop Linkage Guidance
Function	int IRC_NET_StopLinkageTrack(IRC_NET_HANDLE handle, int linkageId);
Parameter	param[in] handle operation handle param[in] linkageId Target ID for Radar/Swivel

	Guidance Linkage
Return Value	status code
Note	

7) Update Linkage Guidance

IRC_NET_UpdateLinkageTrackParam

Options	Introduction
Description	Update Linkage Guidance
Function	int IRC_NET_UpdateLinkageTrackParam(IRC_NET_HANDLE handle, int linkageId);
Parameter	param[in] handle operation handle param[in] linkageId Target ID for Radar/Swivel Guidance Linkage
Return Value	status code
Note	

6.9 Swivel control

1) Pan/Tilt control IRC_NET_SwivelControl

Options	Introduction
Description	Pan/Tilt Control

Function	int IRC_NET_SwivelControl(IRC_NET_HANDLE handle, int swivelCmd);
Parameter	param[in] handle operation handle param[in] swivelCmd Pan/Tilt control command, refer to IRC_NET_SWIVEL_CMD_TYPE
Return Value	status code
Note	

2) Electronic PTZ IRC_NET_ElectronicPtzControl

Options	Introduction
Description	Electronic PTZ
Function	int IRC_NET_ElectronicPtzControl(IRC_NET_HANDLE handle, const IRC_NET_ELECTRONIC_PTZ_CONTROL_INFO * controlInfo);
Parameter	param[in] handle operation handle param[in] controlInfo Electronic PTZ Control Parameters
Return Value	status code
Note	

Chapter 4 Data Introduction

1. IRC_NET_DEV_SEARCH_INFO

Structure Definition

typedef struct
{
char ip[IRC_NET_IP_LEN_MAX]; ///< Device IP
}

2. IRC_NET_LOGIN_INFO

Structure Definition

typedef struct
{
char ip[16]; ///< device IP
int port; ///< device port
char username[IRC_NET_NAME_LEN_MAX]; ///< user name
char password[IRC_NET_NAME_LEN_MAX]; ///< password
}

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; ///< x-coordinate
int y; ///< y-coordinate
}

6. IRC_NET_RECT

Structure Definition

typedef struct
{
int top; ///< up
int left; ///< left
int right; ///< right
int bottom; ///< down
}

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 defaule
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 measurment 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; ///< temperature rule ID，start from 1，invalid when adding
rules

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

    IRC_NET_POINT startPoint; ///< Start point coordinates

    IRC_NET_POINT endPoint; ///< End point coordinates

    int alarmType; ///< type of alarm rule, 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; ///< dejitter time, default to 5 seconds

    bool alarmLinkageSnapshotEnable; ///< Alarm-triggered snapshot
enable
```

}

11. IRC_NET_TEMP_RULE_INDEX

Structure Definition

typedef struct
{
int presetId; ///< preset Id， start from 0
int type; ///< type of temperature measurement rule， refer to IRC_NET_TEMP_RULE_TYPE
int id; ///< ID or temperature measurement rule， start from 1
}

12. IRC_NET_TEMP_INFO

Structure Definition

typedef struct
{
float avgTemp; ///< average temperature
float centerTemp; ///< center temperature
float maxTemp; ///< highest temperature
float minTemp; ///< lowest temperature
IRC_NET_POINT maxTempPoint;///< 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 thershold
}

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_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
}

20. IRC_NET_TEMP_ABILITY

Structure Definition

Temp ability
typedef struct
{
int singlePointTemp; ///< 单点测温
int tempLevelEnd; ///< 温度截止
}

21.IRC_NET_STATIC_ABILITY

Structure Definition

Static ability
typedef struct
{
int lowLevelMin; ///< 低增益测温范围最小值
int lowLevelMax; ///< 低增益测温范围最大值
int highLevelMin; ///< 高增益测温范围最小值
int highLevelMax; ///< 高增益测温范围最大值
}

22.IRC_NET_PTZ_CONTROL

Structure Definition

Pan tilt direction control

Typedef struct

{

Int channel///< passageway

Int type///< pan tilt control function type, refer to IRC-NET-PTZ-CMD-TYPE

Int param1///< parameter 1, please refer to the document for details on gimbal command parameters

Int param2///< parameter 2, please refer to the document for the PTZ command parameters

Int param3///< parameter 3, please refer to the document for the PTZ command parameters

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

}

23.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 deprostState///< 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
}
```

24.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
}

25.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
}
```

26.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
}

27.IRC_NET_PTZ_PATTERN_CONFIG

Structure Definition

Patrol configuration
Typedef struct
{
IRC-NET-PTZ-PATTERN_INFO patternInfo
[IRC-NET-PATTERN_NUM-MAX]///< patrol information
}

28.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
}

29.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]
}
```

30.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.
}

31.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
}
```

32.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
}

33.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
}

34.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 defaule

bool alarmLinkageSnapshotEnable; ///< Alarm-triggered snapshot enable

}

35.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
}

36.IRC_NET_TEMP_ALARM_RULE_INFO_G1

Structure Definition
Temperature measurement alarm rules typedef struct

```
{
    float thresholdTemp; ///< temperature threshold
    bool enable; ///< Level enable switch
    float tempLevel1; ///< Level 1
    float tempLevel2; ///< Level 2
    float tempLevel3; ///< Level 3
}
```

37.IRC_NET_PTZ_LINKAGE_SET

Structure Definition

typedef struct
{
bool enable; ///< Enable switch
int function; ///< Linkage function, refer to IRC_NET_LINKAGE_FUNCTION_SET
int functionId; ///< Linkage function ID
int trackTime; ///<Tracking duration
int trackMode; ///< Tracking mode: 1-manual setting, 2-field of view synchronization, 3-automatic
int trackZoom; ///< Zoom of auxiliary tracking channel
int trackCoverTimeOut; ///< Loss time
bool trackLaser; ///< Whether to continuously perform laser ranging

during tracking }

38.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
}

39.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
}
```

40.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
}

41.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

}

42.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
}

43.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
}
```

44.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
}

45.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

}

46.IRC_NET_TEMP_EXT_INFO_CB

Structure Definition

Extended information for temperature callback
typedef struct
{
uint64_t utcTime; ///< UTC millisecond timestamp
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, ℃
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
}
```

47.IRC_NET_TEMP_INFO_CB

Structure Definition

Temperature information in temperature callback
typedef struct
{
char* temp; ///< frame data
int width; ///< width
int height; ///< height
}

48.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
}
```

49.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
}

50.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
}
```

51.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
}

52.IRC_NET_IP_TARGET_RECOGNITION_CONFIG

Structure Definition

Channel target recognition configuration
typedef struct
{
bool enable; ///< Enable
int sensitivity; ///< Sensitivity, range: 0-100, default is 70
IRC_NET_LINKAGE_SET linkageSet; ///< Linkage action
IRC_NET_PTZ_LINKAGE_SET ptzLinkageSet; ///< PTZ linkage set
int type; ///< Detection type: 0-all, 1-human, 2-vehicle, 3-object, 4-truck, 5-ship, 6-airplane, 7-UAV, 8-bird
}

53.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
}
```

54.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
}

55.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
}
```

56.IRC_NET_BOOT_ACTION_INFO

Structure Definition

Boot-up Actions
typedef struct
{
bool enable; ///< boot-up actions enable
int actionType; ///< boot-up action type，refer to IRC_NET_ACTION_TYPE
int lineScanId; ///< Line sacn id
int regionScanId; ///< Area scan id
int presetId; ///< Preset id
int patternId; ///< Pattern id
int tourId; ///< Tour id
}

57.IRC_NET_PARK_ACTION_INFO

Structure Definition

typedef struct
{
bool enable; ///< idle actions enable
int actionType; ///< idle action type, refer to IRC_NET_ACTION_TYPE 0: None
int lineScanId; ///< Line sacn id
int regionScanId; ///< Area scan id
int presetId; ///< Preset id
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
}

58.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
}
```

59.IRC_NET_LASER_DISTANCE_OSD_TITLE_FORMAT

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
}

60.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
}

61.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)

}

62.IRC_NET_VIDEOIN_LEVEL

Structure Definition

Global video enhancement parameters
typedef struct
{
bool enable; ///< enable;
int level; ///< level, 0-100;
}

63.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;

}

64.IRC_NET_REGIONAL_VIDEO_ENHANCE_RANGE

Structure Definition

Local video enhancement range
typedef struct
{
int level; ///< level, 0-100;
IRC_NET_POINT region[2];///< Coordinate;
}

65.IRC_NET_PTZ_TRACKING_POSITION_PARAM

Structure Definition

Precise tracking positioning param
typedef struct
{
float panSpeed; ///< Pan Speed 0-655.35
float tiltSpeed; ///< Tilt Speed 0-655.35
int derection; ///< Direction 0x00 Clockwise Downward 0x01
Clockwise Upward 0x10Counter-Clockwise Downward 0x11

Counter-Clockwise Upward

}

66.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:°/s
}

67.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
}
```

68.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
}

69.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

}

70.IRC_NET_DAY_NIGHT_MODE_PARAM

Structure Definition

VIS day/night mode param
typedef struct
{
int dayNightMode; ///< day/night mode，0-auto，1-day, 2-night,
3-time period;
int delay; ///< Delay time, effective in auto mode only
int start; ///< Daytime start time, effective in time-range mode only
int end; ///< Daytime end time, effective in time-range mode only
int sensitivity; ///< Sensitivity range (1-7), effective in auto mode
only
}

71.IRC_NET_PTZ_POWEROFF_CONFIG

Structure Definition

```
typedef struct
{
    bool enable; ///< Enable, 0-disabled, 1-enabled
    int interval; ///< Power-off memory parameter update frequency,
unit: ms, value range: 1000-30000ms
}
```

72.IRC_NET_PTZ_RUNNING_STATUS_PARAM

Structure Definition

```
typedef struct
{
    bool isMoving; ///< Whether the PTZ is rotating
    int timingTaskID; ///< Timing task ID
    int functionID; ///< ID of the running task
    int trigger; ///< Trigger of the running task, refer to
IRC_NET_TASK_TRIGGER_TYPE
    int function; ///< Running task, refer to IRC_NET_TASK_TYPE
    uint64_t startTime; ///< UTC timestamp in milliseconds
}
```

73.IRC_NET_LINKAGE_TRACK_PARAM

Structure Definition

```
typedef struct
{
    float pan; ///< Horizontal angle
    float tilt; ///< Vertical angle
    float zoom[IRC_NET_CHANNEL_NUM_MAX]; ///< Absolute zoom multiple of the tracking channel lens, accurate to 1 decimal place, pass 0 for channels that do not need tracking
    IRC_NET_TARGET_SIZE targetSize;///< Target size, Z value will be preferentially used for direct zooming if available; target size is required for zooming if no Z value
    int linkageId; ///< Linkage target ID for radar/circular scan guidance
    int searchTimeout; ///< Search timeout, unit: seconds, range: 1-10
    int waitTimeout; ///< Wait timeout for guidance, automatically ends guidance tracking after timeout, unit: seconds, range: 180-600
    float distance; ///< Distance from the tracked object to the lens, unit: meters
}
```

74.IRC_NET_GUN_BALL_TRACK_CONFIG_INFO

Structure Definition
typedef struct {

```
    char userName[IRC_NET_NAME_LEN_MAX]; ///< Username
    char password[IRC_NET_NAME_LEN_MAX]; ///< Password
    char ip[IRC_NET_IP_LEN_MAX]; ///< ip
    int port; ///< Port information
    int keepaliveInterval; ///< Keep-alive interval
    bool enable; ///< Enable
    bool isHttps; ///< Whether it is HTTPS
}
```

75.IRC_NET_ELECTRONIC_PTZ_CONTROL_INFO

Structure Definition

```
typedef struct
{
    int channel; ///< Channel
    int cmd; ///< PTZ control function type, refer to
IRC_NET_ELECTRONIC_PTZ_CMD_TYPE
    int speed; ///< Speed: 1-slowest, 8-fastest
    int stop; ///< 0-start, 1-stop
}
```

76.IRC_NET_LINKAGE_SET

Structure Definition

```
typedef struct
{
    IRC_NET_ALARM_LINKAGE_INFO snapshotLinkageInfo; ///<
Snapshot linkage information
    IRC_NET_ALARM_LINKAGE_INFO recordLinkageInfo; ///<
Recording linkage information
}
```

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
typedef enum { IRC_NET_EXCEPTION_PREVIEW_OFFLINE = 1001, ///< preview offline IRC_NET_EXCEPTION_ALARM_OFFLINE = 1002, ///< alarm offline IRC NET EXCEPTION_DEV_OFFLINE = 1003 ///<device offline } IRC_NET_EXCEPTION_TYPE;

3. Alarm Type IRC_NET_ALARM_TYPE

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

4. Frame Forma IRC_NET_FRAME_FMT

Definition Description
typedef enum {

```
    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

typedef enum
{
IRC_NET_STREAM_MAIN = 0, ///< main stream
IRC_NET_STREAM_SUB ///< sub stream
IRC_NET_STREAM_SUN, ///< Third stream
IRC_NET_STREAM_FRAME, ///< Private frame
IRC_NET_STREAM_PANORAMA ///< Panoramic stream; when selecting this stream type, the frame format must be set to "IRC_NET_FRAME_FMT_RGBA"
}

6. Equipment capability type IRC_NET_DEV_ABILITY_TYPE

Definition Description

typedef enum

{

IRC_NET_DEV_ABILITY_PTZ = 0, ///< PTZ ability, refer to IRC_NET_PTZ_ABILITY

IRC_NET_DEV_ABILITY_TEMP, ///< Temperature ability, refer to IRC_NET_TEMP_ABILITY

IRC_NET_DEV_ABILITY_STATIC, ///< Static ability, refer to IRC_NET_STATIC_ABILITY

}

7. Log level IRC_NET_LOG_LEVEL

Definition Description

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
}

8. Temperature measurement alarm rule type IRC_NET_TEMP_ALARM_RULE_TYPE

Definition Description

typedef enum
{
IRC_NET_TEMP_ALARM_RULE_NONE=0,///<no rule
IRC_NET_TEMP_ALARM_RULE_AVG_TEMP_GT,///<the average temperature is greater than
IRC_NET_TEMP_ALARM_RULE_AVG_TEMP_LT,///<the average temperature is less than
IRC_NET_TEMP_ALARM_RULE_HIGHT_TEMP_GT,///<high 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

typedef enum
{
IRC_NET_PTZ_PRESET_CMD_ADD=0,///<add preset point
IRC_NET_PTZ_PRESET_CMD_UPDATE_NAME,///<update preset point names
IRC_NET_PTZ_PRESET_CMD_GOTO,///<move to preset point
IRC_NET_PTZ_PRESET_CMD_DELETE,///<delete a single preset point
IRC_NET_PTZ_PRESET_CMD_DELETE_ALL,///<delete all preset points
}

12. Cruise control function type

IRC_NET_PTZ_TOUR_CMD_TYPE

Definition Description
typedef enum { IRC_NET_PTZ_TOUR_CMD_SAVE=0,///<save Cruise Group IRC_NET_PTZ_TOUR_CMD_START,///<start cruising

```
    IRC_NET_PTZ_TOUR_CMD_STOP,///<stop Cruise
    IRC_NET_PTZ_TOUR_CMD_DELETE,///<delete a single cruise
group
    IRC_NET_PTZ_TOUR_CMD_DELETE_ALL//<delete all cruise
control groups
}
```

13. Type of patrol control function

IRC_NET_PTZ_PATTERN_CMD_TYPE

Definition Description
typedef enum { 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_REPLAY,///<start replay IRC_NET_PTZ_PATTERN_CMD_STOP_REPLAY,///<end replay IRC_NET_PTZ_PATTERN_CMD_ELETE_ALL//<delete all patrols

}

14. SD card file type IRC_NET_SD_FILE_TYPE

Definition Description

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

15. Temperature measurement rule type IRC_NET_TEMP_RULE_TYPE

Definition Description
Temperature measurement rule type typedef struct { IRC_NET_TEMP_RULE_POINT = 0, ///< point IRC_NET_TEMP_RULE_LINE, ///< line

```
    IRC_NET_TEMP_RULE_RECT, ///< rectangle
    IRC_NET_TEMP_RULE_CIRCLE, ///< circle
    IRC_NET_TEMP_RULE_POLYGON ///< polygon
}
```

16. Alarm action type IRC_NET_ALARM_ACTION

Definition Description

Alarm action type
typedef struct
{
IRC_NET_ALARM_ACTION_SINGLE = 0, ///< single pulse alarm
IRC_NET_ALARM_ACTION_START, ///< alarm 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
Color Palette type typedef struct { IRC_NET_PALETTE_WHITE_HOT = 0, ///< WhiteHot IRC_NET_PALETTE_BLACK_HOT, ///< BlackHot IRC_NET_PALETTE_RAINBOW, ///< Rainbow

```
    IRC_NET_PALETTE_RAINBOW_HC, ///< RainbowHC
    IRC_NET_PALETTE_IRON, ///< Iron
    IRC_NET_PALETTE_LAVA, ///< Lava
    IRC_NET_PALETTE_SKY, ///< Sky
    IRC_NET_PALETTE_MID_GREY, ///< MidGrey
    IRC_NET_PALETTE_RDGY, ///< RdGy
    IRC_NET_PALETTE_PUOR, ///< PuOr
    IRC_NET_PALETTE_SPECIAL, ///< Special
    IRC_NET_PALETTE_RED, ///< Red
    IRC_NET_PALETTE_ICE_FIRE, ///< IceFire
    IRC_NET_PALETTE_GREE_RED, ///< GreenRed
    IRC_NET_PALETTE_SPECIAL2, ///< Special2
    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
Action Type

```
typedef enum
{
    IRC_NET_ACTION_AUTO_RESET = 0, ///< Auto Homing
    IRC_NET_ACTION_PRESET, ///< Presets
    IRC_NET_ACTION_TOUR, ///< Tour
    IRC_NET_ACTION_PATTERN, ///< Pattern
    IRC_NET_ACTION_LINE_SCAN, ///< Line scan
    IRC_NET_ACTION_AREA_SCAN, ///< Area scan
}
```

21. PTZ linkage function

IRC_NET_LINKAGE_FUNCTION_SET

Structure Definition
typedef enum { IRC_NET_LINKAGE_FUNCTION_NONE = 0, ///< None IRC_NET_LINKAGE_FUNCTION_PRESET, ///< Preset point IRC_NET_LINKAGE_FUNCTION_TOUR, ///< Cruise IRC_NET_LINKAGE_FUNCTION_PATTERN, ///< Pattern scan IRC_NET_LINKAGE_FUNCTION_LINEAR_SCAN, ///< Linear scan IRC_NET_LINKAGE_FUNCTION_REGION_SCAN, ///< Region

```
scan
        IRC_NET_LINKAGE_FUNCTION_TRACK ///< Track
}
```

22. Task trigger type

IRC_NET_TASK_TRIGGER_TYPE

Structure Definition

typedef enum
{
IRC_NET_TASK_TRIGGER_MANUL = 0, ///< Manual
IRC_NET_TASK_TRIGGER_ALARM_LINKAGE, ///< Alarm linkage
IRC_NET_TASK_TRIGGER_TIME_TASK, ///< Timing task
IRC_NET_TASK_TRIGGER_BOOT_ACTION, ///< Boot action
IRC_NET_TASK_TRIGGER_IDLE_ACTION, ///< Idle action
IRC_NET_TASK_TRIGGER_CRUISE, ///< Comprehensive cruise
}

23. Task type IRC_NET_TASK_TYPE

Structure Definition
typedef enum {

```
    IRC_NET_TASK_NONE = 0, ///< No running task
    IRC_NET_TASK_CRUISE, ///< Cruise
    IRC_NET_TASK_PATROL, ///< Patrol
    IRC_NET_TASK_LINE_SCAN, ///< Linear scan
    IRC_NET_TASK_REGION_SCAN, ///< Region scan
    IRC_NET_TASK_LEVEL_ROTATION, ///< Horizontal rotation
    IRC_NET_TASK_SINGLE_TARGET_TRACK, ///< Single target
track
    IRC_NET_TASK_FIRE_TRACK, ///< Fire point track
    IRC_NET_TASK_SMOKE_TRACK, ///< Smoke track
    IRC_NET_TASK_OPTOELECTRONIC_LANKAGE_TRACK,
///< Optoelectronic linkage track
    IRC_NET_TASK_CRUISE_TO_PRESET, ///< Comprehensive
cruise jump to preset point
}
```

24. Electronic PTZ control function type IRC_NET_ELECTRONIC_PTZ_CMD_TYPE

Structure Definition
typedef enum { IRC_NET_ELECTRONIC_PTZ_CMD_UP = 0, ///< Up

```
    IRC_NET_ELECTRONIC_PTZ_CMD_DOWN, ///< Down
    IRC_NET_ELECTRONIC_PTZ_CMD_LEFT, ///< Left
    IRC_NET_ELECTRONIC_PTZ_CMD_RIGHT, ///< Right
    IRC_NET_ELECTRONIC_PTZ_CMD_LEFT_TOP, ///< Left-top
    IRC_NET_ELECTRONIC_PTZ_CMD_RIGHT_TOP, ///<
Right-top
    IRC_NET_ELECTRONIC_PTZ_CMD_LEFT_DOWN, ///<
Left-bottom
    IRC_NET_ELECTRONIC_PTZ_CMD_RIGHT_DOWN, ///<
Right-bottom
    IRC_NET_ELECTRONIC_PTZ_CMD_ZOOM_OUT, ///< Zoom
out
    IRC_NET_ELECTRONIC_PTZ_CMD_ZOOM_IN, ///< Zoom in
}
```

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.

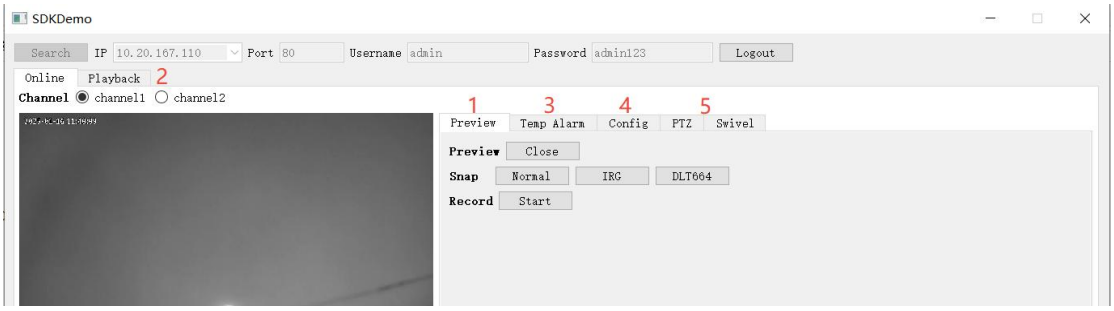
2. Demo Operation

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 Storge managment 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" and "Alarm Data Retrieval" section of this manual, such as Retrieval of tempterature data, Temperature measurement rules and Alarm subscription.

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

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

Appendix

Table 1: PTZ Commands

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

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			