

NetSDK (AI)

Programming Manual



Foreword

Purpose

Welcome to use NetSDK (hereinafter referred to be "SDK") programming manual (hereinafter referred to as "the Manual").

SDK, also known as network device SDK, is a development kit for developer to develop the interfaces for network communication among surveillance products such as Network Video Recorder (NVR), Network Video Server (NVS), IP Camera (IPC), Speed Dome (SD), and intelligence devices.

The Manual describes the SDK interfaces and processes of the intelligent function modules for Intelligent Video Surveillance System (IVSS), Network Video Recorder (NVR), IP Camera (IPC), Intelligent Traffic Camera (ITC), people flow statistics devices and barrier. For more function modules and data structures, refer to NetSDK Development Manual. For detailed information on basic service processes, including initialization, login, general alarms and intelligent alarms, refer to NetSDK Programming Guide.






The example codes provided in the Manual are only for demonstrating the procedure and not assured to copy for use.

Readers

- SDK software development engineers
- Project managers
- Product managers

Safety Instructions

The following categorized signal words with defined meaning might appear in the manual.

Signal Words	Meaning
 DANGER	Indicates a high potential hazard which, if not avoided, will result in death or serious injury.
 WARNING	Indicates a medium or low potential hazard which, if not avoided, could result in slight or moderate injury.
 CAUTION	Indicates a potential risk which, if not avoided, could result in property damage, data loss, lower performance, or unpredictable result.
 TIPS	Provides methods to help you solve a problem or save you time.
 NOTE	Provides additional information as the emphasis and supplement to the text.

Revision History

Version	Revision Content	Release Time
V2.0.0	Updated Appendix 2 Intelligent Events.	February 2025
V1.0.6	Added Appendix 2 Intelligent Events.	December 2023
V1.0.5	Updated some description.	February 2023
V1.0.4	<ul style="list-style-type: none">● Added general behavior event.● Added intelligent transport event.● Added object monitoring event.● Added city management event.● Added parking space detection event.● Added crowd map event.● Added vehicle density map event.● Added heat map event.● Added stereo analysis event.	August 2021
V1.0.3	Deleted fisheye correction library.	May 2021
V1.0.2	Deleted function library avnetsdk.dll and libavnetsdk.so related content, changed font and example codes in 2.5.4.1 and 3.2.4.1.	March 2021
V1.0.1	Changed the callback functions of login and device searching.	March 2020
V1.0.0	First release.	October 2018

Privacy Protection Notice

As the device user or data controller, you might collect personal data of others such as face, fingerprints, car plate number, email address, phone number, GPS and so on. You need to be in compliance with the local privacy protection laws and regulations to protect the legitimate rights and interests of other people by implementing measures include but not limited to: providing clear and visible identification to inform data subject the existence of surveillance area and providing related contact.

About the Manual

- The manual is for reference only. Slight differences might be found between the manual and the product.
- We are not liable for any loss caused by the operations that do not comply with the manual.
- The manual would be updated according to the latest laws and regulations of related jurisdictions. For detailed information, refer to the paper manual, CD-ROM, QR code or our official website. If there is inconsistency between paper manual and the electronic version, the electronic version shall prevail.
- All the designs and software are subject to change without prior written notice. The product

updates might cause some differences between the actual product and the manual. Please contact the customer service for the latest program and supplementary documentation.

- There still might be deviation in technical data, functions and operations description, or errors in print. If there is any doubt or dispute, we reserve the right of final explanation.
- Upgrade the reader software or try other mainstream reader software if the manual (in PDF format) cannot be opened.
- All trademarks, registered trademarks and the company names in the manual are the properties of their respective owners.
- Please visit our website, contact the supplier or customer service if there is any problem occurring when using the device.
- If there is any uncertainty or controversy, we reserve the right of final explanation.

Glossary

This chapter provides the definitions to some of the terms appear in the Manual to help you understand the function of each module.

Term	Definition
IVSS	Intelligent Video Surveillance System is different with the NVR devices which only support the storage function. The IVSS adds the intelligent analysis function to form an integrated management system.
Target Detection	Do the intelligent analysis to detect the target, age, gender and expression in the video.
object Recognition	It contains the target detection. You can detect whether the target in the video are in the face library or not through the intelligent analysis.
History library	It can be used to storage the face pictures that captured by the device.
Face library	You can detect whether the faces are in the face library or not through importing some face images to the IVSS, NVR or the front-end devices in advance.
Arm by channel	Arm/Disarm one or multiple face library to one channel. The scene: The detected face in this channel contrast with the arm library and return the result. It belongs to one mode of the face library arming.
Arm by library	Arm the face library to one or multiple channel. The scene: The people target detected by the channel contrast with this face library and return the result. It belongs to one mode of the face library arming.
Search picture by picture	You can import a picture and a similarity value, and then the IVSS and NVR will search the history library and the face library by this picture to make sure whether there have two same faces between the two libraries. And then it will return the right picture.
ITC	Intelligent Traffic Camera. It can capture the vehicle pictures and automatically analyze the traffic events.
Tripwire detect	Automatically detect the cross trip wire.
Intrusion detect	Automatically detect whether the object enter the alert area or not.
People flow	People information in the camera marking area.
People counting	Real-time count the number of people entering or leaving the camera marking area.
People counting in area	Real-time count the number of people in the camera marking area.

Table of Contents

Foreword	I
Glossary	IV
1 Overview	1
1.1 General	1
1.2 Applicability	2
1.3 Application Scenario	2
1.3.1 People Flow Statistics	2
1.3.2 Intelligent Traffic	2
1.3.3 General Behavior	4
1.3.4 Access Control System	5
2 Function Module	7
2.1 SDK Initialization	7
2.1.1 Introduction	7
2.1.2 Interface Overview	7
2.1.3 Process	7
2.1.4 Example Code	8
2.2 Device Login	9
2.2.1 Introduction	9
2.2.2 Interface Overview	9
2.2.3 Process	9
2.2.4 Example Code	11
2.3 Real-time Monitoring	11
2.3.1 Introduction	11
2.3.2 Interface Overview	11
2.3.3 Process	12
2.3.4 Example Code	15
2.4 Subscribing to Intelligent Event	16
2.4.1 Introduction	16
2.4.2 Interface Overview	16
2.4.3 Process	17
2.4.4 Example Code	18
2.5 Searching for/Playing/Downloading Video and Picture	18
2.5.1 Introduction	18
2.5.2 Interface Overview	19
2.5.3 Process	20
2.5.4 Example Code	22
3 Target Detection and Recognition	27
3.1 Subscribing Face Event	27
3.2 Adding/Deleting/Modifying/Searching the Face Library	27
3.2.1 Introduction	27
3.2.2 Interface Overview	27
3.2.3 Process	28
3.2.4 Example Code	29
3.3 Adding/Deleting/Modifying/Searching People Face	31

3.3.1 Introduction	31
3.3.2 Interface Overview	31
3.3.3 Process	32
3.3.4 Example Code	33
3.4 Arming by Channel or Library	35
3.4.1 Introduction	35
3.4.2 Interface Overview	35
3.4.3 Process	36
3.4.4 Example Code	37
3.5 Searching for Picture by Picture	39
3.5.1 Introduction	39
3.5.2 Interface Overview	39
3.5.3 Process	40
3.5.4 Example Code	41
3.6 Searching for and Downloading Face Video and Picture	43
4 Body Detection	44
4.1 Subscribing Body Event	44
4.2 Searching for the Body Picture	44
4.2.1 Introduction	44
4.2.2 Interface Overview	44
4.2.3 Process	45
4.2.4 Example Code	45
5 People Flow Statistics.....	47
5.1 Subscribing to People Flow Event	47
5.1.1 Introduction	47
5.1.2 Interface Overview	47
5.1.3 Process	48
5.1.4 Example Code	48
5.2 Alarm of People Flow Event	49
5.3 Searching for History Data of People Flow Statistics	49
5.3.1 Introduction	49
5.3.2 Interface Overview	49
5.3.3 Process	50
5.3.4 Example Code	50
6 General Behavior Event	52
6.1 Subscribing to General Behavior Event	52
6.2 Video Searching and Downloading of General Behavior Event	52
7 Intelligent Traffic.....	53
7.1 Subscribing to Intelligent Traffic Event.....	53
7.2 Searching for History Data of Vehicle Flow Statistics	53
7.2.1 Introduction	53
7.2.2 Interface Overview	53
7.2.3 Process	54
7.2.4 Example Code	55
7.3 Adding/deleting/modifying/searching for Blocklist and Allowlist of Vehicle	57
7.3.1 Introduction	57
7.3.2 Interface Overview	57

7.3.3 Process	58
7.3.4 Example Code	59
7.4 Searching for and Downloading Vehicle Picture	61
7.4.1 Introduction	61
7.4.2 Interface Overview	61
7.4.3 Process	62
7.4.4 Example Code	63
8 Barrier.....	65
8.1 Subscribing to Access Control Event	65
8.2 Manager Information of Access Control Card	65
8.2.1 Introduction	65
8.2.2 Interface Overview	65
8.2.3 Process	66
8.2.4 Example Code	67
8.3 Face Management	71
8.3.1 Introduction	71
8.3.2 Interface Overview	71
8.3.3 Process	72
8.3.4 Example Code	73
8.4 Searching for the Record of In-Out the Door	75
8.4.1 Introduction	75
8.4.2 Interface Overview	76
8.4.3 Process	77
8.4.4 Example Code	78
9 Object Monitoring.....	79
9.1 Subscribing to Object Monitoring Events.....	79
9.2 Searching for and Downloading Object Monitoring Video	79
10 City Management.....	80
10.1 Subscribing to management events	80
10.2 Searching for and Downloading City Management Video	80
11 Parking Space Detection.....	82
11.1 Subscribing to Detection Event of Parking Space.....	82
11.2 Searching for and Downloading Event Detection Video of Parking Space.....	82
11.3 Subscribing to Designated Parking Space Picture	83
11.3.1 Overview	83
11.3.2 Interface Overview	83
11.3.3 Process Description	83
11.3.4 Example Code	84
12 Crowd Map	85
12.1 Subscribing to Crowd Map Event.....	85
12.2 Searching for and Downloading Video of Crowd Map	85
13 Vehicle Density Map	86
13.1 Subscribing to Vehicle Density Event	86
13.2 Searching for and Downloading Video of Vehicle Density Event	86
14 Heat Map	87
14.1 Subscribing to Heat Map Data.....	87
14.1.1 Overview	87

14.1.2 Interface Overview	87
14.1.3 Process Description	87
14.1.4 Example Code	88
14.2 Subscribing to Gray Map Data	88
14.2.1 Overview	88
14.2.2 Interface Overview	89
14.2.3 Process Description	89
14.2.4 Example Code	90
15 Stereo Analysis.....	91
15.1 Subscribing to Stereo Analysis Event	91
15.2 Searching for and Downloading Video of Stereo Analysis Event.....	91
15.3 Subscribing to Video Statistics.....	92
15.3.1 Overview	92
15.3.2 Interface Overview	92
15.3.3 Process Description	92
15.3.4 Example Code	93
16 Helme Detection	94
16.1 Subscribing to Helme Detection Event	94
17 Interface Definition	94
17.1 SDK Initialization	94
17.1.1 SDK CLIENT_Init	94
17.1.2 CLIENT_Cleanup.....	94
17.1.3 CLIENT_SetAutoReconnect	95
17.1.4 CLIENT_SetNetworkParam	95
17.2 Device Login	95
17.2.1 CLIENT_LoginWithHighLevelSecurity.....	95
17.2.2 CLIENT_Logout	96
17.3 Real-time Monitoring	97
17.3.1 CLIENT_RealPlayEx	97
17.3.2 CLIENT_StopRealPlayEx	98
17.3.3 CLIENT_SaveRealData	98
17.3.4 CLIENT_StopSaveRealData	98
17.3.5 CLIENT_SetRealDataCallBackEx	98
17.4 Subscribing to Intelligent Event	99
17.4.1 CLIENT_RealLoadPictureEx.....	99
17.4.2 CLIENT_StopLoadPic.....	100
17.5 Searching for and Downloading Intelligent Video and Picture	100
17.5.1 CLIENT_FindFileEx	100
17.5.2 CLIENT_GetTotalFileCount	101
17.5.3 CLIENT_FindNextFileEx	101
17.5.4 CLIENT_FindCloseEx	102
17.5.5 CLIENT_PlayBackByTimeEx2	102
17.5.6 CLIENT_StopPlayBack	103
17.5.7 CLIENT_DownloadByTimeEx	103
17.5.8 CLIENT_StopDownload.....	104
17.5.9 CLIENT_DownloadRemoteFile	104
17.6 Subscribing to Face Event	104

17.7 Adding/Deleting/Modifying/Searching the Face Library	105
17.7.1 CLIENT_OperateFaceRecognitionGroup	105
17.7.2 CLIENT_FindGroupInfo	105
17.8 Adding/Deleting/Modifying/Searching for People Face	106
17.8.1 CLIENT_OperateFaceRecognitionDB	106
17.8.2 CLIENT_OperateFaceRecognitionDB	106
17.8.3 CLIENT_DoFindFaceRecognition.....	106
17.8.4 CLIENT_StopFindFaceRecognition	107
17.8.5 CLIENT_FaceRecognitionPutDisposition.....	107
17.8.6 CLIENT_FaceRecognitionDelDisposition.....	108
17.8.7 CLIENT_SetGroupInfoForChannel	108
17.8.8 CLIENT_AttachFaceFindState.....	108
17.8.9 CLIENT_DetachFaceFindState	109
17.9 Body Detection	109
17.9.1 CLIENT_DownloadRemoteFile	109
17.10 People Flow Statistics	110
17.10.1 CLIENT_AttachVideoStatSummary.....	110
17.10.2 CLIENT_DetachVideoStatSummary	110
17.10.3 CLIENT_StartFindNumberStat.....	110
17.10.4 CLIENT_DoFindNumberStat	111
17.10.5 CLIENT_StopFindNumberStat	111
17.11 Intelligent Traffic	111
17.11.1 CLIENT_FindRecord	111
17.11.2 CLIENT_QueryRecordCount.....	112
17.11.3 CLIENT_FindNextRecord	112
17.11.4 CLIENT_FindRecordClose	113
17.11.5 CLIENT_OperateTrafficList	113
17.11.6 CLIENT_DownloadMultiFile	113
17.11.7 CLIENT_StopLoadMultiFile	114
17.12 Access Control.....	114
17.12.1 CLIENT_FindRecord	114
17.12.2 CLIENT_FindNextRecord	114
17.12.3 CLIENT_FindRecordClose	115
17.12.4 CLIENT_FindRecordClose	115
17.12.5 CLIENT_FindRecordClose	116
17.13 Parking Space Detection	116
17.13.1 Subscribing to designated parking space picture.....	116
17.13.2 Unsubscribing from Designated Parking Space Picture	117
17.14 Heat Map.....	117
17.14.1 Subscribing to Heat Map Data	117
17.14.2 Unsubscribing from Heat Map Data	117
17.14.3 Subscribing to Gray Map Data	118
17.14.4 Unsubscribing from Gray Map Data	118
17.15 Stereo Analysis.....	119
17.15.1 Subscribing to Video Statistics	119
17.15.2 Unsubscribing from Video Statistics	119
18 Callback Function Definition	120

18.1 fDisconnect	120
18.2 fHaveReConnect	120
18.3 fRealDataCallBackEx	121
18.4 fAnalyzerDataCallBack	121
18.5 fDownloadPosCallBack	122
18.6 fDataCallBack	122
18.7 fFaceFindState	123
18.8 fVideoStatSumCallBack	123
18.9 fMultiFileDownloadPosCB	124
18.10 fNotifySnapData	125
18.11 fRawStreamCallBack	125
18.12 fHeatMapGrayCallBack	125
18.13 fVideoStatisticsInfoCallBack	126
Appendix 1 Cybersecurity Recommendations	127
Appendix 2 Intelligent Events	129

1 Overview

1.1 General

The Manual introduces SDK interfaces reference information that includes main function modules, interface definition, and callback definition.

The main function contains the general functions, target detection and object recognition, body detection, people flow statistics, general behavior event, intelligent traffic and barrier.

The development kit might be different dependent on the different environments.

Table 1-1 The files included in development kit

Library type	Library file name	Library file description
Function library	dhnetsdk.h	Header file
	dhnetsdk.lib	Lib file
	dhnetsdk.dll	Library file
	avnetsdk.dll	Library file
Configuration library	avglobal.h	Header file
	dhconfigsdk.h	Header file
	dhconfigsdk.lib	Lib file
	dhconfigsdk.dll	Library file
Auxiliary library of playing (coding and decoding)	dhplay.dll	Playing library
Auxiliary library of "dhnetsdk.dll"	IvsDrawer.dll	Image display library
	StreamConvertor.dll	Transcoding library

Table 1-2 The files included in development kit

Library type	Library file name	Library file description
Function library	dhnetsdk.h	Header file
	libdhnetsdk.so	Library file
	libavnetsdk.so	Library file
Configuration library	avglobal.h	Header file
	dhconfigsdk.h	Header file
	libdhconfigsdk.so	Library file
Auxiliary library of "libdhnetsdk.so"	libStreamConvertor.so	Transcoding library



- The function library and configuration library are necessary libraries.
- The function library is the main body of SDK, which is used for communication interaction between client and products, remotely controls device, queries device data, configures device data information, as well as gets and handles the streams.
- The configuration library packs and parses the structures of configuration functions.
- It is recommended to use auxiliary library of playing (coding and decoding) to parse and play the streams.

- The auxiliary library decodes the audio and video streams for the functions such as monitoring and voice talk, and collects the local audio.

1.2 Applicability

- Recommended memory: No less than 512 M.
- System supported by SDK:
 - ◇ Windows
Windows 10/ Windows 8.1/ Windows 7/ vista/ 2000 and Windows Server 2008/ 2003.
 - ◇ Linux
The common Linux systems such as Red Hat/SUSE

1.3 Application Scenario

1.3.1 People Flow Statistics

For the application of people flow devices.

Figure 1-1 People flow scene



1.3.2 Intelligent Traffic

ITC and ITSE used at the traffic junction capture the traffic violations and count the vehicle flow.

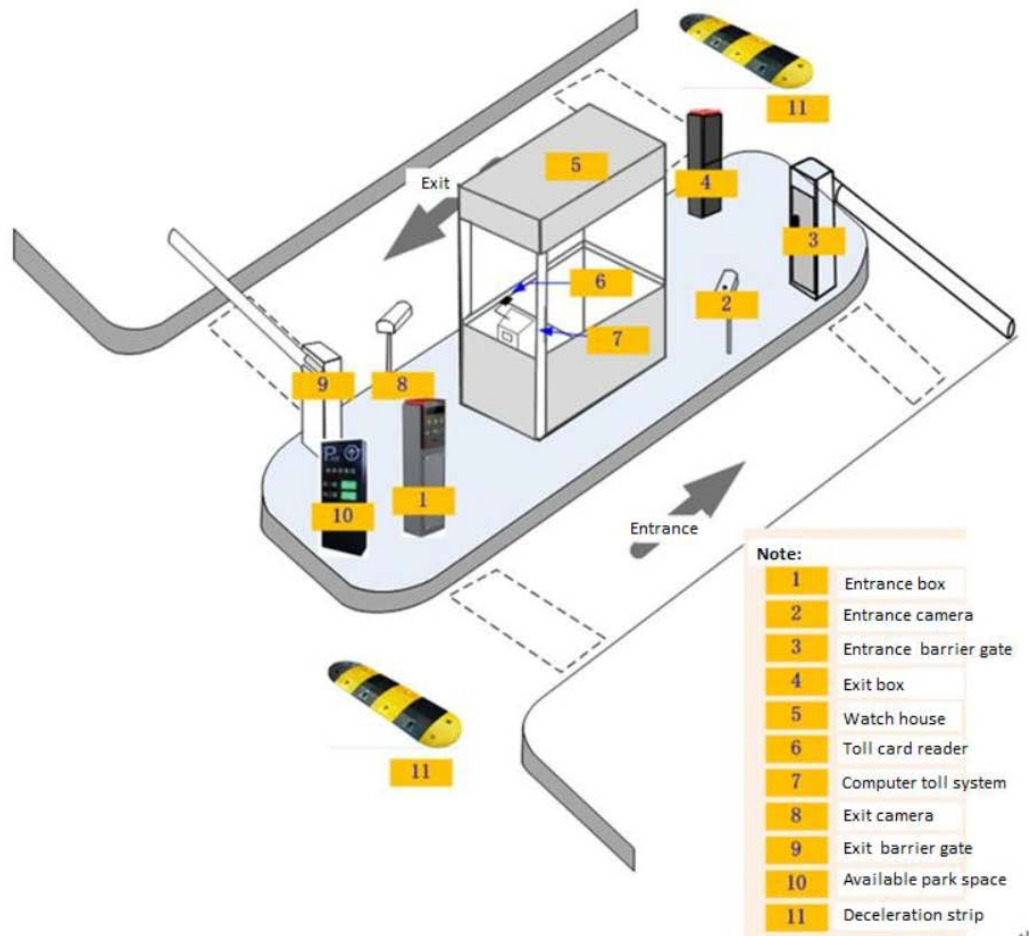
Figure 1-2 ITC and ITSET used at the traffic junction

The screenshot shows the TrafficDemo web interface. At the top, there is a 'Dev Login' section with fields for IP (192.168.1.8), Port (37777), User (admin), and Password (*****), along with a 'Logout' button. Below this are tabs for 'Intelligent Event', 'BW List', 'Traffic Flow', and 'Query Traffic Picture'. The 'Intelligent Event' tab is active, showing a 'Channel' dropdown set to '1', a 'Stop Play' button, and an 'Unsubscribe' button. The main area displays two live video feeds of a road with cars. Below the feeds is a table with the following data:

Index	Time	Event Type	Lane	Plate Number	Plate Color	Vehicle Color
3	2018-10-23 17:14:45	ANPR	1	6G51	Blue	Black
2	2018-10-23 17:14:40	ANPR	1	67BR	Blue	Black
1	2018-10-23 17:14:40	ANPR	2	60609	Black	White

ITC, ITSE and IPMECK used at the parking access control the vehicle enter and exit the parking and monitor whether there is any parking space.

Figure 1-3 ITC, ITSE and IPMECK used at the parking access



1.3.3 General Behavior

People or vehicle across the rule line (tripwire) or intruding the alert zone (intrusion) can generate alarm events. Meanwhile this function can distinguish the target objects (People or vehicle).

Figure 1-4 General behavior scenario (tripwire)



Figure 1-5 General behavior scenario (intrusion)



1.3.4 Access Control System

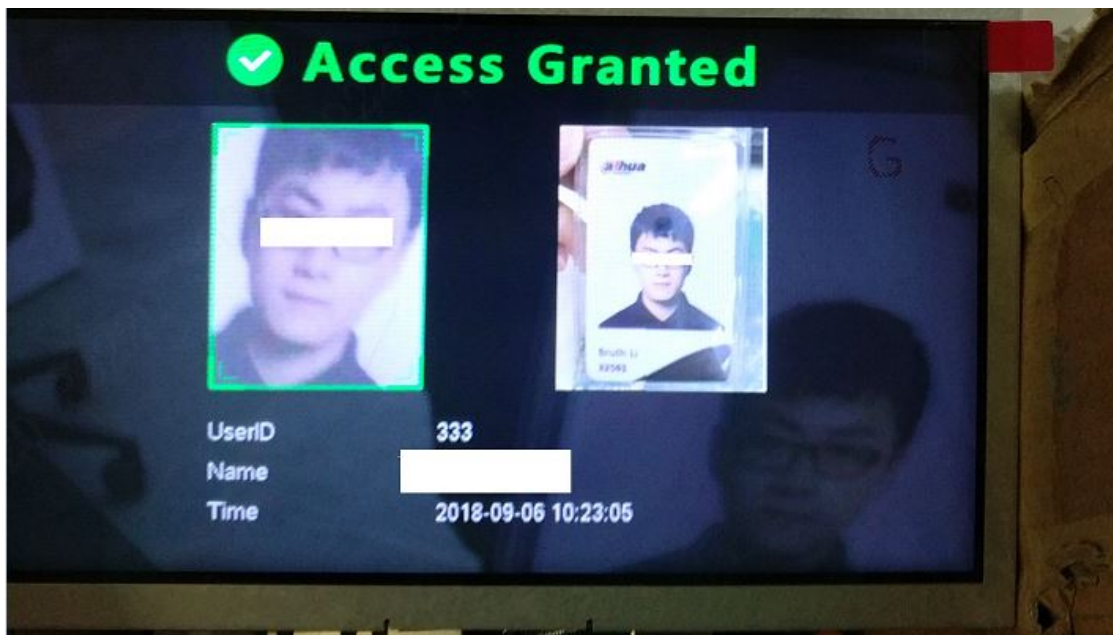
Access control barrier mainly used at the campus, school, business building. You can send the people face pictures and people information to the platform, and then the platform send the data to the barrier system.

Figure 1-6 The appearance of swing barrier



You can open the barrier system by people face or card.

Figure 1-7 Open barrier by people face



2 Function Module

2.1 SDK Initialization

2.1.1 Introduction

Initialization is the first step of SDK to conduct all the function modules. It does not have the surveillance function but can set some parameters that affect the SDK overall functions.

- Initialization occupies some memory.
- Only the first initialization is valid within one process.
- After using this function, call **CLIENT_Cleanup** to release SDK resource.

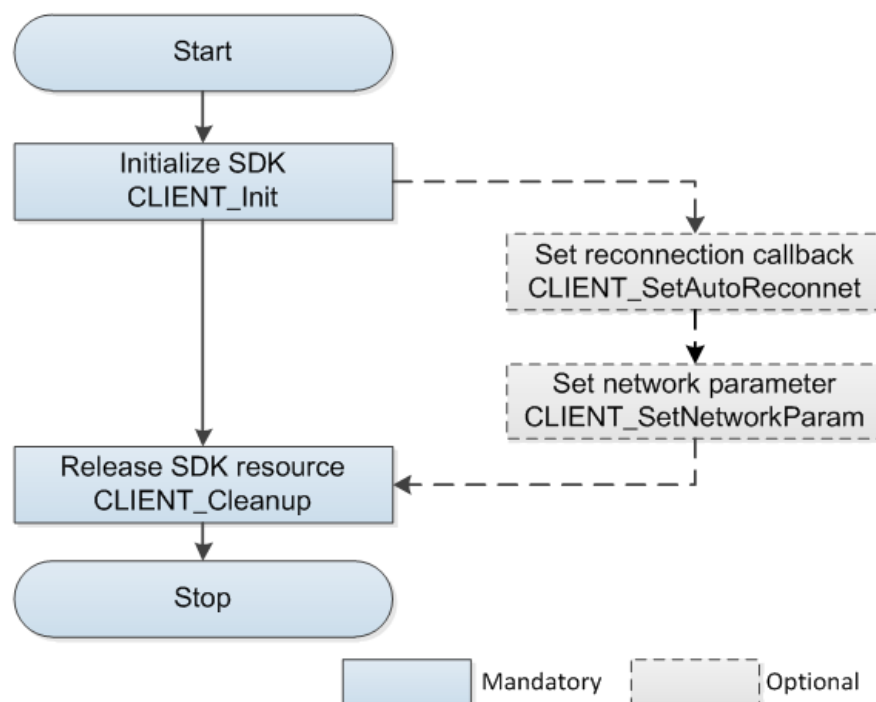
2.1.2 Interface Overview

Table 2-1 Interfaces of SDK initialization

Interface	Implication
CLIENT_Init	SDK initialization.
CLIENT_Cleanup	SDK cleaning up.
CLIENT_SetAutoReconnect	Setting of reconnection after disconnection.
CLIENT_SetNetworkParam	Setting of network environment.

2.1.3 Process

Figure 2-1 Process of SDK initialization



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 (Optional) Call **CLIENT_SetAutoReconnect** to set reconnection callback to allow the auto reconnecting after disconnection.
- Step 3 (Optional) Call **CLIENT_SetNetworkParam** to set network login parameter that includes connection timeout and connection attempts.
- Step 4 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Call **CLIENT_Init** and **CLIENT_Cleanup** in pairs. It supports multiple calling but it is suggested to call the pair for only one time overall.
- Initialization: Calling **CLIENT_Init** multiple times is only for internal count without repeating applying resources.
- Cleaning up: The interface **CLIENT_Cleanup** clears all the opened processes, such as login, real-time monitoring, and alarm subscription.
- Reconnection: SDK can set the reconnection function for the situations such as network disconnection and power off. SDK will keep logging until succeeded. Only the real-time monitoring, alarm and snapshot subscription can be resumed after reconnection is successful.

2.1.4 Example Code

// Set this callback through CLIENT_Init. When the device is disconnected, SDK informs the user through this callback.

```
void CALLBACK DisConnectFunc(LONG lLoginID, char *pchDVRIP, LONG nDVRPort, DWORD dwUser)
{
    printf("Call DisConnectFunc: lLoginID[0x%x]\n", lLoginID);
}

// Initialize SDK
BOOL bNetSDKInitFlag = CLIENT_Init(DisConnectFunc, 0);
if (FALSE == bNetSDKInitFlag)
{
    printf("Initialize client SDK fail; \n");
    return -1;
}

// Clean up the SDK resource
if (TRUE == bNetSDKInitFlag)
{
    CLIENT_Cleanup();
}
```

2.2 Device Login

2.2.1 Introduction

Device login, also called user authentication, is the precondition of all the other function modules.

You will obtain a unique login ID upon logging in to the device and should call login ID before using other SDK interfaces. The login ID becomes invalid once logged out.

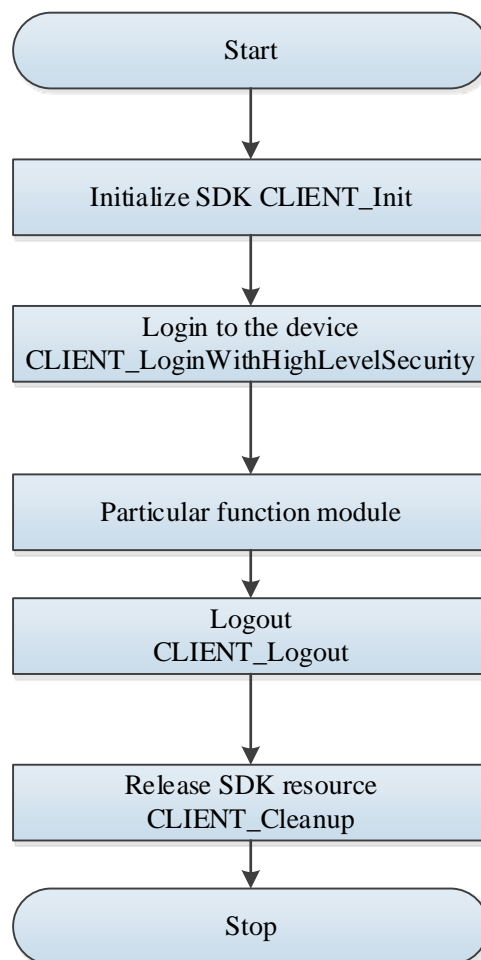
2.2.2 Interface Overview

Table 2-2 Interfaces of device login

Interface	Implication
CLIENT_LoginWithHighLevelSecurity	Log in to the device with high level security. CLIENT_LoginEx2 can still be used, but there are security risks, so it is highly recommended to use the interface CLIENT_LoginWithHighLevelSecurity to log in to the device.
CLIENT_Logout	Logout.

2.2.3 Process

Figure 2-2 Proces of login



Process Description

- Step 1** Call **CLIENT_Init** to initialize SDK.
- Step 2** Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3** After successful login, you can realize the required function module.
- Step 4** After using the function module, call **CLIENT_Logout** to logout the device.
- Step 5** After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Login handle: When the login is successful, the returned value is not 0 (even the handle is smaller than 0, the login is also successful). One device can login multiple times with different handle at each login. If there is not special function module, it is suggested to login only one time. The login handle can be repeatedly used on other function modules.
- Logout: The interface will release the opened functions internally, but it is not suggested to rely on the cleaning up function. For example, if you opened the monitoring function, you should call the interface that stops the monitoring function when it is no longer required.
- Use login and logout in pairs: The login consumes some memory and socket information and release sources once logout.
- Login failure: It is suggested to check the failure through the error parameter of the login interface.

Table 2-3 Common error code

Error code	Meaning
1	Wrong password.
2	The user name does not exist.
3	Login timeout.
4	The account has logged in.
5	The account has been locked.
6	The account is in the blocklist..
7	The device resource is insufficient and the system is busy.
8	Sub connection failed.
9	Main connection failed.
10	Exceeds the maximum allowed number of user connections.
11	Lack avnetsdk or the dependent libraries of avnetsdk.
12	USB flash disk is not inserted into device, or the USB flash disk information error.
13	The IP at client is not authorized for login.

For more information about error codes, see " CLIENT_LoginWithHighLevelSecurity interface" in Network SDK Development Manual.chm. When the network is poor and this make the error code 3 occur easily, then you can use the following codes to increase timeout time:

```
NET_PARAM stuNetParam = {0};
stuNetParam.nWaittime = 8000; // unit ms
CLIENT_SetNetworkParam (&stuNetParam);
```

2.2.4 Example Code

```
NET_IN_LOGIN_WITH_HIGHLEVEL_SECURITY stInparam;
memset(&stInparam, 0, sizeof(stInparam));
stInparam.dwSize = sizeof(stInparam);
strncpy(stInparam.szIP, "192.168.1.108", sizeof(stInparam.szIP) - 1);
strncpy(stInparam.szPassword, "123456", sizeof(stInparam.szPassword) - 1);
strncpy(stInparam.szUserName, "admin", sizeof(stInparam.szUserName) - 1);
stInparam.nPort = 37777;
stInparam.emSpecCap = EM_LOGIN_SPEC_CAP_TCP;

NET_OUT_LOGIN_WITH_HIGHLEVEL_SECURITY stOutparam;
memset(&stOutparam, 0, sizeof(stOutparam));
stOutparam.dwSize = sizeof(stOutparam);
LLONG ILoginID = CLIENT_LoginWithHighLevelSecurity(&stInparam, &stOutparam);
```

2.3 Real-time Monitoring

2.3.1 Introduction

Real-time monitoring obtains the real-time stream from the storage device or front-end device, which is an important part of the surveillance system.

SDK can get the main stream and sub stream from the device once it logged.

- Supports calling the window handle for SDK to directly decode and play the stream (Windows system only).
- Supports calling the real-time stream for you to perform independent treatment.
- Supports saving the real-time record to the specific file though saving the callback stream or calling the SDK interface.

2.3.2 Interface Overview

Table 2-4 Interfaces of real-time monitoring

Interface	Implication
CLIENT_RealPlayEx	Start real-time monitoring.
CLIENT_StopRealPlayEx	Stop real-time monitoring.
CLIENT_SaveRealData	Start saving the real-time monitoring data to the local path.
CLIENT_StopSaveRealData	Stop saving the real-time monitoring data to the local path.
CLIENT_SetRealDataCallBackEx	Set real-time monitoring data callback.

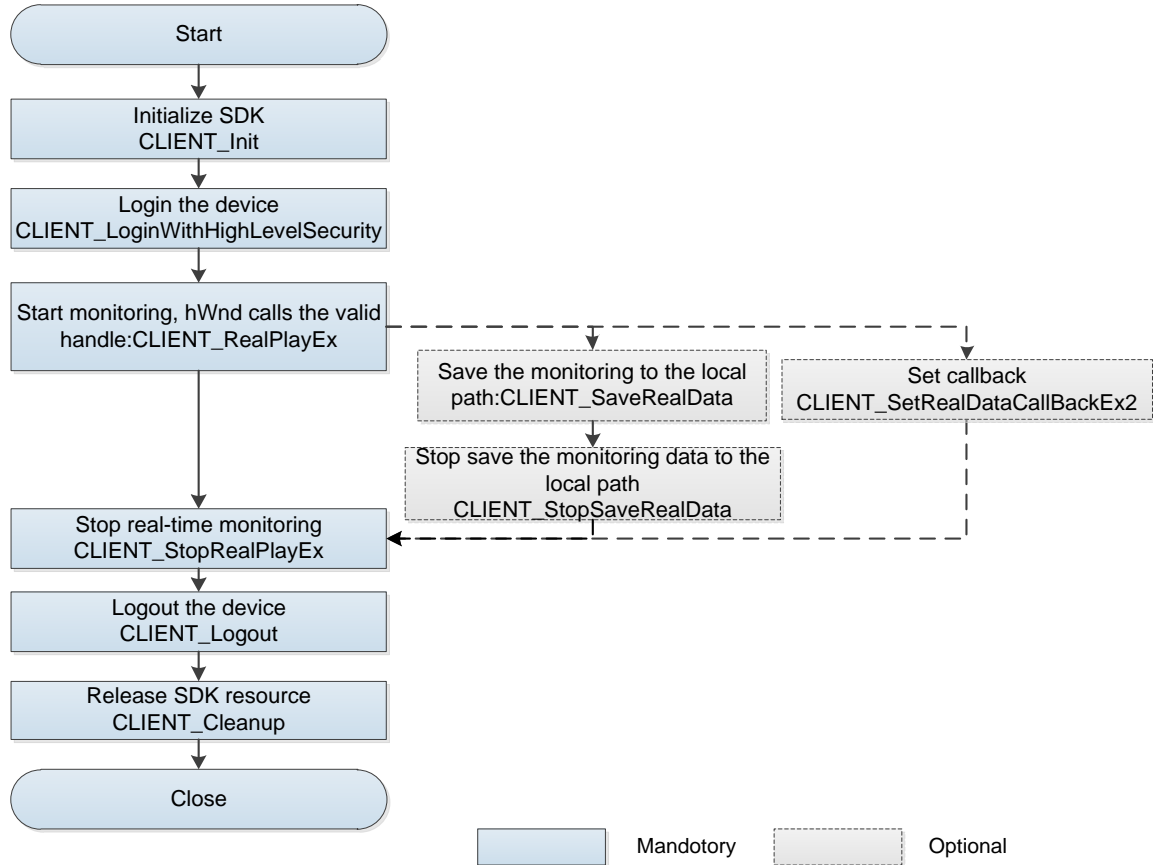
2.3.3 Process

You can realize the real-time monitoring through SDK decoding library or your play library.

2.3.3.1 SDK Decoding Play

Call PlaySDK library from the SDK auxiliary library to realize real-time play.

Figure 2-3 Process of playing by SDK decoding library



Process Description

- Step 1** Call **CLIENT_Init** to initialize SDK.
- Step 2** Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3** Call **CLIENT_RealPlayEx** to enable the real-time monitoring. The parameter hWnd is a valid window handle.
- Step 4** (Optional) Call **CLIENT_SaveRealData** to start saving the monitoring data.
- Step 5** (Optional) Call **CLIENT_StopSaveRealData** to end the saving process and generate the local video file.
- Step 6** (Optional) If you call **CLIENT_SetRealDataCallBackEx2**, you can choose to save or forward the video file. If save the video file, see the step 4 and step 5.
- Step 7** After completing the real-time monitoring, call **CLIENT_StopRealPlayEx** to stop real-time monitoring.
- Step 8** After using the function module, call **CLIENT_Logout** to logout the device.
- Step 9** After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- SDK decoding play only supports Windows system. You need to call the decoding after getting the stream in other systems.
- Multi-thread calling: Multi-thread calling is not supported for the functions within the same login session; however, multi-thread calling can deal with the functions of different login sessions although such calling is not recommended.
- Timeout: The request on applying for monitoring resources should have made some agreement with the device before requiring the monitoring data. There are some timeout settings (see "NET_PARAM structure"), and the field about monitoring is nGetConnInfoTime. If there is timeout due to the reasons such as bad network connection, you can modify the value of nGetConnInfoTime bigger. The example code is as follows. Call it for only one time after having called CLIENT_Init.

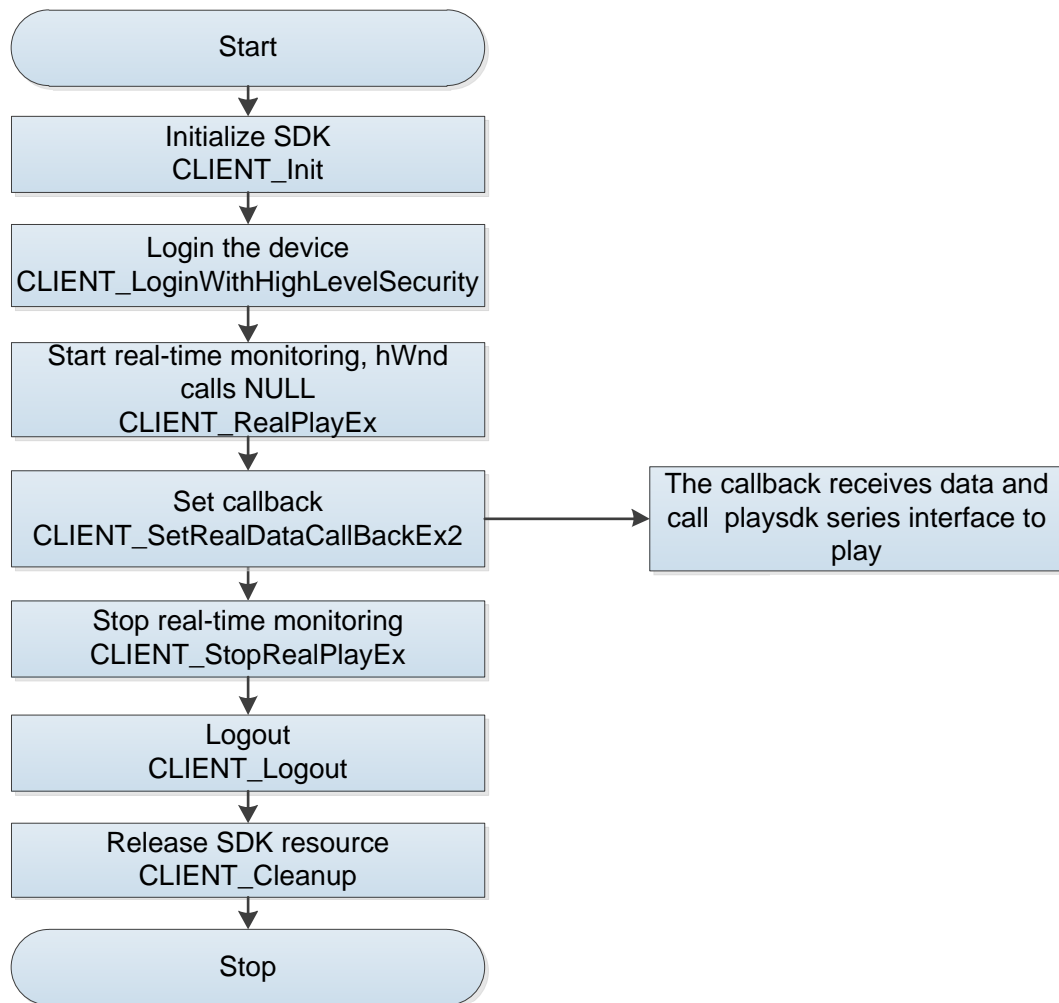
```
NET_PARAM stuNetParam = {0};  
stuNetParam.nGetConnInfoTime = 5000; // unit ms  
CLIENT_SetNetworkParam (&stuNetParam);
```

- Failed to repeat opening: For some models, the same channel cannot be opened for multiple times during a login. If you are trying to open it repeatedly, you will success in the first try but get failed afterwards. In this case, you can try the following:
 - ◇ Close the opened channel. For example, if you have already opened the main stream video on the channel 1 and still want to open the sub stream video on the same channel, you can close the main stream first and then open the sub stream.
 - ◇ Login twice to obtain two login handles to deal with the main stream and sub stream respectively.
- Calling succeeded but no image: SDK decoding needs to use dhplay.dll. It is suggested to check if dhplay.dll and its auxiliary library are missing under the running directory. See Table 1-1.
- If the system resource is insufficient, the device might return error instead of stream. You can receive an event DH_REALPLAY_FAILED_EVENT in the alarm callback that is set in CLIENT_SetDVRMessCallBack. This event includes the detailed error codes. See "DEV_PLAY_RESULT Structure" in Network SDK Development Manual.chm.
- 32 channels limit: The decoding consumes resources especially for the high definition videos. Considering the limited resources at the client, currently the maximum channels are set to be 32. If more than 32, it is suggested to use third party play library. See "2.3.3.2 Call Third Party Library".

2.3.3.2 Call Third Party Library

SDK calls back the real-time monitoring stream to you and you call PlaySDK to decode and play.

Figure 2-4 Process of calling the third party library



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 After successful login, call **CLIENT_RealPlayEx** to enable real-time monitoring. The parameter hWnd is NULL.
- Step 4 Call **CLIENT_SetRealDataCallBackEx** to set the real-time data callback.
- Step 5 In the callback, pass the data to PlaySDK to finish decoding.
- Step 6 After completing the real-time monitoring, call **CLIENT_StopRealPlayEx** to stop real-time monitoring.
- Step 7 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 8 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Stream format: It is recommended to use PlaySDK for decoding.
- Lag image
 - ◇ When using PlaySDK for decoding, there is a default channel cache size (the PLAY_OpenStream interface in playsdk) for decoding. If the stream resolution value is big, it is recommended to modify the parameter value smaller such as 3 M.

- ◇ SDK callbacks can only moves into the next process after returning from you. It is not recommended for you to consume time for the unnecessary operations; otherwise the performance could be affected.

2.3.4 Example Code

2.3.4.1 SDK Decoding Play

```
// Take opening the main stream monitoring of channel 1 as an example. The parameter hWnd is a handle of
interface window.
LLONG IRealHandle = CLIENT_RealPlayEx(ILoginHandle, 0, hWnd, DH_RType_Realplay);
if (NULL == IRealHandle)
{
    printf("CLIENT_RealPlayEx: failed! Error code: %x.\n", CLIENT_GetLastError());
}
printf("input any key to quit!\n");
getchar();
// Stop preview
if (NULL != IRealHandle)
{
    CLIENT_StopRealPlayEx(IRealHandle);
}
```

2.3.4.2 Call Third Party Library

```
Take opening the main stream monitoring of channel 1 as an example.
LLONG IRealHandle = CLIENT_RealPlayEx(ILoginHandle, 0, NULL, DH_RType_Realplay);
if (NULL == IRealHandle)
{
    printf("CLIENT_RealPlayEx: failed! Error code: %x.\n", CLIENT_GetLastError());
}
else
{
    DWORD dwFlag = 0x00000001;
    CLIENT_SetRealDataCallBackEx(IRealHandle, &RealDataCallBackEx, NULL, dwFlag);
}
// Stop preview
if (0 != IRealHandle)
{
    CLIENT_StopRealPlayEx(IRealHandle);
}
```

```

}

void CALLBACK RealDataCallBackEx(LLONG IRealHandle, DWORD dwDataType, BYTE *pBuffer, DWORD
dwBufSize, LONG param, LDWORD dwUser)
{
// Call PlaySDK interface to get the stream data from the device. See SDK monitoring demo source data for
more details.
    printf("receive real data, param: IRealHandle[%p], dwDataType[%d], pBuffer[%p], dwBufSize[%d]\n",
IRealHandle, dwDataType, pBuffer, dwBufSize);
}

```

2.4 Subscribing to Intelligent Event

2.4.1 Introduction

Intelligent event subscribe, is that the front-end devices or the back-end devices do the real-time stream analyzing. When detect the preset intelligent event, it uploads the event to the user. The intelligent events in this manual contain general action analysis (such as tripwire, Intrusion), target detection, object recognition, body detection, the intelligent events of intelligent traffic (such as traffic junction, over speed, low speed and traffic jam).

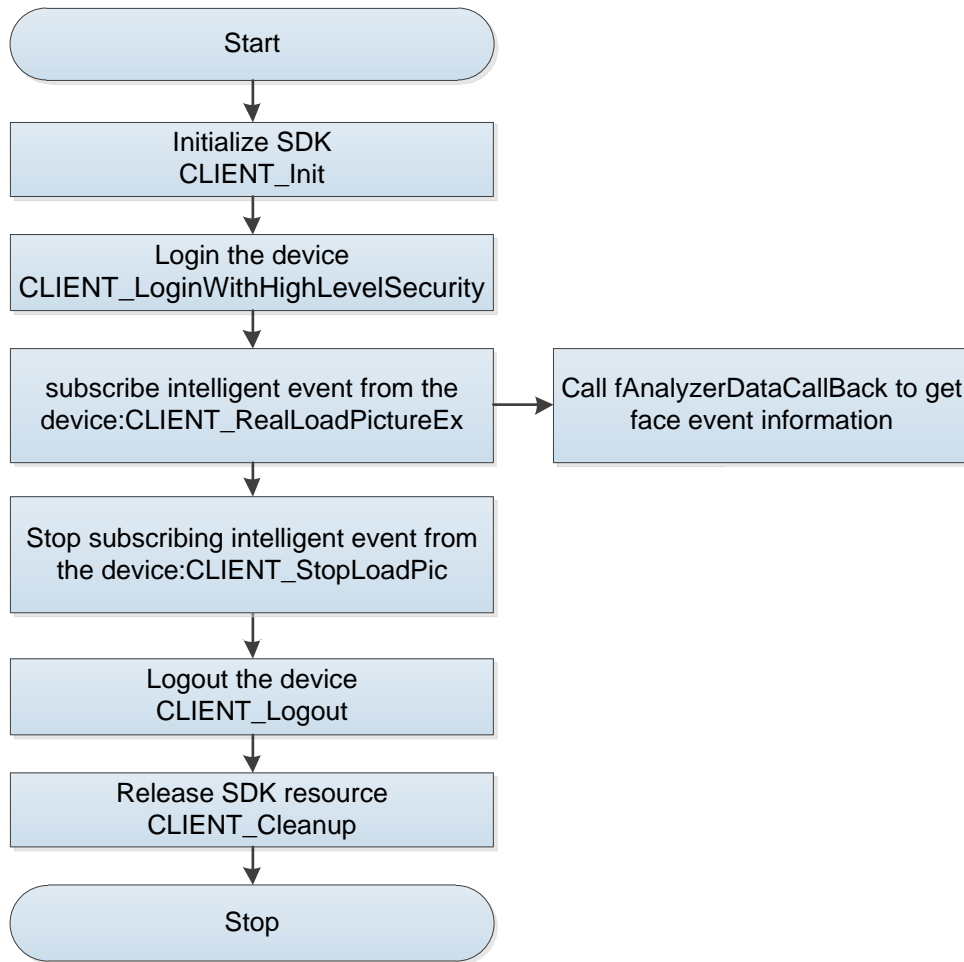
2.4.2 Interface Overview

Table 2-5 Interfaces of subscribing to intelligent event

Interface	Implication
CLIENT_RealLoadPictureEx	Subscribe intelligent event.
CLIENT_StopLoadPic	Cancel subscribing the intelligent event.

2.4.3 Process

Figure 2-5 Process of uploading face event



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.
- Step 3 Call **CLIENT_RealLoadPictureEx** to subscribe intelligent event from the device.
- Step 4 After successful subscribe, call fAnalyzerDataCallBack to upload the intelligent events. Through this function, you can filter out the intelligent events you need.
- Step 5 After using the intelligent event function, call **CLIENT_StopLoadPic** to stop subscribing intelligent events.
- Step 6 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 7 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Support to subscribe single intelligent event and all the intelligent events (EVENT_IVS_ALL).
- Setting of cache for receiving pictures: Because SDK default cache is 2M, when the data is over 2M, call CLIENT_SetNetworkParam to set the receiving cache; otherwise the data pack will be lost.

- Set whether to receive picture or not: You can call `CLIENT_RealLoadPictureEx` to set `bNeedPicFile` as `False`, and then SDK will only receive the face event without picture.

2.4.4 Example Code

```
// Intelligent event uploading callback function
int CALLBACK AnalyzerDataCallBack(LLONG IAnalyzerHandle, DWORD dwAlarmType, void* pAlarmInfo, BYTE
*pBuffer, DWORD dwBufSize, LDWORD dwUser, int nSequence, void *reserved)
{
    switch(dwAlarmType)
    {
        // Filter out the right intelligent events
        .....
        case EVENT_IVS_FACERECOGNITION: // Object recognition events
        .....
        default:
        break;
    }
}

// Subscribe the uploading of the intelligent event
LLONG IAnalyzerHandle = CLIENT_RealLoadPictureEx(ILoginHandle, 0, (DWORD)EVENT_IVS_ALL, TRUE,
AnalyzerDataCallBack, NULL, NULL);
if(NULL == IAnalyzerHandle)
{
    printf("CLIENT_RealLoadPictureEx: failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}

// Cancel Subscribing the uploading of the intelligent event
CLIENT_StopLoadPic(IAnalyzerHandle);
```

2.5 Searching for/Playingback/Downloading Video and Picture

2.5.1 Introduction

When the device intelligent calculation analysis the real-time stream, once one intelligent event is detected, and then the video and picture of this intelligent event will be saved. You can search the

video and picture of the intelligent events which are saved in the device, and also you can do the downloading and playing back operation to the searching result.

2.5.2 Interface Overview

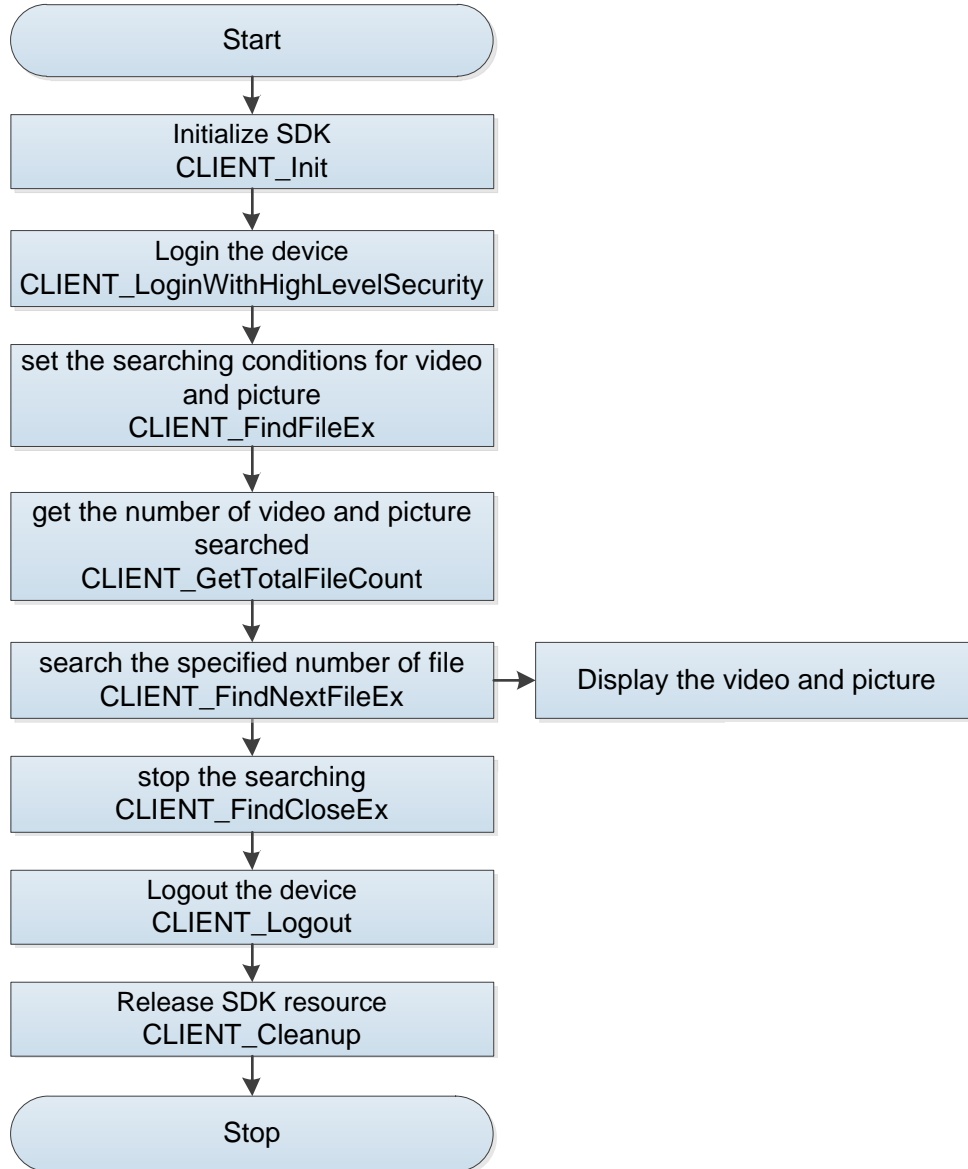
Table 2-6 Interfaces of searching/playvacking/downloading video and picture

Interface	Implication
CLIENT_FindFileEx	Search the video and picture by conditions, and set the searching conditions.
CLIENT_GetTotalFileCount	Obtain the number of video and picture searched now.
CLIENT_FindNextFileEx	Search the specified number of video and picture.
CLIENT_FindCloseEx	Stop searching.
CLIENT_PlayBackByTimeEx2	Start playing back the video by time.
CLIENT_StopPlayBack	Stop playing back the video.
CLIENT_DownloadByTimeEx	Download video.
CLIENT_StopDownload	Stop downloading the video.
CLIENT_DownloadRemoteFile	Download pictures.

2.5.3 Process

2.5.3.1 Searching Process of Video and Picture

Figure 2-6 Searching process of video and picture



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_FindFileEx** to set the searching conditions. After successfully setting, return the searching handle. To judge the right searching type according to the different values of emType.
- Step 4 Call **CLIENT_GetTotalFileCount** to get the total number of video and picture searched.
- Step 5 Call **CLIENT_FindNextFileEx** to search the specified number of video and picture. Save the video and picture and do the playing back and downloading operation to the video and picture.

- Step 6** Call **CLIENT_FindCloseEx** to stop the searching.
- Step 7** After using the function module, call **CLIENT_Logout** to logout the device.
- Step 8** After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

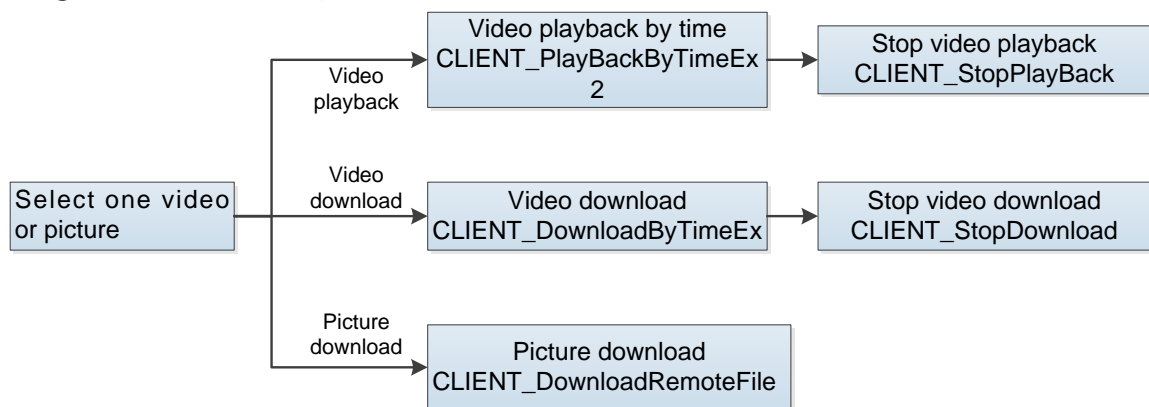
Notes for Process

- The parameter pQueryCondition of **CLIENT_FindFileEx** is requested and released by the user. The specific type is defined by the enumeration type of emType.
- If **CLIENT_FindFileEx** successfully search, the searching handle will be returned. **CLIENT_FindNextFileEx** will take the searching handle as a parameter to search specific video and picture. You should call **CLIENT_FindCloseEx** to close the searching handle.
- Call **CLIENT_FindNextFileEx** to set the searching number. If the number is more than 1, then the parameter pMediaFileInfo should be taken as a data pointer.

2.5.3.2 Process of Playing Back and Downloading Video and Downloading Picture

Picture

Figure 2-7 Process of playing back and downloading video and downloading picture



Process Description

Select one result searched by **CLIENT_FindNextFileEx**, and then download or playback the result.

- Playing back the video

Step 1 If is video file, use start time and end time in the video searching result, call **CLIENT_PlayBackByTimeEx2** to playback the video.

Step 2 During the playback process or after playing back, call **CLIENT_StopPlayBack** to stop playing back the video.

- Download video

Step 1 If is video file, use start time and end time in the video searching result, call **CLIENT_DownloadByTimeEx** to download the video.

Step 2 After downloading, call **CLIENT_StopDownload** to stop downloading the video.

- Download the picture

If is picture file, use file name and picture type in the picture searching result, call **CLIENT_DownloadRemoteFile** to download the picture.

Notes for Process

The video playing back and downloading and picture downloading are all relied on the searching result of video and picture, and then you can take the result as the condition of playing back and downloading.

2.5.4 Example Code

2.5.4.1 Searching for Video and Picture

```
// Searching conditions
MEDIAFILE_FACE_DETECTION_PARAM param;
memset(&param, 0, sizeof(param));
param.dwSize = sizeof(param);
param.stuDetail.dwSize = sizeof(MEDIAFILE_FACE_DETECTION_DETAIL_PARAM);
param.nChannelID = -1;
param.stuStartTime = startTime;
param.stuEndTime = endTime
param.emPicType = NET_FACEPIC_TYPE_SMALL; // The small picture of people face.
param.bDetailEnable = FALSE;
param.emSex = EM_DEV_EVENT_FACEDETECT_SEX_TYPE_MAN;
param.bAgeEnable = FALSE;
param.nEmotionValidNum = 0;
param.emGlasses = EM_FACEDETECT_WITH_GLASSES;

// Search the small picture of target detection
LLONG IFindFileHandle = CLIENT_FindFileEx(g_ILoginHandle, DH_FILE_QUERY_FACE_DETECTION, &param,
NULL,5000);
if (IFindFileHandle == 0)
{
    printf("CLIENT_FindFileEx: failed! Error code: %x.\n", CLIENT_GetLastError());
    return;
}

// Get the number of people face that searched
BOOL nRet = CLIENT_GetTotalFileCount(IFindFileHandle,&nCount,NULL);
if (!nRet)
{
    printf("CLIENT_GetTotalFileCount: failed! Error code: %x.\n", CLIENT_GetLastError());
    return;
}
```

```

}

// Searching number
int nMaxConut = 10;
MEDIAFILE_FACE_DETECTION_INFO* pMediaFileInfo = NEW MEDIAFILE_FACE_DETECTION_INFO[nMaxConut];
memset (pMediaFileInfo, 0, sizeof (MEDIAFILE_FACE_DETECTION_INFO) * nMaxConut);
for (int i = 0; i < nMaxConut; i++)
{
    pMediaFileInfo[i].dwSize = sizeof(MEDIAFILE_FACE_DETECTION_INFO);
}

// Start searching
int nRet = CLIENT_FindNextFileEx(IFindFileHandle, nMaxConut, (void*)pMediaFileInfo, nMaxConut *
sizeof(MEDIAFILE_FACE_DETECTION_INFO), NULL, 3000);
if (nRet < 0)
{
    printf("CLIENT_FindNextFileEx: failed! Error code: %x.\n", CLIENT_GetLastError());
    return;
}

Close searching
CLIENT_FindCloseEx(IFindFileHandle);

```

2.5.4.2 Playing Back the Video

```

// Set the stream type when the video is playing back, here set it as the main stream
int nStreamType = 0; // 0-main and sub stream, 1-main stream, 2-sub stream
CLIENT_SetDeviceMode(ILoginHandle, DH_RECORD_STREAM_TYPE, &nStreamType);
// Set the file type of the video when playing back, here set it as all video.
NET_RECORD_TYPE emFileType = NET_RECORD_TYPE_ALL; // All video
CLIENT_SetDeviceMode(ILoginHandle, DH_RECORD_TYPE, &emFileType);
// Start playing back the video
int nChannelID = 0; // Channel number.
NET_IN_PLAY_BACK_BY_TIME_INFO stIn = {0};
NET_OUT_PLAY_BACK_BY_TIME_INFO stOut = {0};
memcpy(&stIn.stStartTime, &stuStartTime, sizeof(stuStartTime));
memcpy(&stIn.stStopTime, &stuStopTime, sizeof(stuStopTime));
stIn.hWnd = hWnd;
stIn.fDownloadDataCallBack = DataCallBack;
stIn.dwDataUser = NULL;
stIn.cbDownloadPos = NULL;
stIn.dwPosUser = NULL;

```

```

stIn.nPlayDirection = emDirection;
stIn.nWaittime = 10000;
LLONG IPlayHandle = CLIENT_PlayBackByTimeEx2(ILoginHandle, nChannelID, &stIn, &stOut);
if (0 == IPlayHandle)
{
    printf("CLIENT_PlayBackByTimeEx2: failed! Error code: %x.\n", CLIENT_GetLastError());
}

if (FALSE == CLIENT_StopPlayBack(IPlayHandle))
{
    printf("CLIENT_StopPlayBack Failed, IRealHandle[%x]!Last Error[%x]\n", IPlayHandle,
        CLIENT_GetLastError());
}

```

2.5.4.3 Downloading Video

```

// Playback progress function
void CALLBACK TimeDownLoadPosCallBack(LLONG IPlayHandle, DWORD dwTotalSize, DWORD
dwDownLoadSize, int index, NET_RECORDFILE_INFO recordfileinfo, LDWORD dwUser);

// Playback or download data callback function
int CALLBACK DataCallBack(LLONG IRealHandle, DWORD dwDataType, BYTE *pBuffer, DWORD
dwBufSize, LDWORD dwUser);

int main()
{
    // Set the video stream type when searching, here set it as the main and sub stream.
    int nStreamType = 0; // 0-main and sub stream, 1-main stream, 2-sub stream
    CLIENT_SetDeviceMode(ILoginHandle, DH_RECORD_STREAM_TYPE, &nStreamType);

    // Set the downloading start and end time
    int nChannelID = 0; // Channel number.

    NET_TIME stuStartTime = {0};
    stuStartTime.dwYear = 2018;
    stuStartTime.dwMonth = 9;
    stuStartTime.dwDay = 17;

    NET_TIME stuStopTime = {0};
    stuStopTime.dwYear = 2018;

```

```

    stuStopTime.dwMonth = 9;
    stuStopTime.dwDay = 18;

    Start downloading the video.

    // One of the formal parameters sSavedFileName and fDownLoadDataCallBack should be valid, otherwise
the input parameter is wrong.
    IDownloadHandle = CLIENT_DownloadByTimeEx(ILoginHandle, nChannelID, EM_RECORD_TYPE_ALL,
&stuStartTime, &stuStopTime, "test.dav", TimeDownLoadPosCallBack, NULL, DataCallBack, NULL);
    if (IDownloadHandle == 0)
    {
        printf("CLIENT_DownloadByTimeEx: failed! Error code: %x.\n", CLIENT_GetLastError());
    }

    // Close downloading. Call the function after or during the downloading.
    if (0 != IDownloadHandle)
    {
        if (!CLIENT_StopDownload(IDownloadHandle))
        {
            printf("CLIENT_StopDownload Failed, IDownloadHandle[%x]!Last Error[%x]\n",
IDownloadHandle, CLIENT_GetLastError());
        }
    }
}

void CALLBACK TimeDownLoadPosCallBack(LLONG IPlayHandle, DWORD dwTotalSize, DWORD
dwDownLoadSize, int index, NET_RECORDFILE_INFO recordfileinfo, LDWORD dwUser)
{
    // You can deal with the progress callback function.
}

int CALLBACK DataCallBack(LLONG IRealHandle, DWORD dwDataType, BYTE *pBuffer, DWORD
dwBufSize, LDWORD dwUser)
{
    switch(dwDataType)
    {
        case 0:
            // Original data
            // You can save the stream data here. After leaving callback function, do the decoding and
forwarding and so on.
            break;
    }
}

```

```

    case 1://Standard video data
        break;
    case 2:  //yuv data
        break;
    case 3://pcm audio data
        break;
    default:
        break;
}
return 0;
}

```

2.5.4.4 Downloading the Picture

```

DH_IN_DOWNLOAD_REMOTE_FILE stuRemoteFileParm;
memset(&stuRemoteFileParm, 0, sizeof(DH_IN_DOWNLOAD_REMOTE_FILE));
stuRemoteFileParm.dwSize = sizeof(DH_IN_DOWNLOAD_REMOTE_FILE);
stuRemoteFileParm.pszFileName = pInfo->stObjectPic.szFilePath ;
stuRemoteFileParm.pszFileDst = szFileName;

DH_OUT_DOWNLOAD_REMOTE_FILE *fileinfo = NEW DH_OUT_DOWNLOAD_REMOTE_FILE;
fileinfo->dwSize = sizeof(DH_OUT_DOWNLOAD_REMOTE_FILE);

if (!CLIENT_DownloadRemoteFile(g_LoginHandle, &stuRemoteFileParm, fileinfo))
{
    printf("CLIENT_DownloadRemoteFile Failed,Last Error[%x]\n", CLIENT_GetLastError());
}

```

3 Target Detection and Recognition

3.1 Subscribing Face Event

About more details, see "2.4 Subscribing to Intelligent Event". Call fAnalyzerDataCallBack to filter out target detection and recognition events, which are EVENT_IVS_FACEDETECT for people target detection events and EVENT_IVS_FACERECOGNITION for people object recognition events.

3.2 Adding/Deleting/Modifying/Searching the Face Library

3.2.1 Introduction

A face library includes face picture and face information, supports the adding, deleting, modifying and searching the face library function.

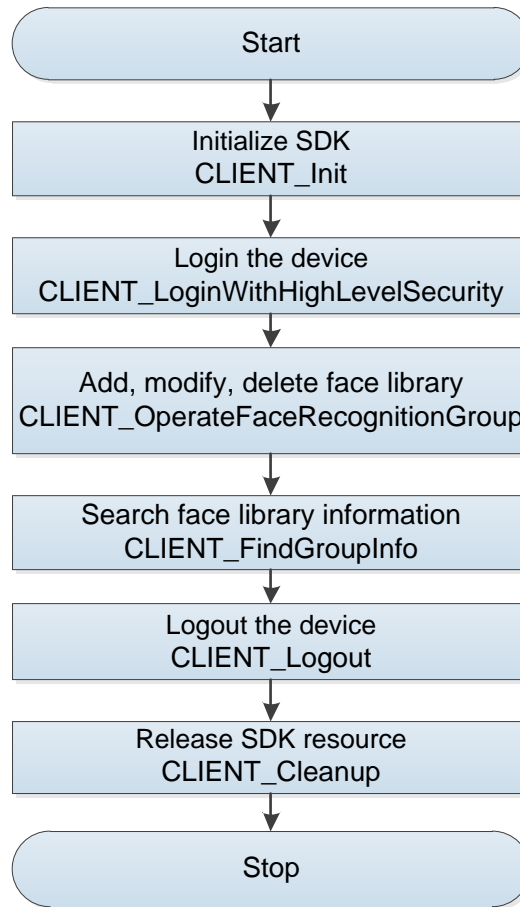
3.2.2 Interface Overview

Table 3-1 Interfaces of adding/deleting/modifying/searching the face library

Interface	Implication
CLIENT_OperateFaceRecognitionGroup	Add, delete and modify the face library.
CLIENT_FindGroupInfo	Search the information of face library.

3.2.3 Process

Figure 3-1 Process of face library operation



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_OperateFaceRecognitionGroup** to add, modify and delete the face library according to enumeration type.
- Step 4 Call **CLIENT_FindGroupInfo** to get the information of face library.
- Step 5 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 6 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Adding face library: The corresponding value of operation type `emOperateType` is `NET_FACERECONGNITION_GROUP_ADD`, the corresponding structure is `NET_ADD_FACERECONGNITION_GROUP_INFO`.
- Modify face library: The corresponding value of operation type `emOperateType` is `NET_FACERECONGNITION_GROUP_MODIFY`, the corresponding structure is `NET_MODIFY_FACERECONGNITION_GROUP_INFO`. You need to specify `GroupID` and the face library type `emFaceDBType` when modifying face library. The specified `GroupID` should be exists in the device.

- Deleting face library: The corresponding value of operation type emOperateType is NET_FACERECONGNITION_GROUP_DELETE, the corresponding structure is NET_DELETE_FACERECONGNITION_GROUP_INFO. If you specify GroupID when deleting face library, the corresponding GroupID of face library is deleted; If you do not specify GroupID, all of the face libraries are deleted.

3.2.4 Example Code

3.2.4.1 Searching for Face Library Information

```
// Set the searching conditions of face library
NET_IN_FIND_GROUP_INFO stuInParam = {sizeof(stuInParam)};
NET_OUT_FIND_GROUP_INFO stuOutParam = {sizeof(stuOutParam)};
stuOutParam.nMaxGroupNum = 100;
NET_FACERECONGNITION_GROUP_INFO *pGroupInfo = NULL;
stuOutParam.pGroupInfos = new NET_FACERECONGNITION_GROUP_INFO[100];
memset(stuOutParam.pGroupInfos, 0, sizeof(NET_FACERECONGNITION_GROUP_INFO)*100);
for (int i = 0; i < 100; i++)
{
    stuOutParam.pGroupInfos[i].dwSize = sizeof (FACERECONGNITION_GROUP_INFO);
}
// Search the face library
BOOL bRet = CLIENT_FindGroupInfo(ILLoginHandle, &stuInParam, &stuOutParam, 5000);
if(FALSE == bRet)
{
    printf("CLIENT_FindGroupInfo: failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}
delete[] pGroupInfo;
```

3.2.4.2 Adding/Deleting/Modifying the Face Library

```
enum EM_OPERATION_TYPE
{
    FACEDB_DELETE, // Delete
    FACEDB_ADD,    // Add
    FACEDB_MODIFY  // Modify
};
// Set the face library ID for deleting.
NET_FACERECONGNITION_GROUP_INFO *pstGroupInfo = m_pstSelectGroup;
NET_IN_OPERATE_FACERECONGNITION_GROUP stuInParam = {sizeof(stuInParam)};
```



```

NET_OUT_OPERATE_FACERECONGNITION_GROUP stuOutParam = {sizeof(stuOutParam)};
NET_ADD_FACERECONGNITION_GROUP_INFO stuAddGroupInfo = {sizeof(stuAddGroupInfo)};
NET_MODIFY_FACERECONGNITION_GROUP_INFO stuEditGroupInfo = {sizeof(stuEditGroupInfo)};

EM_OPERATION_TYPE emType = mType;
switch(emType)
{
    // Delete the face library
    case FACEDB_DELETE:
    {
        stuInParam.emOperateType = NET_FACERECONGNITION_GROUP_DELETE;
        NET_DELETE_FACERECONGNITION_GROUP_INFO stuDeleteInfo;
        memset(&stuDeleteInfo, 0, sizeof(stuDeleteInfo));
        stuDeleteInfo.dwSize = {sizeof(stuDeleteInfo)};
        strncpy(stuDeleteInfo.szGroupId, pstGroupInfo->szGroupId, sizeof(stuDeleteInfo.szGroupId)-1);
        stuInParam.pOperateInfo = &stuDeleteInfo;
        break;
    }
    // Add the face library
    case FACEDB_ADD:
    {
        stuInParam.emOperateType = NET_FACERECONGNITION_GROUP_ADD;
        stuAddGroupInfo.stuGroupInfo.dwSize = sizeof(stuAddGroupInfo.stuGroupInfo);
        stuAddGroupInfo.stuGroupInfo.emFaceDBType = NET_FACE_DB_TYPE_BLACKLIST;

        strncpy(stuAddGroupInfo.stuGroupInfo.szGroupName, pcGroupName, sizeof(stuAddGroupInfo.stuGroupInfo.szGroupName)-1);
        stuInParam.pOperateInfo = &stuAddGroupInfo;
        break;
    }
    // Modifying the face library
    case FACEDB_MODIFY:
    {
        stuInParam.emOperateType = NET_FACERECONGNITION_GROUP_MODIFY;
        stuEditGroupInfo.stuGroupInfo.dwSize = sizeof(stuEditGroupInfo.stuGroupInfo);
        stuEditGroupInfo.stuGroupInfo.emFaceDBType = NET_FACE_DB_TYPE_BLACKLIST;
        strncpy(stuEditGroupInfo.stuGroupInfo.szGroupName, pcGroupName,
        sizeof(stuEditGroupInfo.stuGroupInfo.szGroupName)-1);
        strncpy(stuEditGroupInfo.stuGroupInfo.szGroupId, m_stuGroupInfo.szGroupId,
        sizeof(stuEditGroupInfo.stuGroupInfo.szGroupId)-1);
    }
}

```

```

        stuInParam.pOperateInfo = &stuEditGroupInfo;
        break;
    }
    default:
        break;
}
BOOL bRet = CLIENT_OperateFaceRecognitionGroup(m_ILoginID, &stuInParam, &stuOutParam, 5000);
if(FALSE == bRet)
{
    printf("CLIENT_OperateFaceRecognition: failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}

```

3.3 Adding/Deleting/Modifying/Searching People Face

3.3.1 Introduction

The face library includes face information. This function supports adding, deleting, modifying and searching people face information.

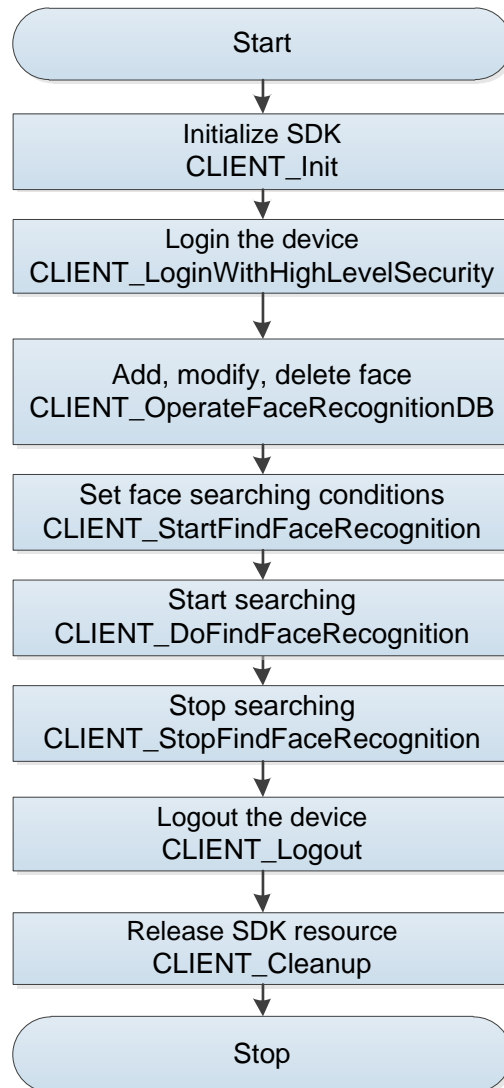
3.3.2 Interface Overview

Table 3-2 Interfaces of adding/deleting/modifying/searching people face

Interface	Implication
CLIENT_OperateFaceRecognitionDB	Add, delete and modify the people face.
CLIENT_StartFindFaceRecognition	Set the searching conditions of people face.
CLIENT_DoFindFaceRecognition	Search the face data of specified number.
CLIENT_StopFindFaceRecognition	Stop searching.

3.3.3 Process

Figure 3-2 Process of adding/deleting/modifying/searching people face



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_OperateFaceRecognitionDB** to add, modify and delete the face library according to enumeration type.
- Step 4 Call **CLIENT_StartFindFaceRecognition** to set the searching conditions of people face.
- Step 5 Call **CLIENT_DoFindFaceRecognition** to get the searching result.
- Step 6 Call **CLIENT_StopFindFaceRecognition** to stop searching.
- Step 7 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 8 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Adding people face to the library: The corresponding value of operation type emOperateType is NET_FACERECONGNITIONDB_ADD. The input parameter in the structure, pBuffer can save

people face you need. You can request and release the resource by yourself. GroupID must be filled-in. If people add successful, the device returns UID, is also called the only people ID, means the only people. See pstOutParam -> szUID.

- Modify the people face in the library: The corresponding value of operation type emOperateType is NET_FACERECONGNITIONDB_MODIFY, The input parameter in the structure pBuffer can save people face you need. You can request and release the resource by yourself. The GroupID and szUID must be filled-in in the stPersonInfo.
- Delete the people face in the library: The corresponding value of operation type emOperateType is NET_FACERECONGNITIONDB_DELETE. The GroupID and szUID must be filled-in in the stPersonInfo.

3.3.4 Example Code

3.3.4.1 Adding/Deleting/Modifying the People Face

```
// Add and modify the people
NET_IN_OPERATE_FACERECONGNITIONDB stuInParam = { sizeof(stuInParam) };
NET_OUT_OPERATE_FACERECONGNITIONDB stuOutParam = { sizeof(stuOutParam) };
// Add the information of people
{
    stuInParam.emOperateType = NET_FACERECONGNITIONDB_ADD;
}

// Modify the information of people
{
    //stuInParam.emOperateType = NET_FACERECONGNITIONDB_MODIFY;
    //strncpy(stuInParam.stPersonInfoEx.szUID, strUID, sizeof(stuInParam.stPersonInfoEx.szUID) - 1);
}

stuInParam.bUsePersonInfoEx = TRUE;
stuInParam.stPersonInfoEx.bySex = 1; // Male
stuInParam.stPersonInfoEx.byIDType = 1; // ID card
stuInParam.stPersonInfoEx.wYear = time.GetYear();
stuInParam.stPersonInfoEx.byMonth = time.GetMonth();
stuInParam.stPersonInfoEx.byDay = time.GetDay();
strncpy(stuInParam.stPersonInfoEx.szPersonName, pstrName, sizeof(stuInParam.stPersonInfoEx.szPersonName)
- 1);
strncpy(stuInParam.stPersonInfoEx.szID, pstrCardID, sizeof(stuInParam.stPersonInfoEx.szID) - 1);
strncpy(stuInParam.stPersonInfoEx.szGroupName, m_szGroupName,
sizeof(stuInParam.stPersonInfoEx.szGroupName) - 1);
strncpy(stuInParam.stPersonInfoEx.szGroupID, m_szGroupID, sizeof(stuInParam.stPersonInfoEx.szGroupID) - 1);
stuInParam.nBufferLen = nPictureBufferLen;
stuInParam.pBuffer = pPictureBuffer;
stuInParam.stPersonInfoEx.wFacePicNum = 1;
stuInParam.stPersonInfoEx.szFacePicInfo[0].dwOffSet = 0;
stuInParam.stPersonInfoEx.szFacePicInfo[0].dwFileLenth = nLength;
```

```

bRet = CLIENT_OperateFaceRecognitionDB(m_lLoginID, &stuInParam, &stuOutParam, 5000);
if (FACE_PERSON_ADD == m_nOpreateType)
{
    printf("CLIENT_OperateFaceRecognitionDB failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}

// Delete the information of people
NET_IN_OPERATE_FACERECONGNITIONDB stuInParam = { sizeof(stuInParam) };
NET_OUT_OPERATE_FACERECONGNITIONDB stuOutParam = { sizeof(stuOutParam) };
// Only need szGroupID and szUID.
stuInParam.emOperateType = NET_FACERECONGNITIONDB_DELETE;
stuInParam.bUsePersonInfoEx = TRUE;
strncpy(stuInParam.stPersonInfoEx.szUID,
m_pstPersonSelectInfo->stuCandidate.stPersonInfo.szUID, sizeof(stuInParam.stPersonInfoEx.szUID) - 1);
strncpy(stuInParam.stPersonInfoEx.szGroupID, m_szGroupID, sizeof(stuInParam.stPersonInfoEx.szGroupID) - 1);

BOOL bRet = CLIENT_OperateFaceRecognitionDB(lLoginHandle, &stuInParam, &stuOutParam, 5000);
if (!bRet)
{
    printf("CLIENT_OperateFaceRecognitionDB failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}

```

3.3.4.2 Searching the People Face

```

// Set searching conditions of people face.
NET_IN_STARTFIND_FACERECONGNITION stuInParam = { sizeof(stuInParam) };
NET_OUT_STARTFIND_FACERECONGNITION stuOutParam = { sizeof(stuOutParam) };

stuInParam.stMatchOptions.dwSize = sizeof(stuInParam.stMatchOptions);
stuInParam.stFilterInfo.dwSize = sizeof(stuInParam.stFilterInfo);
stuInParam.bPersonExEnable = TRUE;
stuInParam.stFilterInfo.nRangeNum = 1;
stuInParam.stFilterInfo.szRange[0] = (BYTE)NET_FACE_DB_TYPE_BLACKLIST;
strncpy(stuInParam.stPersonInfoEx.szPersonName, m_PersonName,
sizeof(stuInParam.stPersonInfoEx.szPersonName)-1);
stuInParam.stPersonInfoEx.bySex = 0;
stuInParam.stFilterInfo.stBirthdayRangeStart = BirthdayRangeStart;
stuInParam.stFilterInfo.stBirthdayRangeEnd = BirthdayRangeEnd;
strncpy(stuInParam.stPersonInfoEx.szID, pcCard, sizeof(stuInParam.stPersonInfoEx.szID)-1);
strncpy(stuInParam.stFilterInfo.szGroupID[0], m_szGroupID, sizeof(m_szGroupID)-1);

```

```

stuInParam.stFilterInfo.nGroupIdNum = 1;
strncpy(stuInParam.stPersonInfoEx.szGroupID, m_szGroupId, sizeof(stuInParam.stPersonInfoEx.szGroupID)-1);

BOOL    bRet    =    CLIENT_StartFindFaceRecognition(m_ILoginID,    &stuInParam,    &stuOutParam,
DEFAULT_WAIT_TIME);
if (!bRet)
{
    printf("CLIENT_StartFindFaceRecognition failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

// Start searching
NET_IN_DOFIND_FACERECONGNITION stuInDoFind = {sizeof(stuInDoFind)};
NET_OUT_DOFIND_FACERECONGNITION stuOutDoFind = {sizeof(stuOutDoFind)};
stuOutDoFind.bUseCandidatesEx = TRUE;

stuInDoFind.IFindHandle = m_IFindPersonHandle;
stuInDoFind.emDataType = EM_NEEDED_PIC_TYPE_HTTP_URL;
stuInDoFind.nCount = 10;
stuInDoFind.nBeginNum = m_nCurPos;
bRet = CLIENT_DoFindFaceRecognition(&stuInDoFind, &stuOutDoFind, WAIT_TIMEOUT);
if (!bRet)
{
    printf("CLIENT_DoFindFaceRecognition failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

```

3.4 Arming by Channel or Library

3.4.1 Introduction

Arm by channel, means one channel arm one or multiple face libraries.

Arm by library, means one face library arm one or multiple channels.

These two ways are all arms of face library.

3.4.2 Interface Overview

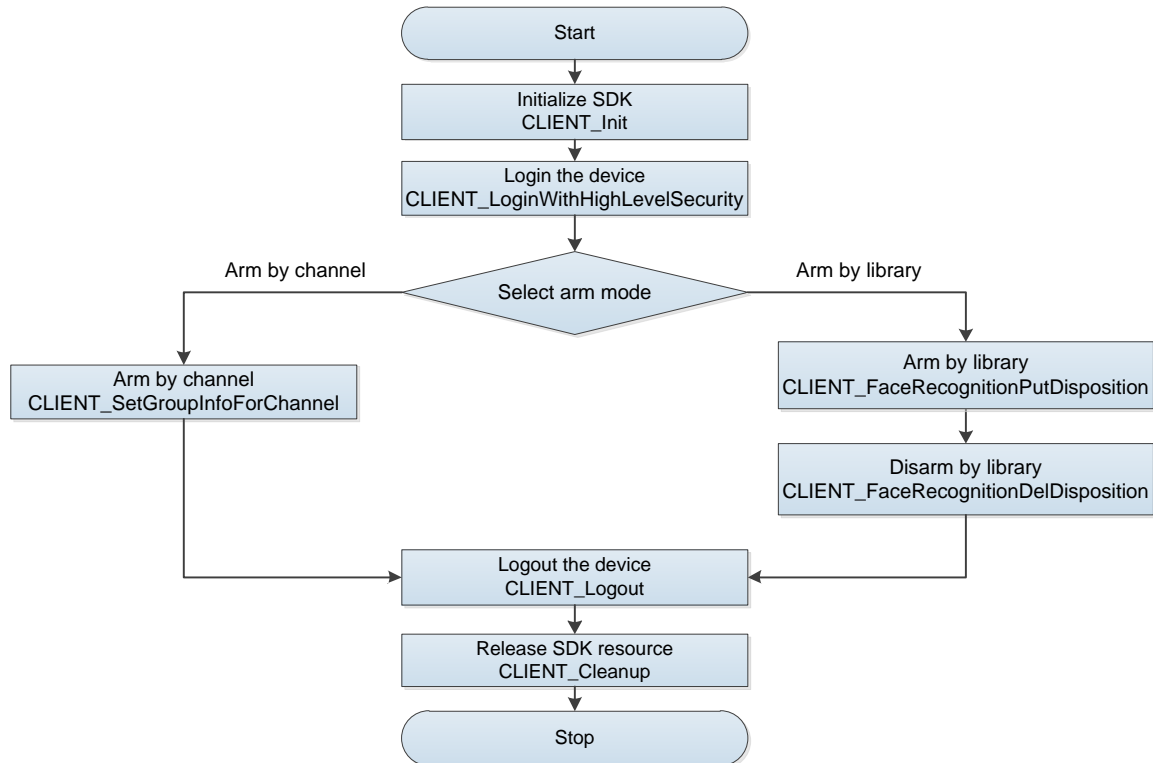
Table 3-3 Interfaces of arming by channel or library

Interface	Implication
CLIENT_FaceRecognitionPutDisposition	Arm by library.

Interface	Implication
CLIENT_FaceRecognitionDelDisposition	Disarm by library.
CLIENT_SetGroupInfoForChannel	Arm by channel.

3.4.3 Process

Figure 3-3 Process of arming by channel or library



Process Description

Step 1 Call **CLIENT_Init** to initialize SDK.

Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.

Step 3 Select arm way.

- Arm by library
 - 1) Select arm by library, call **CLIENT_FaceRecognitionPutDisposition** to arm the library.
 - 2) After using the function module, call **CLIENT_FaceRecognitionDelDisposition** to disarm the library.
- Arm by channel

Select arm by channel, call **CLIENT_SetGroupInfoForChannel** to arm the channel.

Step 4 After using the function module, call **CLIENT_Logout** to logout the device.

Step 5 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Arm by channel and arm by library are two ways of arming the face library.
- Arm by channel, means one channel arm multiple face libraries. Arm by library, means multiple channel arm one face libraries.

- When arm by channel, call **CLIENT_SetGroupInfoForChannel** to cover the old configurations. Disarm by channel, sent the empty arm information.
- When arm by library, call **CLIENT_FaceRecognitionDelDisposition** to disarm some channels. For example, 3 channels are armed, you can disarm 2 channels, and the other one stay the same.

3.4.4 Example Code

3.4.4.1 Arm by channel

```
// Input parameter
NET_IN_SET_GROUPINFO_FOR_CHANNEL          stInChannelDeploy          =
{ sizeof(NET_IN_SET_GROUPINFO_FOR_CHANNEL)};
stInChannelDeploy.nChannelID = 0;
stInChannelDeploy.nGroupIDNum = 2; // Set the face library number at this channel.
strncpy(stInChannelDeploy.szGroupID[0], strGroupID1, DH_COMMON_STRING_64-1); // Copy the face library
ID.
strncpy(stInChannelDeploy.szGroupID[1], strGroupID2, DH_COMMON_STRING_64-1);
stInChannelDeploy.nSimilaryNum = 2; // The similarity value number, which is equal to people groups.
stInChannelDeploy.nSimilary[0] = 85; // The similarity value at the first face library.
stInChannelDeploy.nSimilary[1] = 90; // The similarity value at the second face library.

// Output parameter
NET_OUT_SET_GROUPINFO_FOR_CHANNEL          stOutChannelDeploy          =
{ sizeof(NET_OUT_SET_GROUPINFO_FOR_CHANNEL)};

// Arm by library
BOOL bRet = CLIENT_SetGroupInfoForChannel(ILLoginHandle, &stInChannelDeploy, &stOutChannelDeploy);
if (flase == bRet)
{
    printf("CLIENT_SetGroupInfoForChannel: failed! Error code: %x.\n", CLIENT_GetLastError());
}

// Disarm by channel, sent the empty arm information.
if (NULL != IRealHandle)
{
    memset(stInChannelDeploy, 0, sizeof(NET_IN_SET_GROUPINFO_FOR_CHANNEL));
    memset(stOutChannelDeploy, 0, sizeof(NET_OUT_SET_GROUPINFO_FOR_CHANNEL));
    stInChannelDeploy.dwSize = sizeof(NET_IN_SET_GROUPINFO_FOR_CHANNEL);
    stOutChannelDeploy.dwSize = sizeof(NET_OUT_SET_GROUPINFO_FOR_CHANNEL);
    CLIENT_SetGroupInfoForChannel(ILLoginHandle, &stInChannelDeploy, &stOutChannelDeploy);
}
```


3.4.4.2 Arm by library

```
// Input parameter
NET_IN_FACE_RECOGNITION_PUT_DISPOSITION_INFO          stInFaceRecognitionDeploy          =
{ sizeof(NET_IN_FACE_RECOGNITION_PUT_DISPOSITION_INFO);
strncpy(stInFaceRecognitionDeploy.szGroupId, strGroupId, DH_COMMON_STRING_64-1); // Face library that
need to arm
stInFaceRecognitionDeploy.nDispositionChnNum = 2; // Number of video channel that armed
stInFaceRecognitionDeploy.stuDispositionChnInfo[0].nChannelID = 0;    // Face library deploy channel.
stInFaceRecognitionDeploy.stuDispositionChnInfo[0].nSimilary = 90;    // Similarity value.
stInFaceRecognitionDeploy.stuDispositionChnInfo[1].nChannelID = 2;
stInFaceRecognitionDeploy.stuDispositionChnInfo[1].nSimilary = 85;

// Output parameter
NET_OUT_FACE_RECOGNITION_PUT_DISPOSITION_INFO          stOutFaceRecognitionDeploy          =
{sizeof(NET_OUT_FACE_RECOGNITION_PUT_DISPOSITION_INFO));

// Arm by face library
bool    nRet    =    CLIENT_FaceRecognitionPutDisposition(ILLoginHandle,    &stInFaceRecognitionDeploy,
&stOutFaceRecognitionDeploy);
if(false = nRet)
{
    printf("CLIENT_FaceRecognitionPutDisposition: failed! Error code: %x.\n", CLIENT_GetLastError());
}

// Disarm input parameter, you can disarm some channels.
NET_IN_FACE_RECOGNITION_DEL_DISPOSITION_INFO          stInFaceRecognitionDel          =
{sizeof(NET_IN_FACE_RECOGNITION_DEL_DISPOSITION_INFO));
stInFaceRecognitionDel
strncpy(stInFaceRecognitionDel.szGroupId, strGroupId, DH_COMMON_STRING_64-1);
stInFaceRecognitionDel.nDispositionChnNum = 2; // Number of channel that armed
stInFaceRecognitionDel.nDispositionChn[0] = 0;    // Disarm channel 1
stInFaceRecognitionDel.nDispositionChn[1] = 1;

// Disarm output parameter
NET_OUT_FACE_RECOGNITION_DEL_DISPOSITION_INFO          stOutFaceRecognitionDel          =
{sizeof(NET_OUT_FACE_RECOGNITION_DEL_DISPOSITION_INFO));

bool    nRet    =    CLIENT_FaceRecognitionDelDisposition(ILLoginHandle,    &stInFaceRecognitionDel,
&stOutFaceRecognitionDel);
if(false = nRet)
```

```
{
    printf("CLIENT_FaceRecognitionDelDisposition: failed! Error code: %x.\n", CLIENT_GetLastError());
}
```

3.5 Searching for Picture by Picture

3.5.1 Introduction

You can import a picture and a similarity value, and then the IVSS and NVR devices will search the history library and the face library by this picture to make sure whether there is the matched face in the two libraries. And then it will return the right picture.

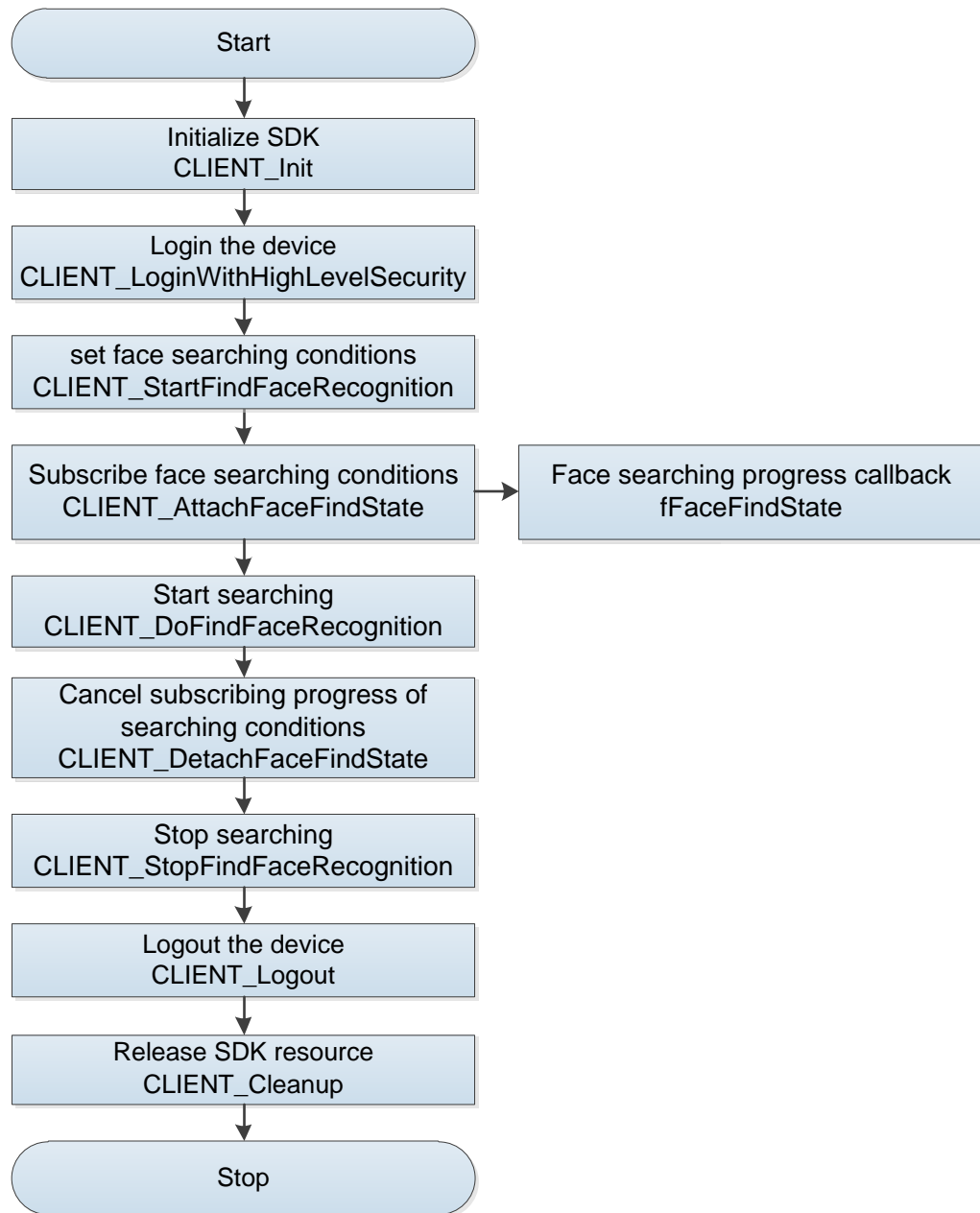
3.5.2 Interface Overview

Table 3-4 Interfaces of searching for picture by picture

Interface	Implication
CLIENT_StartFindFaceRecognition	Set the searching conditions of people face.
CLIENT_AttachFaceFindState	Subscribe searching conditions of people face.
CLIENT_DetachFaceFindState	Cancel subscribing the progress of searching conditions.
CLIENT_DoFindFaceRecognition	Start searching.
CLIENT_StopFindFaceRecognition	Stop searching.

3.5.3 Process

Figure 3-4 Process of searching picture by picture



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_StartFindFaceRecognition** to set the searching conditions of people face.
- Step 4 Check the return value in the Step3, If the nTotalCount in the output parameter structure returns -1, you need to wait until device searching complete.
- Step 5 Call **CLIENT_AttachFaceFindState** to subscribe the status of people face searching. Wait until the return progress of the progress callback function is 100, the searching is complete. Call **CLIENT_DetachFaceFindState** to cancel subscribing the searching progress.
- Step 6 Call **CLIENT_DoFindFaceRecognition** to get the searching result.
- Step 7 Call **CLIENT_StopFindFaceRecognition** to stop searching.

Step 8 After using the function module, call **CLIENT_Logout** to logout the device.

Step 9 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

3.5.4 Example Code

```
// Search progress callback function
void CALLBACK FaceFindState(LLONG ILoginID, LLONG IAttachHandle, NET_CB_FACE_FIND_STATE* pstStates,
int nStateNum, LDWORD dwUser)
{
    if (pstStates->nProgress== 100) // Means the searching progress is 100%.
    {
        //Stop subscribe the progress of people face searching
        CLIENT_DetachFaceFindState(IAttachHandle);
        // Start searching
        DoFind();
    }
    return;
}

// Configure searching conditions
NET_IN_STARTFIND_FACERECONGNITION stulnParam = { sizeof(stulnParam) };
NET_OUT_STARTFIND_FACERECONGNITION stuOutParam = { sizeof(stuOutParam) };
stulnParam.stFilterInfo.dwSize = sizeof(stulnParam.stFilterInfo);
stulnParam.stMatchOptions.dwSize = sizeof(stulnParam.stMatchOptions);
stulnParam.bPersonExEnable = TRUE;
stulnParam.nChannelID = 0;
stulnParam.stMatchOptions.nSimilarity = 80;
stulnParam.stFilterInfo.stStartTime = startTime;
stulnParam.stFilterInfo.stEndTime = endTime;
stulnParam.nBufferLen = nPicBufLen;
stulnParam.pBuffer = strPicBuf; // Picture Buffer
stulnParam.stPersonInfoEx.wFacePicNum = 1;
stulnParam.stPersonInfoEx.szFacePicInfo[0].dwOffSet = 0;
stulnParam.stPersonInfoEx.szFacePicInfo[0].dwFileLenth = nLength;

BOOL bRet = CLIENT_StartFindFaceRecognition(m_ILoginId, &stulnParam, &stuOutParam, 5000);
if (!bRet)
{
    printf("CLIENT_StartFindFaceRecognition: failed! Error code %x.\n", CLIENT_GetLastError());
    return -1;
}
```

```

m_IfindHandle = stuOutParam.IfFindHandle;

if (-1 == stuOutParam.nTotalCount)
{
    // When the total searching number is -1, it means the device searching is not completed. You need to
    subscribe the progress of device searching.
    NET_IN_FACE_FIND_STATE stuInFindState = { sizeof(stuInFindState) };
    NET_OUT_FACE_FIND_STATE stuOutFindState = { sizeof(stuOutFindState) };
    stuInFindState.nTokenNum = 1;
    int nToken = stuOutParam.nToken;
    stuInFindState.nTokens = &nToken;
    stuInFindState.cbFaceFindState = FaceFindState; // Progress callback function.
    stuInFindState.dwUser = (DWORD)this;
    m_IAttachHandle = CLIENT_AttachFaceFindState(mILoginId, &stuInFindState, &stuOutFindState, 5000);
}
else
{
    // Start searching.
    DoFind();
}

void DoFind()
{
    NET_IN_DOFIND_FACERECONGNITION stuInDoFind = { sizeof(NET_IN_DOFIND_FACERECONGNITION) };
    NET_OUT_DOFIND_FACERECONGNITION stuOutDoFind =
{ sizeof(NET_OUT_DOFIND_FACERECONGNITION) };
    stuOutDoFind.bUseCandidatesEx = TRUE;
    stuInDoFind.nCount = 20; // Search for 20 information one time, the total searching number is more than
20.
    stuInDoFind.IfFindHandle = m_IfindHandle;
    stuInDoFind.emDataType = EM_NEEDED_PIC_TYPE_HTTP_URL; // The returned picture format by
specified searching result is http link.
    stuInDoFind.nBeginNum = 0; // Search from 0.

    BOOL bRet = CLIENT_DoFindFaceRecognition(&stuInDoFind, &stuOutDoFind, 10000);
    if (!bRet)
    {
        printf("CLIENT_DoFindFaceRecognition: failed! Error code %x.\n", CLIENT_GetLastError());
        return ;
    }
}

```

```
CLIENT_StopFindFaceRecognition(m_IFindHandle);
```

```
}
```

3.6 Searching for and Downloading Face Video and Picture

The face intelligent event is one of the intelligent events, so for more details, See "2.5 Searching for/Playbacking/Downloading Video and Picture".

Call `CLIENT_FindFileEx` to set searching conditions. The following is about the face searching operation and the value of parameter `emType`.

- `DH_FILE_QUERY_FACE`: Search object recognition picture
- `DH_FILE_QUERY_FACE_DETECTION`: Search target detection picture
- `DH_FILE_QUERY_FILE`: Search video

The structure pointer of `ET_IN_MEDIA_QUERY_FILE`, the `nEventLists` field in the structure is as the following:

- `EVENT_IVS_FACERECOGNITION`: Object recognition video searching
- `EVENT_IVS_FACEDETECT`: Target detection video searching

4 Body Detection

4.1 Subscribing Body Event

About more details, see "2.4 Subscribing to Intelligent Event". Call `fAnalyzerDataCallBack` to filter out body detection, which is `EVENT_IVS_HUMANTRAIT` for body detection events.

4.2 Searching for the Body Picture

4.2.1 Introduction

For the code of body detection picture searching, see "2.5 Searching for/Playing/Downloading Video and Picture". The following section shows the description for body detection picture downloading.

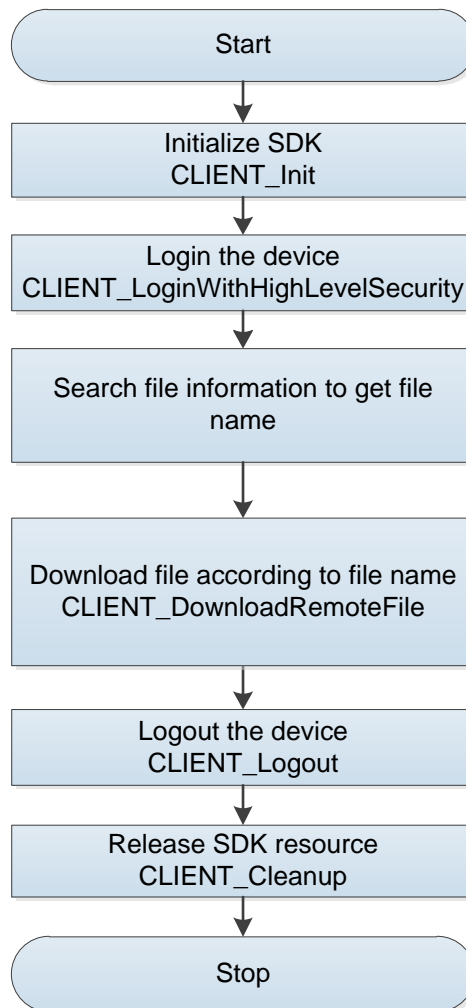
4.2.2 Interface Overview

Table 4-1 Interface of body picture searching

Interface	Implication
<code>CLIENT_DownloadRemoteFile</code>	Download file.

4.2.3 Process

Figure 4-1 Process of body picture searching



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.
- Step 3 Call SDK interface to search file information and to get the file name.
- Step 4 Using the searched file information, call **CLIENT_DownloadRemoteFile** to download file.
- Step 5 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 6 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

4.2.4 Example Code

```
DH_IN_DOWNLOAD_REMOTE_FILE stuInDownloadFile = {sizeof(DH_IN_DOWNLOAD_REMOTE_FILE )};
stuInDownloadFile.pszFileName = strFileName
stuInDownloadFile.pszFileDst = strDownloadName;

DH_OUT_DOWNLOAD_REMOTE_FILE stuOutDownloadFile = {sizeof(DH_OUT_DOWNLOAD_REMOTE_FILE )};
BOOL bRet = CLIENT_DownloadRemoteFile(m_LoginHandle, &stuInDownloadFile, &stuOutDownloadFile);
```



```
if (bRet == FALSE)
{
    MessageBox(ConvertString("Download Failed"), ConvertString("Prompt"));
    return;
}
```

5 People Flow Statistics

5.1 Subscribing to People Flow Event

5.1.1 Introduction

This is the real-time subscribe of flow statistics data function.

You can install the front-end devices in the specified areas to precise statistics the in and out number of people real-time in each entrance by the intelligent analysis server according to video data collected by the front-end devices.

You can also get the total in and out number of people in one single day and real-time in and out number of people.

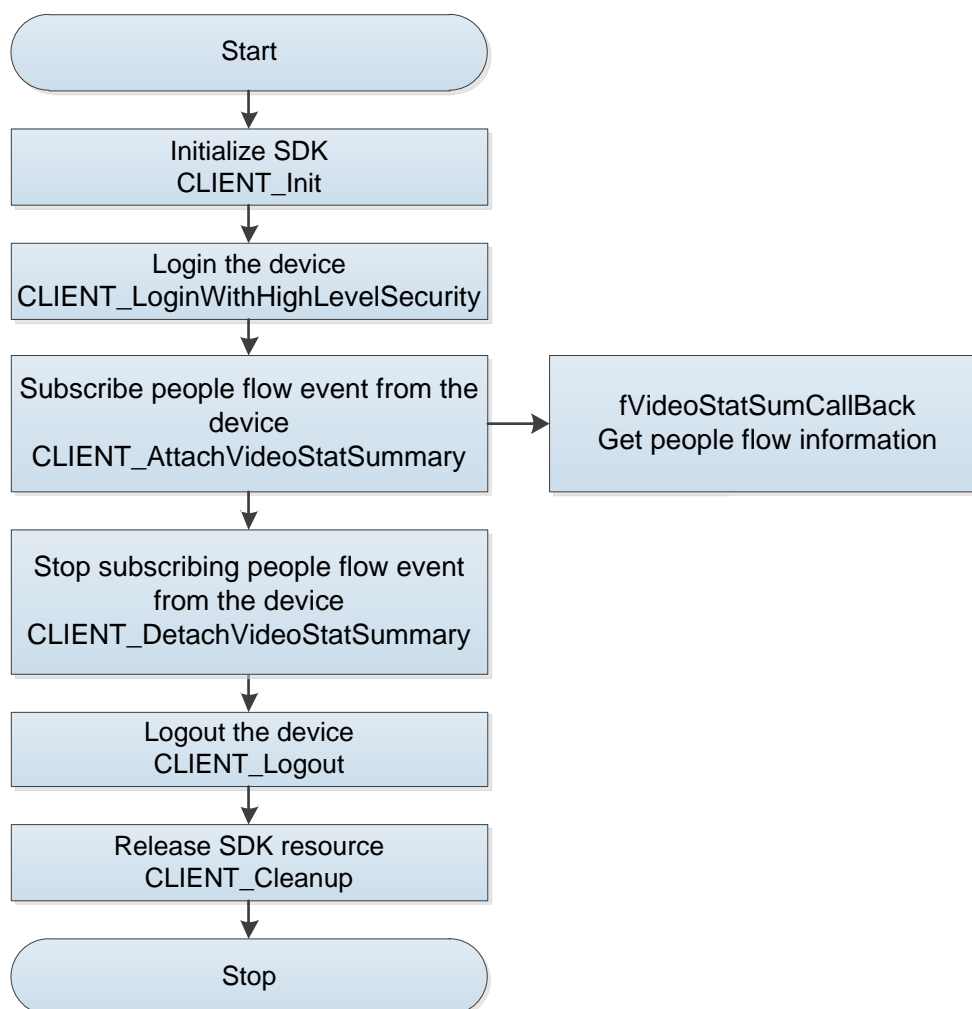
5.1.2 Interface Overview

Table 5-1 Interfaces of subscribing people flow

Interface	Implication
CLIENT_AttachVideoStatSummary	Subscribe people flow event.
CLIENT_DetachVideoStatSummary	Cancel subscribing people flow event.

5.1.3 Process

Figure 5-1 Process of subscribing people flow



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_AttachVideoStatSummary** to subscribe people flow events from the device.
- Step 4 After successful subscribe, call **fVideoStatSumCallBack** to get the face events and notify users.
- Step 5 After using the flow statistics event function, call **CLIENT_DetachVideoStatSummary** to stop subscribing people flow events.
- Step 6 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 7 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

5.1.4 Example Code

```
void_CALLBACK VideoStatSumCallback(LLONG IAttachHandle, NET_VIDEOSTAT_SUMMARY* pBuf, DWORD dwBufLen, LDWORD dwUser)
{
```

```

        // Produce callback data
    }

    NET_IN_ATTACH_VIDEOSTAT_SUM InParam = {sizeof(NET_IN_ATTACH_VIDEOSTAT_SUM)};
    NET_OUT_ATTACH_VIDEOSTAT_SUM OutParam = {sizeof(NET_OUT_ATTACH_VIDEOSTAT_SUM)};
    InParam.nChannel=0;
    InParam.cbVideoStatSum=VideoStatSumCallback; // Subscribe callback function.

    // Subscribe people flow statistics
    LLONG attachHnd = CLIENT_AttachVideoStatSummary(ILLoginID,&InParam,&OutParam,5000)
    if(0 == attachHnd)
    {
        printf("CLIENT_AttachVideoStatSummary failed! Error code %x.\n", CLIENT_GetLastError());
        return;
    }

    // Cancel subscribing people flow statistics
    CLIENT_DetachVideoStatSummary(attachHnd);

```

5.2 Alarm of People Flow Event

About more details, see "2.4 Subscribing to Intelligent Event". Call fAnalyzerDataCallBack to filter out people flow event, which is EVENT_IVS_NUMBERSTAT for people counting events and EVENT_IVS_MAN_NUM_DETECTION for area people counting events.

5.3 Searching for History Data of People Flow Statistics

5.3.1 Introduction

You can specify the start time and the end time of people flow information, and then the device end will send back the searching data to the SDK.

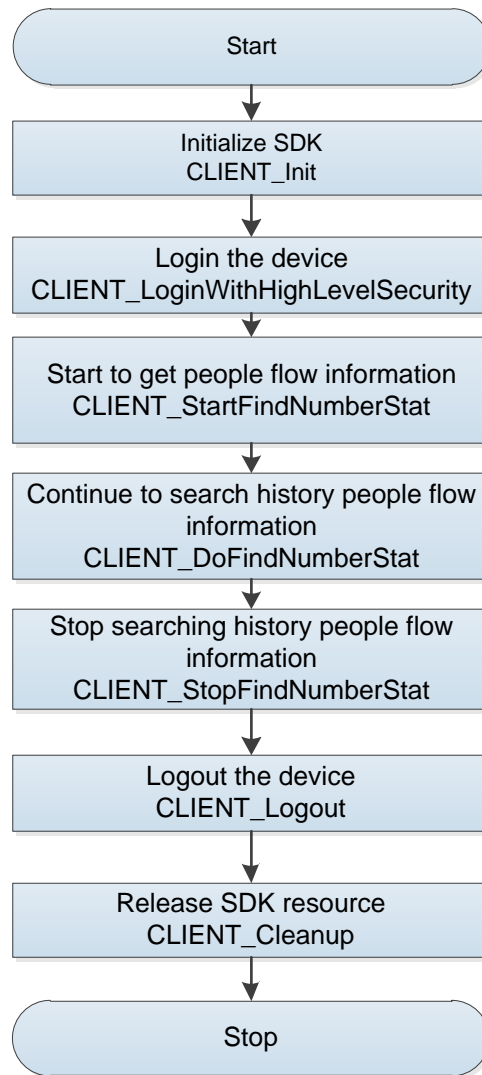
5.3.2 Interface Overview

Table 5-2 Interfaces of searching history data of people flow statistics

Interface	Implication
CLIENT_StartFindNumberStat	Start searching history people flow information.
CLIENT_DoFindNumberStat	Continue to search history people flow information.
CLIENT_StopFindNumberStat	Stop searching history people flow information.

5.3.3 Process

Figure 5-2 Searching process of people flow video and picture



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_StartFindNumberStat** to get people flow statistics information.
- Step 4 Call **CLIENT_DoFindNumberStat** to continue searching people flow statistics information at some period.
- Step 5 Call **CLIENT_StopFindNumberStat** to stop searching records.
- Step 6 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 7 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

5.3.4 Example Code

```
// Set searching conditions
NET_IN_FINDNUMBERSTAT inParam = { sizeof(NET_IN_FINDNUMBERSTAT)};
inParam.nChannelID = nChannelID; // The channel number you want to search
```

```

inParam.nGranularityType = 1; // Searching unit, 0: minute, 1: hour, 2: day, 3: week, 4: month, 5: season, 6: year
inParam.nWaittime = 5000; // Time-out for waiting received data
NET_OUT_FINDNUMBERSTAT outParam { sizeof(NET_OUT_FINDNUMBERSTAT));
LLONG findHnd = CLIENT_StartFindNumberStat(pLoginHandle, &inParam, &outParam);
if (findHand == 0)
{
    printf("CLIENT_StartFindNumberStat failed! Error code %x.\n", CLIENT_GetLastError());
    return ;
}

NET_IN_DOFINDNUMBERSTAT inDoFind = {sizeof(NET_IN_DOFINDNUMBERSTAT));
NET_OUT_DOFINDNUMBERSTAT outDoFind = {sizeof(NET_OUT_DOFINDNUMBERSTAT));
inDoFind.nBeginNumber = 0; // Search from 0
stulnDoFind.nCount = 10; // Search for 10 information one time.
inDoFind.nWaittime = 5000; // The time-out of interface is 5s.

outDoFind.pstuNumberStat = new DH_NUMBERSTAT[10];
outDoFind.nBufferLen = 10 * sizeof(DH_NUMBERSTAT);
for (int i = 0; i < 10 ; i++)
{
    outDoFind.pstuNumberStat[i].dwSize = sizeof(DH_NUMBERSTAT);
}

// Search
BOOL bRet = CLIENT_DoFindNumberStat(findHand, &inDoFind, &outDoFind)
if (FALSE == bRet)
{
    printf("CLIENT_DoFindNumberStat failed! Error code %x.\n", CLIENT_GetLastError());
    delete[] outDoFind.pstuNumberStat;
    return ;
}

// Stop searching people flow statistics
CLINET_StopFindNumberStat(findHand);
delete[] outDoFind.pstuNumberStat;

```

6 General Behavior Event

6.1 Subscribing to General Behavior Event

For more details, see "2.4 Subscribing to Intelligent Event". Call `fAnalyzerDataCallBack` to filter out general behavior events:

- `EVENT_IVS_CROSSLINEDETECTION`, tripwire event
- `EVENT_IVS_CROSSREGIONDETECTION`, intrusion event
- `EVENT_IVS_CROSSFENCEDETECTION`, cross fence event
- `EVENT_IVS_LEFTDETECTION`, abandoned object event
- `EVENT_IVS_MOVEDETECTION`, fast moving event
- `EVENT_IVS_RIOTERDETECTION`, crowd gathering event
- `EVENT_IVS_TAKENAWAYDETECTION`, missing object event
- `EVENT_IVS_PARKINGDETECTION`, parking detection event
- `EVENT_IVS_WANDERDETECTION`, loitering detection event

6.2 Video Searching and Downloading of General Behavior Event

The general behavior is one of the intelligent events. For more details, see "2.5 Searching for/Playingback/Downloading Video and Picture".

Call `CLIENT_FindFileEx` to set the searching conditions when searching. For the searching operation of general behavior video, the value of `emType` is as the follow:

- `emType`: `DH_FILE_QUERY_FILE`, searching for video,
- `pQueryCondition`: The structure pointer of type `NET_IN_MEDIA_QUERY_FILE`. The segments of `nEventLists` in the structure are as the follow:
 - ◇ `EVENT_IVS_CROSSLINEDETECTION`: Video search of tripwire.
 - ◇ `EVENT_IVS_CROSSREGIONDETECTION`: Video search of intrusion.
 - ◇ `EVENT_IVS_CROSSFENCEDETECTION`: Video search of crossing fence
 - ◇ `EVENT_IVS_LEFTDETECTION`: Video search of abandoned object
 - ◇ `EVENT_IVS_MOVEDETECTION`: Video search of fast moving
 - ◇ `EVENT_IVS_RIOTERDETECTION`: Video search of crowd gathering
 - ◇ `EVENT_IVS_TAKENAWAYDETECTION`: Video search of missing object
 - ◇ `EVENT_IVS_PARKINGDETECTION`: Video search of parking detection
 - ◇ `EVENT_IVS_WANDERDETECTION`: Video search of loitering detection

7 Intelligent Traffic

7.1 Subscribing to Intelligent Traffic Event

About more details, see "2.4 Subscribing to Intelligent Event". Call `fAnalyzerDataCallBack` to filter out the intelligent traffic event, which is as the following:

- `EVENT_IVS_TRAFFICJUNCTION`: Traffic junction event.
- `EVENT_IVS_TRAFFICJAM`: Traffic jam event.
- `EVENT_IVS_TRAFFIC_OVERSPEED`: Over speed event.
- `EVENT_IVS_TRAFFIC_UNDERSPEED`: Low speed event.
- `EVENT_IVS_TRAFFIC_PEDESTRAIN`: Passerby event.
- `EVENT_IVS_TRAFFIC_FLOWSTATE`: Vehicle flow event.
- `EVENT_IVS_TRAFFIC_VEHICLEINROUTE`: Vehicle in lane
- `EVENT_IVS_TRAFFIC_NON_MOTOR_RETROGRADE`: Wrong-way driving of non-motor vehicle
- `EVENT_IVS_TRAFFIC_OVERLINE`: Crossing solid white line
- `EVENT_IVS_TRAFFIC_OVERYELLOWLINE`: Crossing solid yellow line
- `EVENT_IVS_TRAFFIC_RETROGRADE`: Wrong-way driving
- `EVENT_IVS_TRAFFIC_CROSSLANE`: Illegal lane change
- `EVENT_IVS_TRAFFIC_QUEUEJUMP`: Vehicle queue jumping
- `EVENT_IVS_TRAFFIC_YELLOWPLATEINLANE`: Heavy vehicle in lane
- `EVENT_IVS_TRAFFIC_WRONGROUTE`: Disobeying lane direction sign
- `EVENT_IVS_TRAFFIC_UNDERSPEED`: Underspeed
- `EVENT_IVS_TRAFFIC_OVERSPEED`: Overspeed
- `EVENT_IVS_TRAFFIC_TRUCKFORBID`: No trucks
- `EVENT_IVS_TRAFFIC_UTURN`: Illegal U-turn
- `EVENT_IVS_TRAFFIC_TURNLEFT`: Illegal left turn
- `EVENT_IVS_TRAFFIC_TURNRIGHT`: Illegal right turn
- `EVENT_IVS_TRAFFIC_BACKING`: Illegal backing
- `EVENT_IVS_TRAFFIC_PEDESTRAINPRIORITY`: Failed to yield to pedestrians

7.2 Searching for History Data of Vehicle Flow Statistics

7.2.1 Introduction

Search the history data of vehicle flow statistics.

7.2.2 Interface Overview

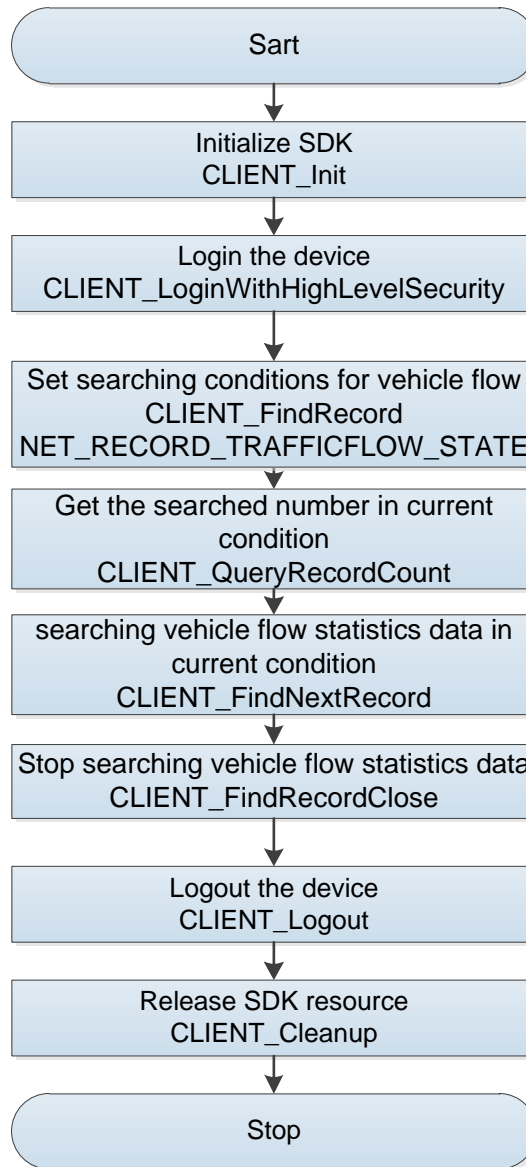
Table 7-1 Interfaces of searching history data of vehicle flow statistics

Interface	Implication
<code>CLIENT_FindRecord</code>	Set searching conditions.
<code>CLIENT_QueryRecordCount</code>	Get searching number.

Interface	Implication
CLIENT_FindNextRecord	Search data in the current searching condition.
CLIENT_FindRecordClose	Stop searching.

7.2.3 Process

Figure 7-1 Searching history data of vehicle flow statistics



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.
- Step 3 Call **CLIENT_FindRecord** to set searching conditions for vehicle flow statistics. The enumeration is NET_RECORD_TRAFFICFLOW_STATE.
- Step 4 Call **CLIENT_QueryRecordCount** to get the total number in the current searching conditions.
- Step 5 Call **CLIENT_FindNextRecord** to search the specified number in the current searching

conditions.

Step 6 After searching, call **CLIENT_FindRecordClose** to clean up the searching resource.

Step 7 After using the function module, call **CLIENT_Logout** to logout the device.

Step 8 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- During searching the vehicle flow, at first, the device must support this function and have the vehicle flow data during the searching time. In addition, the device should have an SD card to save the vehicle flow data.
- The segment of the average speed is -1, which means that no vehicle has passed this period. If it is more than 1, it means that the average speed of the vehicle. If it is equal to 0, it means that the average speed is 0.

7.2.4 Example Code

```
// Start searching and set searching conditions
FIND_RECORD_TRAFFICFLOW_CONDITION stTrafficFlow = {sizeof(FIND_RECORD_TRAFFICFLOW_CONDITION)};
stTrafficFlow.abChannelId = TRUE;
stTrafficFlow.nChannelId = 0;
stTrafficFlow.abLane = FALSE;
stTrafficFlow.bStartTime = TRUE;
stTrafficFlow.bEndTime = TRUE;
stTrafficFlow.stStartTime = startTime;
stTrafficFlow.stEndTime = endTime;
stTrafficFlow.bStatisticsTime = TRUE;

NET_IN_FIND_RECORD_PARAM stuFindInParam = {sizeof(NET_IN_FIND_RECORD_PARAM)};
stuFindInParam.emType = NET_RECORD_TRAFFICFLOW_STATE;
stuFindInParam.pQueryCondition = &stTrafficFlow;

NET_OUT_FIND_RECORD_PARAM stuFindOutParam = {sizeof(NET_OUT_FIND_RECORD_PARAM)};
bool bRet = CLIENT_FindRecord(mILoginHandle, &stuFindInParam, &stuFindOutParam, MAX_TIMEOUT);
if (!bRet)
{
    return;
}

// Search the total number
NET_IN_QUEYT_RECORD_COUNT_PARAM inQueryCountParam =
{ sizeof(NET_IN_QUEYT_RECORD_COUNT_PARAM)};
inQueryCountParam.lFindeHandle = stuFindOutParam.lFindeHandle;
```

```

NET_OUT_QUEYT_RECORD_COUNT_PARAM          outQueryCountParam          =
{ sizeof(NET_OUT_QUEYT_RECORD_COUNT_PARAM) };

bRet = CLIENT_QueryRecordCount(&inQueryCountParam, &outQueryCountParam, MAX_TIMEOUT);
if (!bRet)
{
    Printf("Query record count failed!\n");
    return;
}

// Search 100 information.
int nQueryCount = 100;
NET_RECORD_TRAFFIC_FLOW_STATE*             pRecordList                =          new
NET_RECORD_TRAFFIC_FLOW_STATE[nQueryCount];
memset(pRecordList, 0, sizeof(NET_RECORD_TRAFFIC_FLOW_STATE) * nQueryCount);
for (int unIndex = 0; unIndex < nQueryCount; ++unIndex)
{
    pRecordList[unIndex].dwSize = sizeof(NET_RECORD_TRAFFIC_FLOW_STATE);
}

NET_IN_FIND_NEXT_RECORD_PARAM               stuFindNextInParam          =
{sizeof(NET_IN_FIND_NEXT_RECORD_PARAM)};
stuFindNextInParam.IFindeHandle = stuFindOutParam.IFindeHandle;
stuFindNextInParam.nFileCount = nQueryCount;

NET_OUT_FIND_NEXT_RECORD_PARAM              stuFindNextOutParam          =
{sizeof(NET_OUT_FIND_NEXT_RECORD_PARAM)};
stuFindNextOutParam.pRecordList = pRecordList;
stuFindNextOutParam.nMaxRecordNum = nQueryCount;
bRet = CLIENT_FindNextRecord(&stuFindNextInParam, &stuFindNextOutParam, MAX_TIMEOUT);
if (!bRet)
{
    printf("Query record count failed!");
}

// Stop searching.
CLIENT_FindRecordClose(stuFindOutParam.IFindeHandle);
delete[] pRecordList;

```

7.3 Adding/deleting/modifying/searching for Blocklist and Allowlist of Vehicle

7.3.1 Introduction

You can add, delete, modify and search for blocklist and allowlist of vehicle.

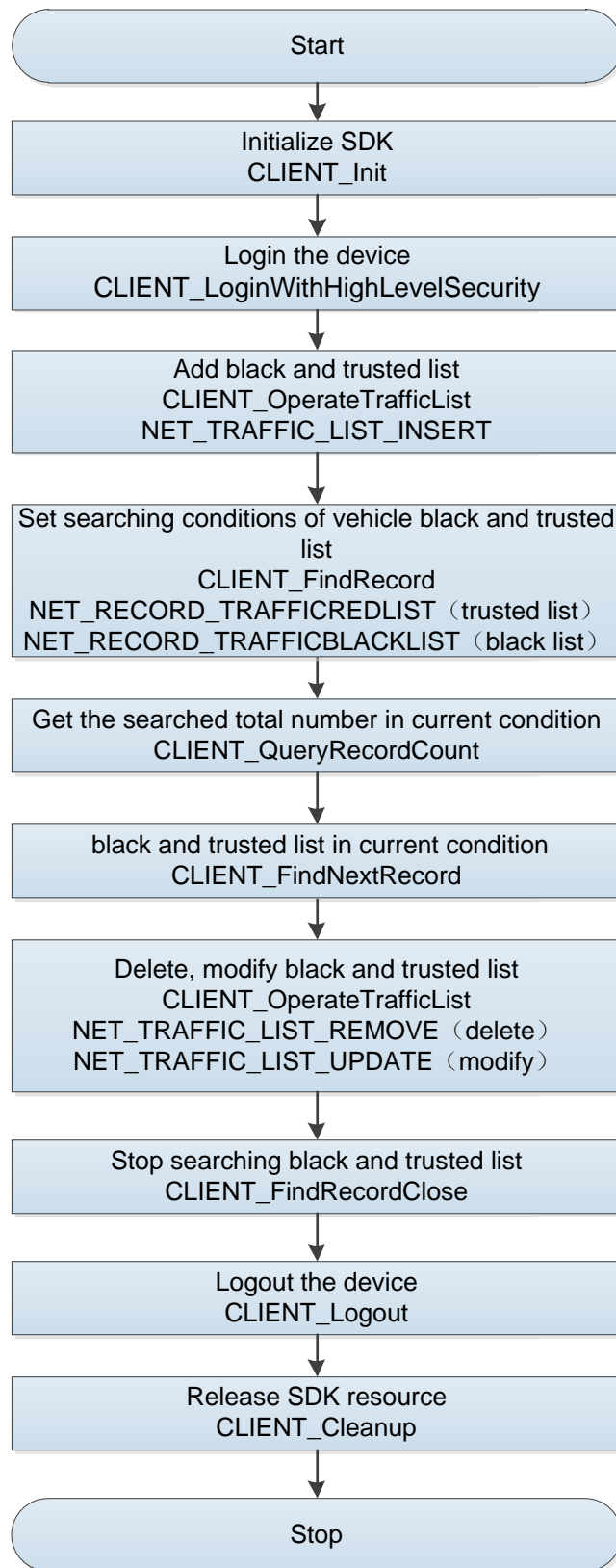
7.3.2 Interface Overview

Table 7-2 Interfaces of adding, deleting, modifying and searching for blocklist and allowlist of vehicle,

Interface	Implication
CLIENT_OperateTrafficList	Add, delete, modify and search for the blocklist and allowlist.
CLIENT_FindRecord	Set searching conditions.
CLIENT_QueryRecordCount	Get searching number.
CLIENT_FindNextRecord	Search data in the current searching condition.
CLIENT_FindRecordClose	Close searching.

7.3.3 Process

Figure 7-2 Adding/deleting/modifying/searching blocklist and allowlist of vehicle



Process Description

Step 1 Call **CLIENT_Init** to initialize SDK.

- Step 2** Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3** Call **CLIENT_OperateTrafficList** to add the blocklist and allowlist. The enumeration type is **NET_TRAFFIC_LIST_INSERT**.
- Step 4** Call **CLIENT_FindRecord** to set searching conditions for blocklist and allowlist. The enumeration is **NET_RECORD_TRAFFICREDLIST** (trusted list) and **NET_RECORD_TRAFFICBLACKLIST** (blocklist).
- Step 5** Call **CLIENT_QueryRecordCount** to get the total number in the current searching conditions.
- Step 6** Call **CLIENT_FindNextRecord** to search the specified number of the blocklist and allowlist in the current searching conditions.
- Step 7** Using the blocklist and allowlist you have searched, call **CLIENT_OperateTrafficList** blocklist and allowlist. The enumeration is **NET_TRAFFIC_LIST_REMOVE** (delete) and **NET_TRAFFIC_LIST_UPDATE** (modify).
- Step 8** After searching, call **CLIENT_FindRecordClose** to clean up the searching resource.
- Step 9** After using the function module, call **CLIENT_Logout** to logout the device.
- Step 10** After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

Call **CLIENT_OperateTrafficList** to add blocklist and allowlist. When the interface returns -1, it means that do not generate a record list number but not means the interface failed. At first, the device saves the blocklist and allowlist to the cache, and then saves the data in the cache to the database. The device returns -1 because at this time the lists have not been added to the database.

7.3.4 Example Code

To check the code of blocklist and allowlist, see "7.2.4 Example Code". The following shows the example code of adding/deleting/modifying/searching blocklist and allowlist.

```
// Add blocklist and allowlist.
NET_IN_OPERATE_TRAFFIC_LIST_RECORD stInParam = { sizeof(NET_IN_OPERATE_TRAFFIC_LIST_RECORD) };
NET_OUT_OPERATE_TRAFFIC_LIST_RECORD stOutParam =
{ sizeof(NET_OUT_OPERATE_TRAFFIC_LIST_RECORD) };
stInParam.emOperateType = NET_TRAFFIC_LIST_INSERT;
stInParam.emRecordType = NET_RECORD_TRAFFICBLACKLIST; // Blocklist
//stInParam.emRecordType = NET_RECORD_TRAFFICREDLIST; // Allowlist

NET_TRAFFIC_LIST_RECORD stTrafficListRecord = { sizeof(NET_TRAFFIC_LIST_RECORD) };
stTrafficListRecord.stBeginTime = startTime;
stTrafficListRecord.stCancelTime = endTime;
strncpy(stTrafficListRecord.szPlateNumber, strPlateNumber.GetBuffer(), DH_MAX_PLATE_NUMBER_LEN-1);
strncpy(stTrafficListRecord.szMasterOfCar, strOwner.GetBuffer(), DH_MAX_NAME_LEN-1);

NET_INSERT_RECORD_INFO stInsertInfo = { sizeof( NET_INSERT_RECORD_INFO ) };
stInsertInfo.pRecordInfo = &stTrafficListRecord;
```

```

stInParam.pstOpreateInfo = &stInsertInfo;

bool bRet = CLIENT_OperateTrafficList(m_ILoginHandle, &stInParam, &stOutParam, MAX_TIMEOUT);
if (!bRet)
{
    return;
}

// Modify blocklist.
NET_IN_OPERATE_TRAFFIC_LIST_RECORD stInParam = { sizeof(NET_IN_OPERATE_TRAFFIC_LIST_RECORD) };
NET_OUT_OPERATE_TRAFFIC_LIST_RECORD stOutParam = { sizeof(NET_OUT_OPERATE_TRAFFIC_LIST_RECORD) };

stInParam.emOperateType = NET_TRAFFIC_LIST_UPDATE;
stInParam.emRecordType = NET_RECORD_TRAFFICBLACKLIST; // Blocklist
//stInParam.emRecordType = NET_RECORD_TRAFFICREDLIST; // Trusted list

NET_TRAFFIC_LIST_RECORD stTrafficListRecord = { sizeof(NET_TRAFFIC_LIST_RECORD) };
stTrafficListRecord.stBeginTime = startTime;
stTrafficListRecord.stCancelTime = endTime;
strncpy(stTrafficListRecord.szPlateNumber, strPlateNumber.GetBuffer(), DH_MAX_PLATE_NUMBER_LEN-1);
strncpy(stTrafficListRecord.szMasterOfCar, strOwner.GetBuffer(), DH_MAX_NAME_LEN-1);
stTrafficListRecord.nRecordNo = m_stTrafficListInfo.nRecordNo; // Recording list number

NET_UPDATE_RECORD_INFO stModifyRecord = { sizeof(NET_UPDATE_RECORD_INFO) };
stModifyRecord.pRecordInfo = &stTrafficListRecord;
stInParam.pstOpreateInfo = &stModifyRecord;

bool bRet = CLIENT_OperateTrafficList(m_ILoginHandle, &stInParam, &stOutParam, MAX_TIMEOUT);
if (!bRet)
{
    return;
}

// Modify blocklist and allowlist.
NET_REMOVE_RECORD_INFO stRemoveRecord = { sizeof(NET_REMOVE_RECORD_INFO) };
stRemoveRecord.nRecordNo = m_vecTrafficListInfo[nSelect]->nRecordNo; // Recording list number

NET_IN_OPERATE_TRAFFIC_LIST_RECORD stInParam = { sizeof(NET_IN_OPERATE_TRAFFIC_LIST_RECORD) };

```

```

stInParam.emOperateType = NET_TRAFFIC_LIST_REMOVE;
stInParam.emRecordType = NET_RECORD_TRAFFICBLACKLIST; // Blocklist
//stInParam.emRecordType = NET_RECORD_TRAFFICREDLIST; // Allowlist

stInParam.pstOpreateInfo = &stRemoveRecord;
NET_OUT_OPERATE_TRAFFIC_LIST_RECORD stOutParam =
{ sizeof(NET_OUT_OPERATE_TRAFFIC_LIST_RECORD) };

bool bRet = CLIENT_OperateTrafficList(m_ILoginHandle, &stInParam, &stOutParam, MAX_TIMEOUT);
if (!bRet)
{
    return;
}

```

7.4 Searching for and Downloading Vehicle Picture

7.4.1 Introduction

This function will save the capture picture for intelligent event to the storage in the device. You can search pictures through the plate number and corresponding intelligent events. This function also supports picture downloading.

For the vehicle picture searching, see "2.5 Searching for/Playing/Downloading Video and Picture". The interface **CLIENT_FindFileEx** searching type is DH_FILE_QUERY_TRAFFICCAR_EX.

The following shows the interface and example code.

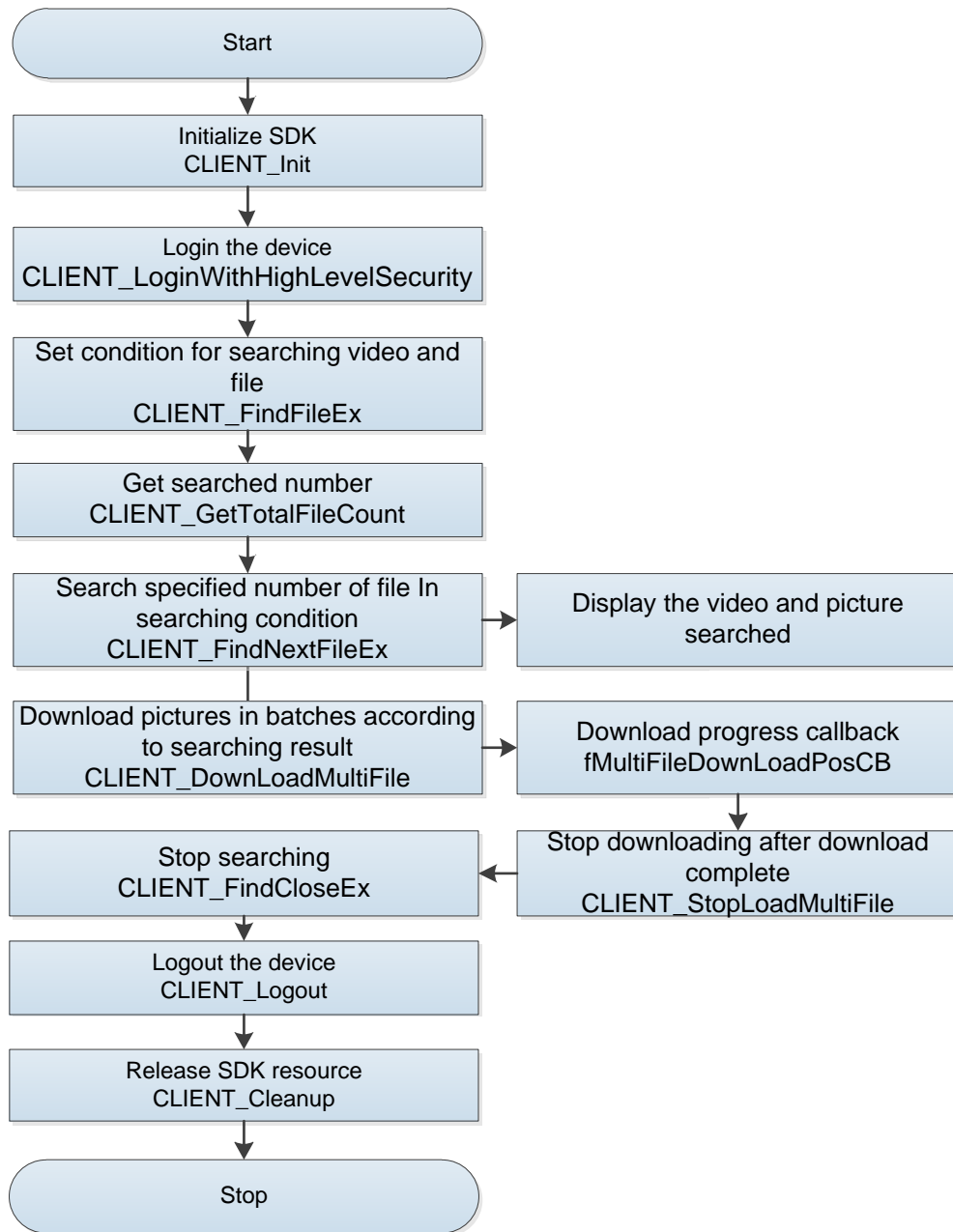
7.4.2 Interface Overview

Table 7-3 Interfaces of searching and downloading picture

Interface	Implication
CLIENT_DownloadMultiFile	Download files in batches.
CLIENT_StopLoadMultiFile	Stop downloading files in batches.

7.4.3 Process

Figure 7-3 Process of searching for and downloading picture



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call **CLIENT_LoginWithHighLevelSecurity** to login the device.
- Step 3 Call **CLIENT_FindFileEx** to set the searching conditions. After successfully setting, return the searching handle. To judge the right searching type according to the different values of emType.
- Step 4 Call **CLIENT_GetTotalFileCount** to get the total number of video and picture searched.
- Step 5 Call **CLIENT_FindNextFileEx** to search the specified number of video and picture. Save the video and picture and do the playing back and downloading operation to the video and picture.
- Step 6 Using the searched file information, call **CLIENT_DownloadMultiFile** to download files in

batches.

Step 7 Call **CLIENT_StopLoadMultiFile** to stop downloading the picture when the value of dwDownloadSize in fMultiFileDownloadPosCB is the maximum value.

Step 8 Call **CLIENT_FindCloseEx** to stop the searching.

Step 9 After using the function module, call **CLIENT_Logout** to logout the device.

Step 10 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- The parameter pQueryCondition in **CLIENT_FindFileEx** is requested and released by the user. The specific type is defined by the enumeration type of emType.
- If **CLIENT_FindFileEx** successfully search, the searching handle will be returned. **CLIENT_FindNextFileEx** will take the searching handle as a parameter to search specific video and picture. You should call **CLIENT_FindCloseEx** to close the searching handle.
- Call **CLIENT_FindNextFileEx** to set the searching number. If the number is more than 1, then the parameter pMediaFileInfo should be taken as a data pointer.

7.4.4 Example Code

```
// Download progress callback function in batches
void CALLBACK DownloadTrafficPicture(LLONG IDownloadHandle, DWORD dwID, DWORD dwFileTotalSize,
DWORD dwDownloadSize, int nError, LDWORD dwUser, void* pReserved)
{
    if (nError != 0 || dwDownloadSize == UINT_MAX)
    {
        // Download error or download completed
        CLIENT_StopLoadMultiFile(m_IDownloadHandle);
        delete[] pDownloadInfo;
    }
}

NET_DOWNLOADFILE_INFO* pDownloadInfo = new NET_DOWNLOADFILE_INFO[10]; // Download 10
information
memset(pDownloadInfo, 0, 10*sizeof(NET_DOWNLOADFILE_INFO));
for(int i=0; i++; i<10)
{
    pDownloadInfo[i].dwFileID = 1;

    // The data of stTrafficPicture is the file searched from CLIENT_FindFileEx. The type is
    MEDIAFILE_TRAFFICCAR_INFO_EX.
    pDownloadInfo[i].nFileSize = stTrafficPicture.stulInfo.sizeEx / 1024;
    strncpy(pDownloadInfo[i].szSourceFilePath, stTrafficPicture.stulInfo.szFilePath, MAX_PATH-1);
}
```

```

    // The save path after downloading
    strncpy(pDownloadInfo[i].szSavedFileName, szFilePathName[i], MAX_PATH-1);
}

NET_IN_DOWNLOAD_MULTI_FILE stInDownloadFile = {sizeof(NET_IN_DOWNLOAD_MULTI_FILE)};
NET_OUT_DOWNLOAD_MULTI_FILE stOutDownloadFile= {sizeof(NET_OUT_DOWNLOAD_MULTI_FILE)};
stInDownloadFile.emDownloadType = EM_DOWNLOAD_BY_FILENAME;
stInDownloadFile.cbPosCallBack = DownloadTrafficPicture; // Download in batches progress callback
function
stInDownloadFile.dwUserData = (LDWORD)this;
stInDownloadFile.nFileCount = 10; // The number of downloading file
stInDownloadFile.pFileInfos = pDownloadInfo; // It means the file information pointer that you need to
download. If you need to download multiple file information pointers, it means the array first address.

// Download picture files in batches
BOOL nRet = CLIENT_DownloadMultiFile(m_ILoginHandle, &stInDownloadFile, &stOutDownloadFile,
MAX_TIMEOUT);
if (!nRet)
{
    printf("CLIENT_DownloadMultiFile failed! Error code %x.\n", CLIENT_GetLastError());
    delete[] pDownloadInfo;
    return;
}
m_IDownloadHandle = stOutDownloadFile.IDownloadHandle;

```

8 Barrier

8.1 Subscribing to Access Control Event

About more details, see "2.4 Subscribing to Intelligent Event". Call fAnalyzerDataCallBack to filter out access control event, which is as the following:

EVENT_IVS_ACCESS_CTL: Access control event

8.2 Manager Information of Access Control Card

8.2.1 Introduction

You can add, delete, modify and search the information of access control card such as the card number, user ID and card name.

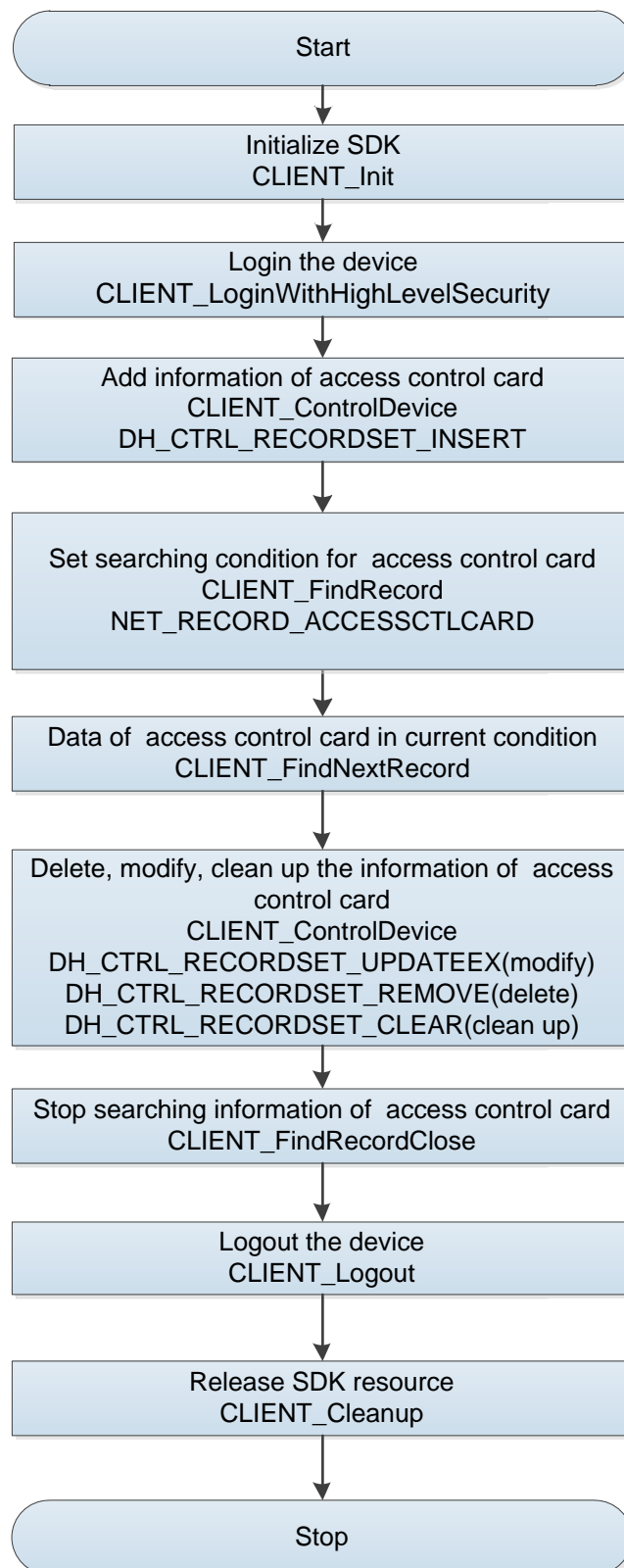
8.2.2 Interface Overview

Table 8-1 Interfaces of manager information of access control card

Interface	Implication
CLIENT_FindRecord	Set searching conditions.
CLIENT_FindNextRecord	Search data in the current searching condition.
CLIENT_FindRecordClose	Close searching.
CLIENT_ControlDevice	Add, delete, modify and search the information of access control card.

8.2.3 Process

Figure 8-1 Adding/deleting/modifying/searching the information of access control card



Process Description

Step 1 Call **CLIENT_Init** to initialize SDK.

Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.

- Step 3** Call **CLIENT_ControlDevice** to add the information of access control card. The enumeration is DH_CTRL_RECORDSET_INSERT.
- Step 4** Call **CLIENT_FindRecord** to set searching conditions for the information of access control card. The enumeration is NET_RECORD_ACCESSCTLCARD.
- Step 5** Call **CLIENT_FindNextRecord** to search the specified number of the information data of access control card in the current searching conditions.
- Step 6** After searching, call **CLIENT_FindRecordClose** to clean up the searching resource.
- Step 7** Call **CLIENT_ControlDevice** to modify the information of access control card. The enumeration is NET_RECORD_ACCESSCTLCARD.
- Step 8** Call **CLIENT_ControlDevice** to delete the information of access control card. The enumeration is DH_CTRL_RECORDSET_REMOVE.
- Step 9** Call **CLIENT_ControlDevice** to clean up the information of access control card. The enumeration is DH_CTRL_RECORDSET_CLEAR.
- Step 10** After using the function module, call **CLIENT_Logout** to logout the device.
- Step 11** After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- When adding the access control card, make sure the CardNo and UserID are different with the information of access control card which saved in the device before.
- When deleting the access control card, and if the access control card binding with people face picture information, you should delete the information of access control card before deleting people face picture.
- During searching the information of access control card, firstly the device should support this function, and the device itself should have the information data of access control card.

8.2.4 Example Code

8.2.4.1 Searching the Information of Access Control Card

```
NET_OUT_FIND_RECORD_PARAM stuOutParam= sizeof(NET_OUT_FIND_RECORD_PARAM);
NET_IN_FIND_RECORD_PARAM stuInParam= sizeof(stuInParam);

stuInParam.emType = NET_RECORD_ACCESSCTLCARD;
FIND_RECORD_ACCESSCTLCARD_CONDITION *pStuCardInfo = NEW
FIND_RECORD_ACCESSCTLCARD_CONDITION;
pStuCardInfo->dwSize = sizeof(FIND_RECORD_ACCESSCTLCARD_CONDITION);
stuInParam.pQueryCondition = (void*)pStuCardInfo;

// Start searching and set searching conditions
bRet = CLIENT_FindRecord(m_lLoginID, &stuInParam, &stuOutParam, DEFAULT_WAIT_TIME);
if (bRet == FALSE)
{
    printf("CLIENT_FindRecord fail \n");
}
```

```

}

if (pStuCardInfo)
{
    delete pStuCardInfo;
    pStuCardInfo = NULL;
}

// Search 100 information.
NET_IN_FIND_NEXT_RECORD_PARAM stuInParam= sizeof(stuInParam);
NET_OUT_FIND_NEXT_RECORD_PARAM stuOutParam = sizeof(stuOutParam);
stuInParam.lFindHandle = m_lFindHandle;
stuInParam.nFileCount = 100;
memset(pAccessCardInfo, 0, stuInParam.nFileCount * sizeof(NET_RECORDSET_ACCESS_CTL_CARD));

for (int i = 0; i < stuInParam.nFileCount; i++)
{
    pAccessCardInfo[i].dwSize = sizeof(NET_RECORDSET_ACCESS_CTL_CARD);
}
stuOutParam.pRecordList = (void*)pAccessCardInfo;
stuOutParam.nMaxRecordNum = stuInParam.nFileCount;

int nRet = CLIENT_FindNextRecord(&stuInParam, &stuOutParam, DEFAULT_WAIT_TIME);

if (!nRet)
{
    printf("CLIENT_FindNextRecord failed \n");
}

// Stop searching.
CLIENT_FindRecordClose(m_lFindHandle);

```

8.2.4.2 Adding/Deleting/Modifying/Searching the Information of Access Control Card

```

// Add the information of access control card
NET_CTRL_RECORDSET_INSERT_PARAM stuInParam = sizeof(stuInParam);
stuInParam.stuCtrlRecordSetInfo.dwSize = sizeof(NET_CTRL_RECORDSET_INSERT_IN);
stuInParam.stuCtrlRecordSetResult.dwSize = sizeof(NET_CTRL_RECORDSET_INSERT_OUT);
stuInParam.stuCtrlRecordSetInfo.emType = NET_RECORD_ACCESSCTLCARD;

```

```

NET_RECORDSET_ACCESS_CTL_CARD *pStrCardInfo = NEW NET_RECORDSET_ACCESS_CTL_CARD;

pStrCardInfo->dwSize = sizeof(NET_RECORDSET_ACCESS_CTL_CARD);
strncpy(pStrCardInfo->szCardNo, m_StuAddCardInfo.szCardNo, DH_MAX_CARDNO_LEN - 1);
strncpy(pStrCardInfo->szCardName, m_StuAddCardInfo.szCardName, DH_MAX_CARDNAME_LEN - 1);
strncpy(pStrCardInfo->szUserID, m_StuAddCardInfo.szUserID, DH_MAX_USERID_LEN - 1);
strncpy(pStrCardInfo->szPsw, m_StuAddCardInfo.szPsw, DH_MAX_CARDPWD_LEN - 1);
pStrCardInfo->emStatus = NET_ACCESSCTLCARD_STATE_NORMAL;
pStrCardInfo->emType = NET_ACCESSCTLCARD_TYPE_GENERAL;
pStrCardInfo->nUserTime = m_StuAddCardInfo.nUserTime;
pStrCardInfo->bFirstEnter = TRUE;
pStrCardInfo->blsValid = TRUE
pStrCardInfo->stuValidStartTime = m_StuAddCardInfo.stuValidStartTime;
pStrCardInfo->stuValidEndTime = m_StuAddCardInfo.stuValidEndTime;

// DoorNum is 2, it indicates the two doors of the gate.
pStrCardInfo->nDoorNum = 2;
pStrCardInfo->sznDoors[0] = 0;
pStrCardInfo->sznDoors[1] = 1;
// Control the valid time of the opening door, 255 indicates all day.
pStrCardInfo->nTimeSectionNum = 2;
pStrCardInfo->sznTimeSectionNo[0] = 255;
pStrCardInfo->sznTimeSectionNo[1] = 255;

stuInParam.stuCtrlRecordSetInfo.nBufLen = sizeof(NET_RECORDSET_ACCESS_CTL_CARD);
stuInParam.stuCtrlRecordSetInfo.pBuf = (void*)pStrCardInfo;

BOOL    bRet    =    CLIENT_ControlDevice(m_ILoginID,    DH_CTRL_RECORDSET_INSERT,    &stuInParam,
DEFAULT_WAIT_TIME);
if (bRet == FALSE)
{
    Printf("CLIENT_ControlDevice insert card fail \n");
    return;
}
// Modify the information of access control card
NET_CTRL_RECORDSET_PARAM stuInParam = sizeof(stuInParam);
stuInParam.emType = NET_RECORD_ACCESSCTLCARD;

NET_RECORDSET_ACCESS_CTL_CARD *pStrCardInfo = NEW NET_RECORDSET_ACCESS_CTL_CARD;

```



```

memset(pStrCardInfo, 0, sizeof(NET_RECORDSET_ACCESS_CTL_CARD));

pStrCardInfo->dwSize = sizeof(NET_RECORDSET_ACCESS_CTL_CARD);
strncpy(pStrCardInfo->szCardNo, m_CardInfo.szCardNo, DH_MAX_CARDNO_LEN - 1);
strncpy(pStrCardInfo->szCardName, m_CardInfo.szCardName, DH_MAX_CARDNAME_LEN - 1);
strncpy(pStrCardInfo->szUserID, m_CardInfo.szUserID, DH_MAX_USERID_LEN - 1);
strncpy(pStrCardInfo->szPsw, m_CardInfo.szPsw, DH_MAX_CARDPWD_LEN - 1);
pStrCardInfo->emStatus = NET_ACCESSCTLCARD_STATE_NORMAL;
pStrCardInfo->emType = NET_ACCESSCTLCARD_TYPE_GENERAL;
pStrCardInfo->nUserTime = m_CardInfo.nUserTime;
pStrCardInfo->bFirstEnter = TRUE;
pStrCardInfo->blsValid = TRUE;
pStrCardInfo->stuValidStartTime = m_CardInfo.stuValidStartTime;
pStrCardInfo->stuValidEndTime = m_CardInfo.stuValidEndTime;
pStrCardInfo->nRecNo = m_CardInfo.nRecNo;

// DoorNum is 2, it indicates the two doors of the gate.
pStrCardInfo->nDoorNum = 2;
pStrCardInfo->sznDoors[0] = 0;
pStrCardInfo->sznDoors[1] = 1;
// Control the valid time of the opening door, 255 indicates all day.
pStrCardInfo->nTimeSectionNum = 2;
pStrCardInfo->sznTimeSectionNo[0] = 255;
pStrCardInfo->sznTimeSectionNo[1] = 255;

stuInParam.nBufLen = sizeof(NET_RECORDSET_ACCESS_CTL_CARD);
stuInParam.pBuf = (void*)pStrCardInfo;

BOOL bRet = CLIENT_ControlDevice(m_ILoginID, DH_CTRL_RECORDSET_UPDATEEX, &stuInParam,
DEFAULT_WAIT_TIME);
if (FALSE == bRet)
{
    printf("CLIENT_ControlDevice modify cardinfo fail");
    return;
}

// Delete the information of access control card
NET_CTRL_RECORDSET_PARAM stuInParam = sizeof(stuInParam);
stuInParam.emType = NET_RECORD_ACCESSCTLCARD;
stuInParam.pBuf = &pCardInfo->nRecNo;

```

```

stulnParam.nBufLen = sizeof(int);
BOOL bRet = CLIENT_ControlDevice(m_lLoginID, DH_CTRL_RECORDSET_REMOVE, &stulnParam,
DEFAULT_WAIT_TIME);
if (FALSE == bRet)
{
    printf("CLIENT_ControlDevice delete cardinfo fail\n");
    return;
}

// Clean up the information of access control card (Delete all information of access control card)
NET_CTRL_RECORDSET_PARAM stulnParam = sizeof(stulnParam);
stulnParam.emType = NET_RECORD_ACCESSCTLCARD;

BOOL bRet = CLIENT_ControlDevice(m_lLoginID, DH_CTRL_RECORDSET_CLEAR, &stulnParam,
DEFAULT_WAIT_TIME);
if (FALSE == bRet)
{
    printf("CLIENT_ControlDevice clear cardinfo fail\n");
    return;
}

```

8.3 Face Management

8.3.1 Introduction

Add, modify, delete and clean up the face picture.

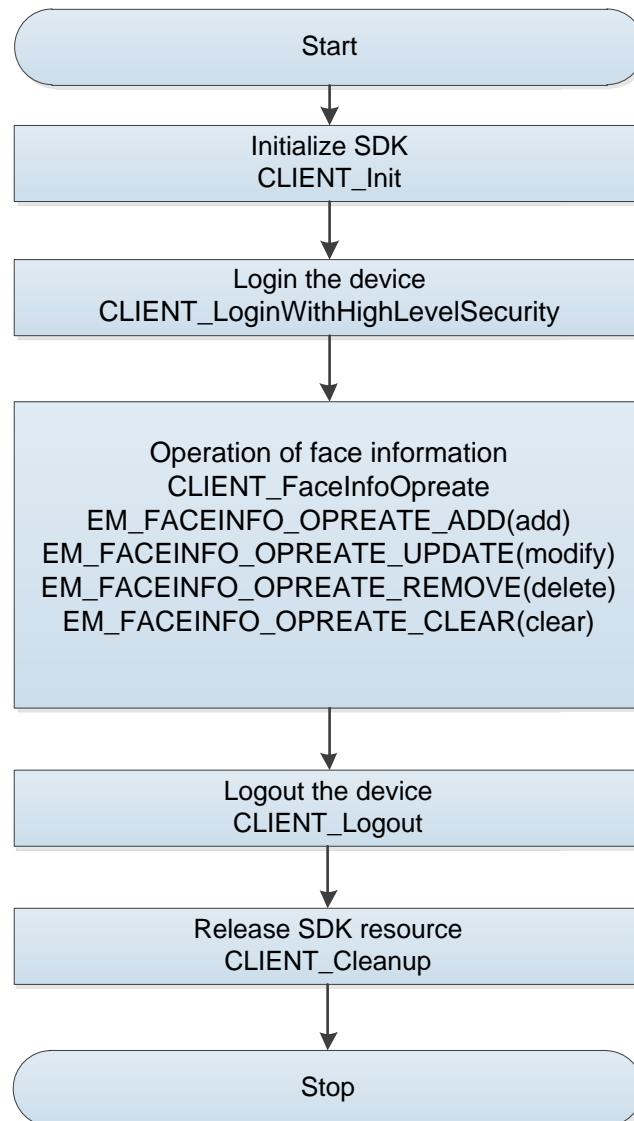
8.3.2 Interface Overview

Table 8-2 Interfaces of face management

Interface	Implication
CLIENT_FaceInfoOpreate	Add, delete, modify and clean up the information of face picture.
CLIENT_FindRecord	Set searching conditions.
CLIENT_FindNextRecord	Search data in the current searching condition.
CLIENT_FindRecordClose	Close searching.
CLIENT_ControlDevice	Add, delete, modify and clean up the people information.

8.3.3 Process

Figure 8-2 Adding/deleting/modifying/cleaning up the information of face picture



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.
- Step 3 Call **CLIENT_FaceInfoOpreate** to add, modify and delete the face according to enumeration type.
- Step 4 After using the function module, call **CLIENT_Logout** to logout the device.
- Step 5 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Face information adding: Use EM_FACEINFO_OPREATE_ADD as the enumeration.
- Face information modifying: Use EM_FACEINFO_OPREATE_UPDATE as the enumeration.
- Face information deleting: Use EM_FACEINFO_OPREATE_REMOVE as the enumeration.

- Face information cleaning up: Use EM_FACEINFO_OPREATE_CLEAR as the enumeration.
- The face information modifying and deleting are according to the UserID of the access control card.

8.3.4 Example Code

```
// Add the information of face picture
NET_IN_ADD_FACE_INFO stuInParam = sizeof(NET_IN_ADD_FACE_INFO);
strncpy(stuInParam.szUserID, m_StuAddCardInfo.szUserID, DH_MAX_USERID_LEN - 1);
stuInParam.stuFaceInfo.nFacePhoto = 1;

FILE *fPic = fopen(m_szFilePath, "rb");
fseek(fPic, 0, SEEK_END);
int nLength = ftell(fPic);
rewind(fPic);

stuInParam.stuFaceInfo.nFacePhotoLen[0] = nLength;
stuInParam.stuFaceInfo.pszFacePhoto[0] = new char[nLength];

memset(stuInParam.stuFaceInfo.pszFacePhoto[0], 0, nLength);
int nReadLen = fread(stuInParam.stuFaceInfo.pszFacePhoto[0], 1, nLength, fPic);
fclose(fPic);
fPic = NULL;

NET_OUT_ADD_FACE_INFO stuOutParam = sizeof(stuOutParam);

BOOL bRet = CLIENT_FaceInfoOpreate(m_ILoginID, EM_FACEINFO_OPREATE_ADD, (void*)&stuInParam,
(void*)&stuOutParam, DEFAULT_WAIT_TIME);
if (bRet == FALSE)
{
    printf("CLIENT_FaceInfoOpreate add face fail");
}

if (fPic)
{
    fclose(fPic);
    fPic = NULL;
}

if (stuInParam.stuFaceInfo.pszFacePhoto[0])
{
    delete[] stuInParam.stuFaceInfo.pszFacePhoto[0];
}
```

```

        stuInParam.stuFaceInfo.pszFacePhoto[0] = NULL;
    }
    // Modify the information of face picture
    NET_IN_UPDATE_FACE_INFO stuInParam = sizeof(stuInParam);
    strncpy(stuInParam.szUserID, m_CardInfo.szUserID, sizeof(m_CardInfo.szUserID) - 1);
    stuInParam.stuFaceInfo.nFacePhoto = 1;

    FILE *fPic = fopen(m_szFilePath, "rb");

    fseek(fPic, 0, SEEK_END);
    int nLength = ftell(fPic);
    rewind(fPic);

    stuInParam.stuFaceInfo.nFacePhotoLen[0] = nLength;
    stuInParam.stuFaceInfo.pszFacePhoto[0] = new char[nLength];

    memset(stuInParam.stuFaceInfo.pszFacePhoto[0], 0, nLength);
    int nReadLen = fread(stuInParam.stuFaceInfo.pszFacePhoto[0], 1, nLength, fPic);
    fclose(fPic);
    fPic = NULL;

    NET_OUT_UPDATE_FACE_INFO stuOutParam;
    memset(&stuOutParam, 0, sizeof(stuOutParam));
    stuOutParam.dwSize = sizeof(stuOutParam);

    BOOL bRet = CLIENT_FaceInfoOpreate(m_LoginID, EM_FACEINFO_OPREATE_UPDATE, (void*)&stuInParam,
    (void*)&stuOutParam, DEFAULT_WAIT_TIME);
    if (bRet == FALSE)
    {
        printf ("CLIENT_FaceInfoOpreate modify face fail");
    }

    if (fPic)
    {
        fclose(fPic);
        fPic = NULL;
    }
    if (stuInParam.stuFaceInfo.pszFacePhoto[0])
    {
        delete[] stuInParam.stuFaceInfo.pszFacePhoto[0];
    }

```

```

        stuInParam.stuFaceInfo.pszFacePhoto[0] = NULL;
    }

    // Delete the face picture
    NET_IN_REMOVE_FACE_INFO stuInParam = sizeof(stuInParam);
    // Delete the face picture according to the UserID of the information of access control card.
    strncpy(stuInParam.szUserID, cardInfo.szUserID, DH_MAX_USERID_LEN - 1);

    NET_OUT_REMOVE_FACE_INFO stuOutParam;
    memset(&stuOutParam, 0, sizeof(stuOutParam));
    stuOutParam.dwSize = sizeof(stuOutParam);

    bRet = CLIENT_FaceInfoOpreate(m_lLoginID, EM_FACEINFO_OPREATE_REMOVE, (void*)&stuInParam,
    (void*)&stuOutParam, DEFAULT_WAIT_TIME);
    if (bRet == FALSE)
    {
        printf ("CLIENT_FaceInfoOpreate delete face fail");
    }

    // Clean up the information of face picture(Delete all information of face picture)
    NET_IN_CLEAR_FACE_INFO stuInParam = sizeof(stuInParam);
    NET_OUT_CLEAR_FACE_INFO stuOutParam;
    memset(&stuOutParam, 0, sizeof(stuOutParam));
    stuOutParam.dwSize = sizeof(stuOutParam);

    BOOL bRet = CLIENT_FaceInfoOpreate(m_lLoginID, EM_FACEINFO_OPREATE_CLEAR, (void*)&stuInParam,
    (void*)&stuOutParam, DEFAULT_WAIT_TIME);
    if (bRet == FALSE)
    {
        printf ("CLIENT_FaceInfoOpreate clear face fail");
    }
}

```

8.4 Searching for the Record of In-Out the Door

8.4.1 Introduction

Search the record list of access control. The searching information includes card No., user serial number, the status of opening the door, card type, the mode of opening the door and time.

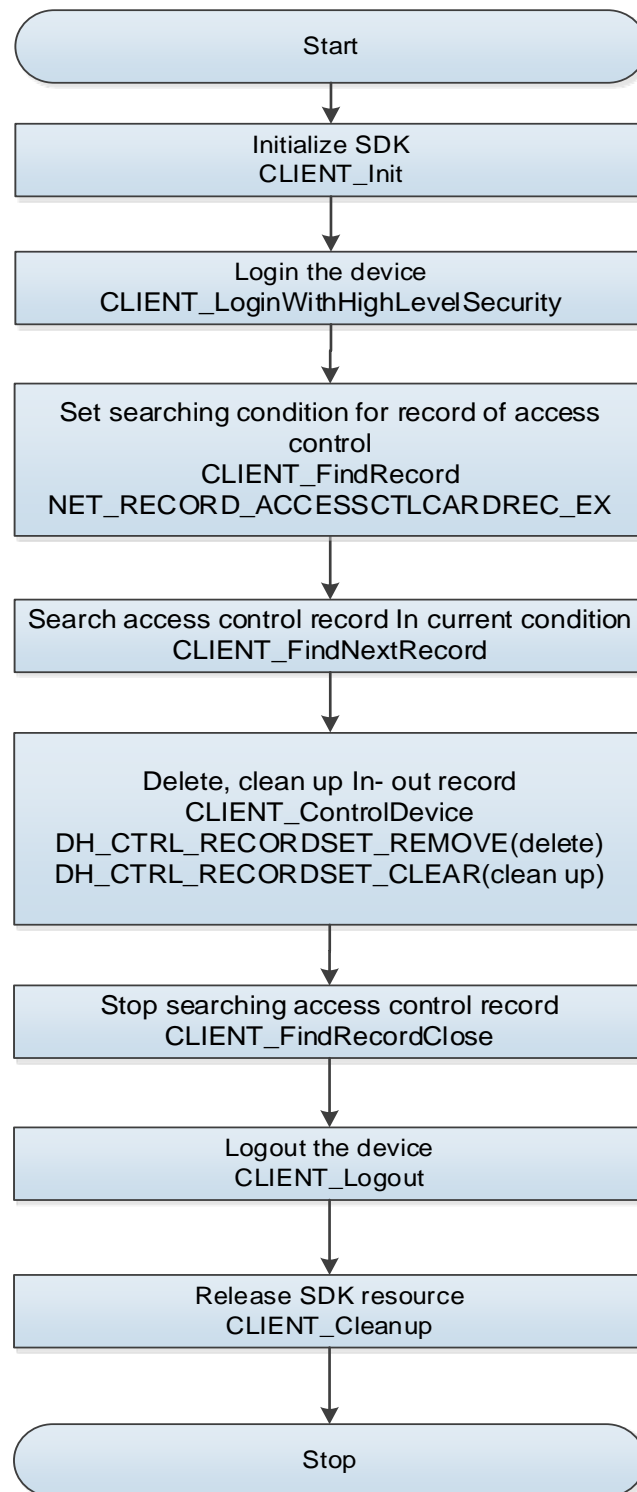
8.4.2 Interface Overview

Table 8-3 Interfaces of searching the record of in-out the door

Interface	Implication
CLIENT_FindRecord	Set searching conditions.
CLIENT_FindNextRecord	Search data in the current searching condition.
CLIENT_FindRecordClose	Close searching.
CLIENT_ControlDevice	Delete and clean up the record of access control.

8.4.3 Process

Figure 8-3 Searching/deleting/cleaning up the record of access control



Process Description

- Step 1 Call **CLIENT_Init** to initialize SDK.
- Step 2 Call CLIENT_LoginWithHighLevelSecurity to login the device.
- Step 3 Call **CLIENT_FindRecord** to set searching conditions for the record of access control.
- Step 4 Call **CLIENT_FindNextRecord** to search the specified number of the record of access control

in the current searching conditions.

Step 5 Call **CLIENT_ControlDevice** to delete and clean up the record of access control.

Step 6 After searching, call **CLIENT_FindRecordClose** to clean up the searching resource Information.

Step 7 After using the function module, call **CLIENT_Logout** to logout the device.

Step 8 After using all SDK functions, call **CLIENT_Cleanup** to release SDK resource.

Notes for Process

- Access control record deleting: Use DH_CTRL_RECORDSET_REMOVE as the enumeration.
- Access control record cleaning up: Use DH_CTRL_RECORDSET_CLEAR as the enumeration.

8.4.4 Example Code

About the codes, see "8.2.4.1 Searching the Information of Access Control Card".

The example code for deleting and cleaning up the access control records, see "8.2.4.2 Adding/Deleting/Modifying/Searching the Information of Access Control Card".

9 Object Monitoring

9.1 Subscribing to Object Monitoring Events

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events through the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_OBJECT_PLACEMENT_DETECTION`: Object placement detection event
- `EVENT_IVS_OBJECT_REMOVAL_DETECTION`: Object removal detection event
- `EVENT_IVS_TRASH_WITHOUT_COVER_DETECTION`: Detection events of uncovered garbage can

9.2 Searching for and Downloading Object Monitoring

Video

Object monitoring belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playingback/Downloading Video and Picture”.

Call `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of object monitoring are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
 - ◇ `EVENT_IVS_OBJECT_PLACEMENT_DETECTION`: Video search of object placement
 - ◇ `EVENT_IVS_OBJECT_REMOVAL_DETECTION`: Video search of object removal
 - ◇ `EVENT_IVS_TRASH_WITHOUT_COVER_DETECTION`: Video search of uncovered garbage can

10 City Management

10.1 Subscribing to management events

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events through the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_CITY_MOTORPARKING`: Illegal parking of motor vehicles
- `EVENT_IVS_CITY_NONMOTORPARKING`: Illegal parking of non-motor vehicles
- `EVENT_IVS_DOOR_FRONT_DIRTY`: Dirty front door
- `EVENT_IVS_DUSTBIN_OVER_FLOW`: Full garbage can
- `EVENT_IVS_FLOWBUSINESS`: Mobile vendor
- `EVENT_IVS_GARBAGE_EXPOSURE`: Exposed garbage
- `EVENT_IVS_HOLD_UMBRELLA`: Illegal umbrella
- `EVENT_IVS_HUDDLE_MATERIAL`: Piles of material
- `EVENT_IVS_OUTDOOR_ADVERTISEMENT`: Outdoor advertisement
- `EVENT_IVS_SHOPPRESENCE`: Roadside stall
- `EVENT_IVS_SHOP_SIGN_ABNORMAL`: Abnormal store sign
- `EVENT_IVS_SHOP_WINDOW_POST`: Window posting
- `EVENT_IVS_STREET_SUNCURE`: Drying along the street

10.2 Searching for and Downloading City Management

Video

City management belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playing/Downloading Video and Picture”.

Call the interface `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of city management are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
 - ◇ `EVENT_IVS_CITY_MOTORPARKING`: Video search of illegal parking of motor vehicles
 - ◇ `EVENT_IVS_CITY_NONMOTORPARKING`: Video search of illegal parking of non-motor vehicles
 - ◇ `EVENT_IVS_DOOR_FRONT_DIRTY`: Video search of dirty front door
 - ◇ `EVENT_IVS_DUSTBIN_OVER_FLOW`: Video search of full garbage can
 - ◇ `EVENT_IVS_FLOWBUSINESS`: Video search of of mobile vendor
 - ◇ `EVENT_IVS_GARBAGE_EXPOSURE`: Video search of exposed garbage
 - ◇ `EVENT_IVS_HOLD_UMBRELLA`: Video search of illegal umbrella.
 - ◇ `EVENT_IVS_HUDDLE_MATERIAL`: Video search of piles of material
 - ◇ `EVENT_IVS_OUTDOOR_ADVERTISEMENT`: Video search of outdoor advertisement
 - ◇ `EVENT_IVS_SHOPPRESENCE`: Video search of roadside stall
 - ◇ `EVENT_IVS_SHOP_SIGN_ABNORMAL`: Video search of abnormal store sign
 - ◇ `EVENT_IVS_SHOP_WINDOW_POST`: Video search of window posting

◇ EVENT_IVS_STREET_SUNCURE: Video search of drying along the street

11 Parking Space Detection

11.1 Subscribing to Detection Event of Parking Space

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events through the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_CAR_DRIVING_IN_OUT`: Vehicle entering or exiting status
- `EVENT_IVS_PARKING_LOT_STATUS_DETECTION`: Status detection of outdoor parking space
- `EVENT_IVS_PARKINGSPACE_STATUS`: Parking space status
- `EVENT_IVS_TRAFFIC_PARKINGSPACE_MANUALSNAP`: Manual snapshot of roadside parking space
- `EVENT_IVS_TRAFFIC_PARKINGSPACENOPARKING`: Parking space not occupied
- `EVENT_IVS_TRAFFIC_PARKINGSPACEOVERLINE`: Cross the parking line
- `EVENT_IVS_TRAFFIC_PARKINGSPACEPARKING`: Parking space occupied
- `EVENT_IVS_TRAFFIC_ANALYSE_PRESNAP`: Pre-analyze snapshot

11.2 Searching for and Downloading Event Detection Video of Parking Space

Parking space detection belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playing/Downloading Video and Picture”.

Call the interface `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of parking space detection event are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
 - ◇ `EVENT_IVS_CAR_DRIVING_IN_OUT`: Video search of vehicle entering or exiting status.
 - ◇ `EVENT_IVS_PARKING_LOT_STATUS_DETECTION`: Video search of outdoor parking space status
 - ◇ `EVENT_IVS_PARKINGSPACE_STATUS`: Video search of parking space status.
 - ◇ `EVENT_IVS_TRAFFIC_PARKINGSPACE_MANUALSNAP`: Video search of manual snapshot of roadside parking space
 - ◇ `EVENT_IVS_TRAFFIC_PARKINGSPACENOPARKING`: Video search of parking space not occupied
 - ◇ `EVENT_IVS_TRAFFIC_PARKINGSPACEOVERLINE`: Video search of parking line crossing
 - ◇ `EVENT_IVS_TRAFFIC_PARKINGSPACEPARKING`: Video search of occupied parking space
 - ◇ `EVENT_IVS_TRAFFIC_ANALYSE_PRESNAP`: Video search of pre-analyzed snapshot

11.3 Subscribing to Designated Parking Space Picture

11.3.1 Overview

Subscribe to or unsubscribe from the designated parking space picture through the designated channel number and picture ID.

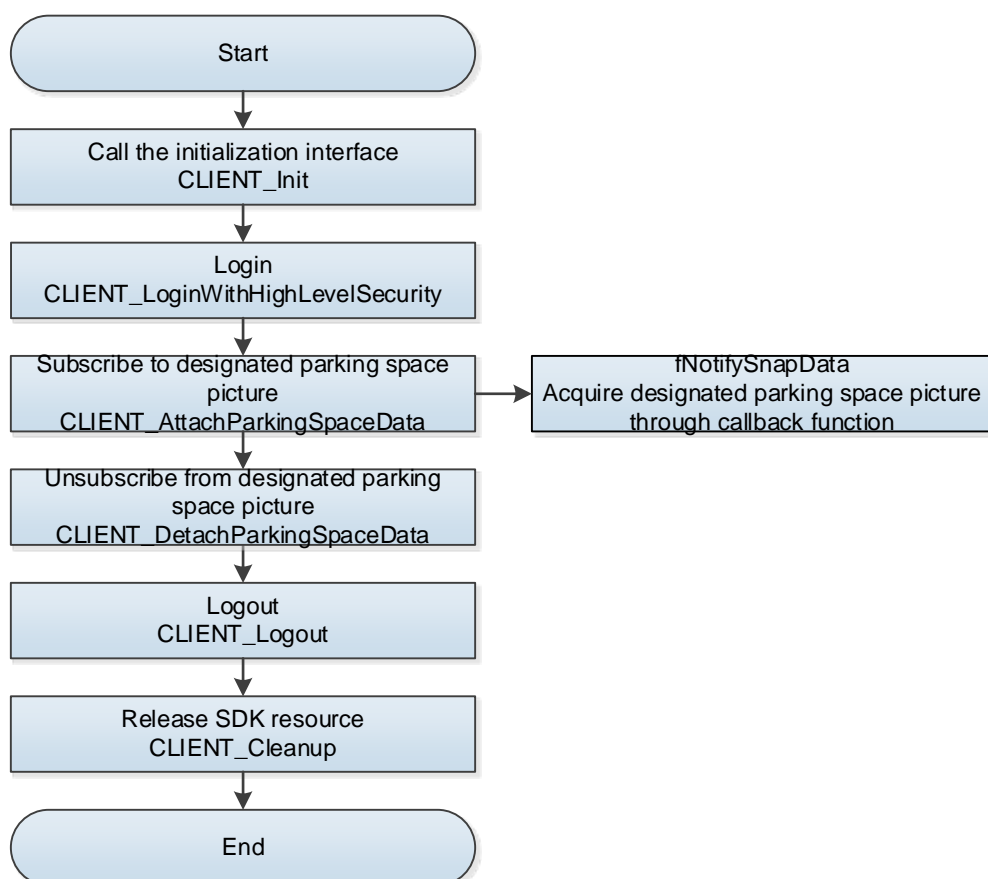
11.3.2 Interface Overview

Table 11-1 Interfaces related to subscription of designated parking space picture

Interface	Description
CLIENT_AttachParkingSpaceData	Subscribe to designated parking space picture
CLIENT_DetachParkingSpaceData	Unsubscribe from designated parking space picture

11.3.3 Process Description

Figure 11-1 Subscribe to or unsubscribe from designated parking space picture



Process Description

Step 1 Call CLIENT_Init to initialize SDK.

- Step 2** After successful initialization, call CLIENT_LoginWithHighLevelSecurity to log in to the device.
- Step 3** Call CLIENT_AttachParkingSpaceData to subscribe to the designated parking space picture from the device.
- Step 4** After the subscription is successful, the parking picture reported by the device is obtained and notified to the user through the fNotifySnapData callback function.
- Step 5** After the parking space picture is reported, call CLIENT_DetachParkingSpaceData to unsubscribe from the designated parking space picture.
- Step 6** After using the function module, call CLIENT_Logout to log out of the device.
- Step 7** After using all SDK functions, call CLIENT_Cleanup to release SDK resource.

11.3.4 Example Code

```
void CALLBACK NotifySnapData(LLONG IParkingHandle, NET_CB_PARKINGSPACE_DATA*
pDiagnosisInfo, void* pBuf, int nBufLen, LDWORD dwUser)
{
    // Process the callback data
}

NET_IN_ATTACH_PARKINGSPACE InParam = {sizeof (NET_IN_ATTACH_PARKINGSPACE)};
NET_OUT_ATTACH_PARKINGSPACE OutParam = {sizeof (NET_OUT_ATTACH_PARKINGSPACE)};
InParam.nChannel=0;
InParam.cbNotifySnapData = NotifySnapData; // Subscribe to callback function

// Subscribe to designated parking space picture
LLONG attachHnd = CLIENT_AttachParkingSpaceData (ILoginID,&InParam,&OutParam,3000)
if(0 == attachHnd)
{
    printf("CLIENT_AttachParkingSpaceData failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

// Unsubscribing from heat map data
CLIENT_DetachParkingSpaceData (attachHnd);
```

12 Crowd Map

12.1 Subscribing to Crowd Map Event

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events in the callback function `fAnalyzerDataCallBack` reported by intelligent events:

EVENT_IVS_CROWDDETECTION: Crowd density detection

12.2 Searching for and Downloading Video of Crowd Map

Crowd map event belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playing/Downloading Video and Picture”.

Call the interface `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of crowd map event are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
EVENT_IVS_CROWDDETECTION: Video search of crowd density detection.

13 Vehicle Density Map

13.1 Subscribing to Vehicle Density Event

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events through the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_CONGESTION_DETECTION`: Vehicle congestion alarm event (road scenarios)
- `EVENT_IVS_VEHICLELIMIT_DETECTION`: Upper limit alarm of parking vehicles (parking lot scenarios)

13.2 Searching for and Downloading Video of Vehicle Density Event

Vehicle density event belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playing/Downloading Video and Picture”.

Call the interface `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of vehicle density event are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
 - ◇ `EVENT_IVS_CONGESTION_DETECTION`: Video search of vehicle congestion alarm (road scenarios).
 - ◇ `EVENT_IVS_VEHICLELIMIT_DETECTION`: Video search of upper limit alarm of parking vehicles (parking lot scenarios).

14 Heat Map

14.1 Subscribing to Heat Map Data

14.1.1 Overview

Subscribe to or unsubscribe from heat map

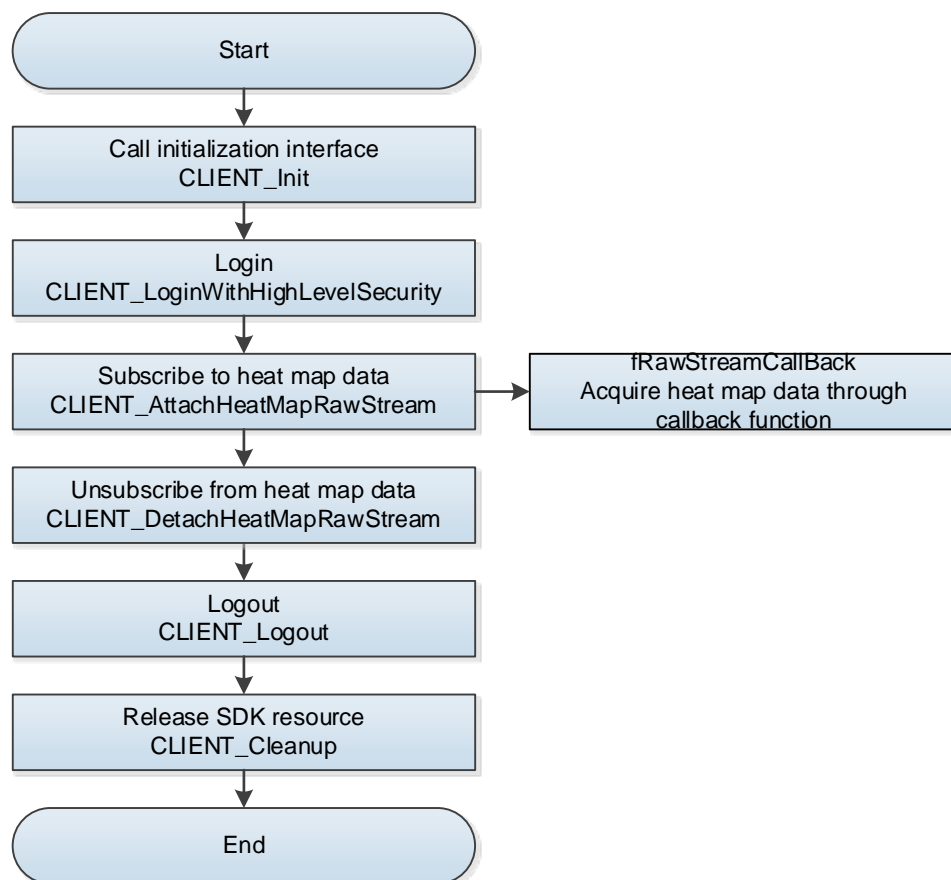
14.1.2 Interface Overview

Table 14-1 Interface description of heat map

Interface	Description
CLIENT_AttachHeatMapRawStream	Subscribe to original data of heat map
CLIENT_DetachHeatMapRawStream	Unsubscribe from original data of heat map.

14.1.3 Process Description

Figure 14-1 Subscribe to or unsubscribe from heat map data



Process Description

- Step 1 Call CLIENT_Init to initialize SDK.
- Step 2 After successful initialization, call CLIENT_LoginWithHighLevelSecurity to log in to the device.
- Step 3 Call CLIENT_AttachHeatMapRawStream to subscribe to heat map data from device.
- Step 4 After the subscription is successful, the heat map data reported by the device is notified to the user through the fRawStreamCallBack callback function.
- Step 5 After the original data of heat map is reported, call CLIENT_DetachHeatMapRawStream to unsubscribe from the heat map data.
- Step 6 After using the function module, call CLIENT_Logout to log out of the device.
- Step 7 After using all SDK functions, call CLIENT_Cleanup to release SDK resource.

14.1.4 Example Code

```
void CALLBACK RawStreamCallBack(LLONG lAttachHandle, NET_RAWSTREAM_NOTIFY_INFO* pBuf,
DWORD dwBufLen, LDWORD dwUser)
{
    // Process callback data
}

NET_IN_RAWSTREAM_ATTACH_INFO InParam = {sizeof(NET_IN_RAWSTREAM_ATTACH_INFO)};
NET_OUT_RAWSTREAM_ATTACH_INFO OutParam = {sizeof(NET_OUT_RAWSTREAM_ATTACH_INFO)};
InParam.nChannel=0;
InParam.cbNotify= RawStreamCallBack; // Subscribe to callback function

// Subscribe to heat map data
LLONG attachHnd = CLIENT_AttachHeatMapRawStream(lLoginID,&InParam,&OutParam,3000)
if(0 == attachHnd)
{
    printf("CLIENT_AttachHeatMapRawStream failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

// Unsubscribe from heat map data
CLIENT_DetachHeatMapRawStream(attachHnd);
```

14.2 Subscribing to Gray Map Data

14.2.1 Overview

Subscribe to or unsubscribe from gray map

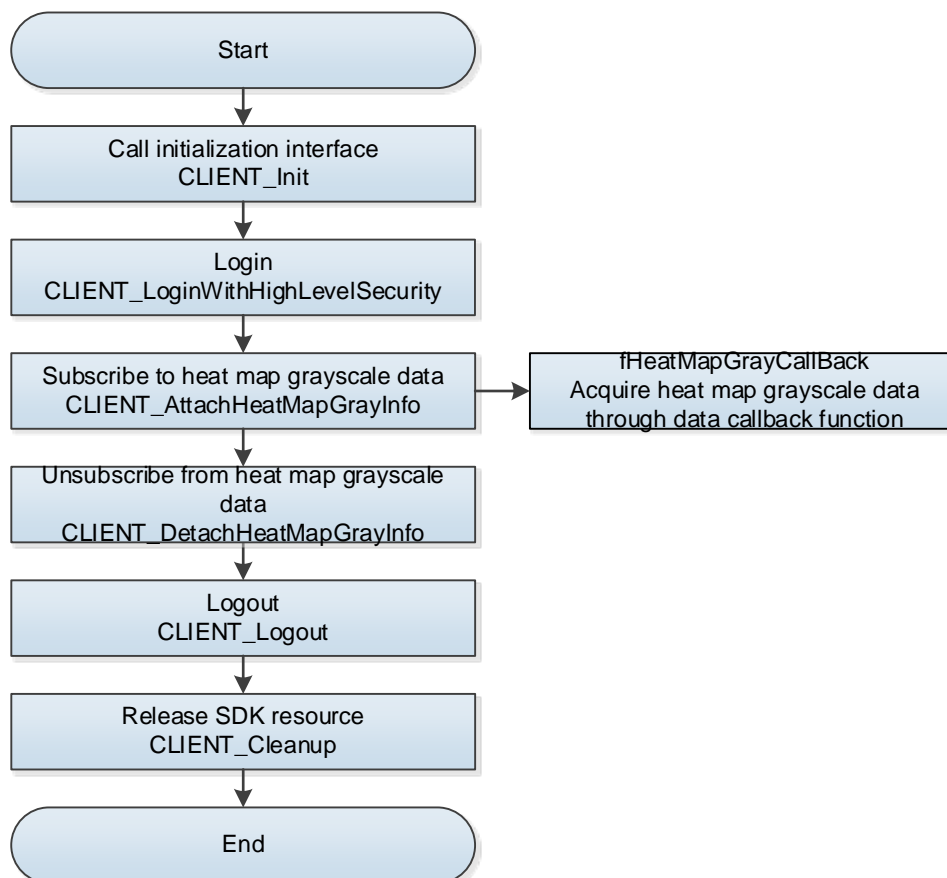
14.2.2 Interface Overview

Table 14-2 Interface description of gray map

Interface	Description
CLIENT_AttachHeatMapGrayInfo	Subscribe to the grayscale data of heat map
CLIENT_DetachHeatMapGrayInfo	Unsubscribe to the grayscale data of heat map

14.2.3 Process Description

Figure 14-2 Subscribe to or unsubscribe from grayscale data of heat map



Process Description

- Step 1 Call CLIENT_Init to initialize SDK.
- Step 2 After successful initialization, call CLIENT_LoginWithHighLevelSecurity to log in to the device.
- Step 3 Call CLIENT_AttachHeatMapRawStream to subscribe to the grayscale data of heat map from device.
- Step 4 After the subscription is successful, the grayscale data of heat map reported by the device is notified to the user through the fHeatMapGrayCallBack callback function.
- Step 5 After grayscale data of heat map is reported, call CLIENT_DetachHeatMapGrayInfo to unsubscribe from the grayscale data of heat map.
- Step 6 After using the function module, call CLIENT_Logout to log out of the device.
- Step 7 After using all SDK functions, call CLIENT_Cleanup to release SDK resource.

14.2.4 Example Code

```
void CALLBACK HeatMapGrayCallBack(LLONG IAttachHandle, NET_CB_HEATMAP_GRAY_INFO*
pstGrayInfo, LDWORD dwUser)
{
    // Process the callback data
}

NET_IN_GRAY_ATTACH_INFO InParam = {sizeof (NET_IN_GRAY_ATTACH_INFO)};
NET_OUT_GRAY_ATTACH_INFO OutParam = {sizeof(NET_OUT_GRAY_ATTACH_INFO)};
InParam.nChannel=0;
InParam.cbHeatMapGray = HeatMapGrayCallBack; // Subscribe to callback function

// Subscribe to heat map data
LLONG attachHnd = CLIENT_AttachHeatMapGrayInfo (ILoginID,&InParam,&OutParam,3000)
if(0 == attachHnd)
{
    printf("CLIENT_AttachHeatMapGrayInfo failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

// Unsubscribe from heat map data
CLIENT_DetachHeatMapGrayInfo (attachHnd);
```

15 Stereo Analysis

15.1 Subscribing to Stereo Analysis Event

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events in the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_BACK_TO_DETECTION`: Back detection
- `EVENT_IVS_BIG_BAGGAGE_DETECTION`: Luggage detection
- `EVENT_IVS_DISTANCE_DETECTION`: Abnormal spacing detection
- `EVENT_IVS_PRAM_DETECTION`: Baby stroller detection
- `EVENT_IVS_ABNORMALRUNDETECTION`: Abnormal running detection
- `EVENT_IVS_STEREO_DISTANCE_DETECTION`: Abnormal spacing of stereo analysis detection
- `EVENT_IVS_TICKET_EVADE_DETECTION`: Fare evasion detection
- `EVENT_IVS_WALK_DETECTION`: Walking detection
- `EVENT_IVS_WRITE_ON_THE_BOARD_DETECTION`: Blackboard writing detection

15.2 Searching for and Downloading Video of Stereo Analysis Event

Stereo analysis belongs to intelligent event. For details on related video and picture search, see “2.5 Searching for/Playing/Downloading Video and Picture”.

Call the interface `CLIENT_FindFileEx` to set the search conditions. The video search and interface parameters of stereo analysis are as follows:

- `emType`: `DH_FILE_QUERY_FILE`: Search for video.
- `pQueryCondition`: `NET_IN_MEDIA_QUERY_FILE` structure pointer. The `nEventLists` field in the structure is:
 - ◇ `EVENT_IVS_BACK_TO_DETECTION`: Video search of back detection
 - ◇ `EVENT_IVS_BIG_BAGGAGE_DETECTION`: Video search of luggage detection
 - ◇ `EVENT_IVS_DISTANCE_DETECTION`: Video search of abnormal spacing
 - ◇ `EVENT_IVS_PRAM_DETECTION`: Video search of baby stroller detection
 - ◇ `EVENT_IVS_ABNORMALRUNDETECTION`: Video search of abnormal running
 - ◇ `EVENT_IVS_STEREO_DISTANCE_DETECTION`: Video search of abnormal spacing of stereo analysis.
 - ◇ `EVENT_IVS_TICKET_EVADE_DETECTION`: Video search of fare evasion
 - ◇ `EVENT_IVS_WALK_DETECTION`: Video search of walking detection
 - ◇ `EVENT_IVS_WRITE_ON_THE_BOARD_DETECTION`: Video search of blackboard writing detection.

15.3 Subscribing to Video Statistics

15.3.1 Overview

Subscribe to or unsubscribe from the video statistics of stereo analysis.

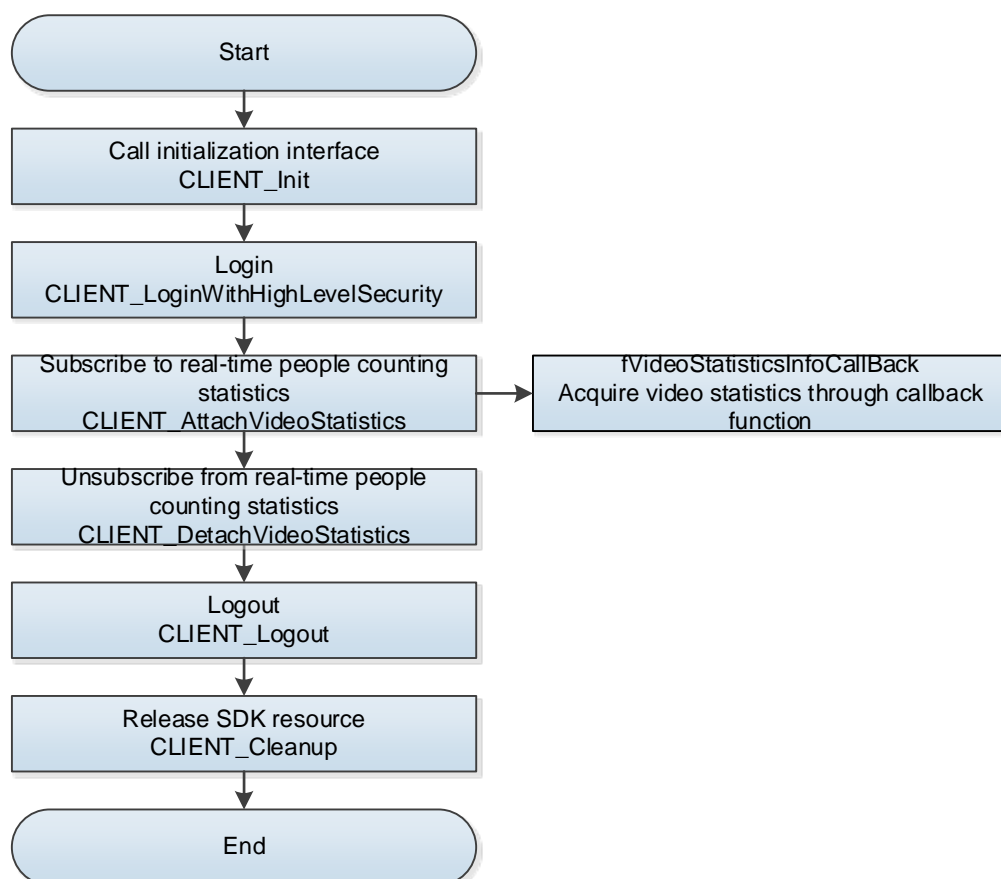
15.3.2 Interface Overview

Table 15-1 Interface description of video statistics

Interface	Description
CLIENT_AttachVideoStatistics	Subscribe to real-time people counting statistics.
CLIENT_DetachVideoStatistics	Unsubscribe from real-time people counting statistics.

15.3.3 Process Description

Figure 15-1 Subscribe to or unsubscribe from real-time people counting statistics



Process Description

Step 1 Call CLIENT_Init to initialize SDK.

Step 2 After successful initialization, call CLIENT_LoginWithHighLevelSecurity to log in to the

device.

Step 3 Call CLIENT_AttachVideoStatistics to subscribe to real-time people counting statistics from device.

Step 4 After the subscription is successful, the real-time people counting statistics reported by the device is notified to the user through the fVideoStatisticsInfoCallBack callback function.

Step 5 After real-time people counting statistics is reported, call CLIENT_DetachVideoStatistics to unsubscribe from the real-time people counting statistics.

Step 6 After using the function module, call CLIENT_Logout to log out of the device.

Step 7 After using all SDK functions, call CLIENT_Cleanup to release SDK resource.

15.3.4 Example Code

```
void CALLBACK VideoStatisticsInfoCallBack (LLONG IAttachHandle, NET_EM_VS_TYPE emType, void*
pBuf, DWORD dwBufLen, LDWORD dwUser)
{
    // Process the callback data
}

NET_IN_ATTACH_VIDEO_STATISTICS InParam = {sizeof (NET_IN_ATTACH_VIDEO_STATISTICS)};
NET_OUT_ATTACH_VIDEO_STATISTICS OutParam = {sizeof(NET_OUT_ATTACH_VIDEO_STATISTICS)}
InParam.nChannel=0;
InParam.cbCallBack = VideoStatisticsInfoCallBack; // Subscribe to callback function

// Subscribe to real-time people counting statistics
LLONG attachHnd = CLIENT_AttachVideoStatistics (ILoginID,&InParam,&OutParam,3000)
if(0 == attachHnd)
{
    printf("CLIENT_AttachVideoStatistics failed! Error code %x.\n", CLIENT_GetLastError());
    return;
}

// Unsubscribe from real-time people counting statistics
CLIENT_DetachVideoStatistics(attachHnd);
```


16 Helme Detection

16.1 Subscribing to Helme Detection Event

For details, see “2.4 Subscribing to Intelligent Event”. Filter out general behavior events through the callback function `fAnalyzerDataCallBack` reported by intelligent events:

- `EVENT_IVS_HELMET_DETECTION`: Helme Detection event

17 Interface Definition

17.1 SDK Initialization

17.1.1 SDK CLIENT_Init

Table 17-1 Initialize SDK

Item	Description	
Name	Initialize SDK.	
Function	BOOL CLIENT_Init(fDisconnect cbDisconnect, LDWORD dwUser);	
Parameter	[in]cbDisconnect	Disconnection callback.
	[in]dwUser	User parameter of disconnection callback.
Return value	Success: TRUE. Failure: FALSE.	
Note	The precondition for calling other function modules of SDK. The callback will not send to the user after the device is disconnected if the callback is set as NULL.	

17.1.2 CLIENT_Cleanup

Table 17-2 Clean up SDK

Item	Description
Name	Clean up SDK.
Function	void CLIENT_Cleanup()
Parameter	None.
Return value	None.
Note	Call SDK cleanup interface before the process stops.

17.1.3 CLIENT_SetAutoReconnect

Table 17-3 Set reconnection callback

Item	Description	
Name	Set auto reconnection callback.	
Function	void CLIENT_SetAutoReconnect(fHaveReConnect cbAutoConnect, LDWORD dwUser);	
Parameter	[in]cbAutoConnect	Reconnection callback.
	[in]dwUser	User parameter of disconnection callback.
Return value	None.	
Note	Set the reconnection callback interface. If the callback is set as NULL, it will not connect automatically.	

17.1.4 CLIENT_SetNetworkParam

Table 17-4 Set network parameter

Item	Description	
Name	Set the related parameters for network environment.	
Function	void CLIENT_SetNetworkParam(NET_PARAM *pNetParam);	
Parameter	[in]pNetParam	Parameters such as network delay, reconnection times and cache size.
Return value	None.	
Note	Adjust the parameters according to the actual network environment.	

17.2 Device Login

17.2.1 CLIENT_LoginWithHighLevelSecurity

Table 17-5 Log in with high level security

Item	Description	
Name	Login the device with high level security.	
Function	LLONG CLIENT_LoginWithHighLevelSecurity (NET_IN_LOGIN_WITH_HIGHLEVEL_SECURITY* pstInParam, NET_OUT_LOGIN_WITH_HIGHLEVEL_SECURITY* pstOutParam);	
Parameter	[in] pstInParam	[in] dwSize
		[in] szIP
		[in] nPort
		[in] szUserName
		[in] szPassword

Item	Description	
		[in] emSpecCap
		[in] pCapParam
	[out] pstOutParam	[in] dwSize
		[out] stuDeviceInfo
		[out] nError
Return value	<ul style="list-style-type: none"> ● Success: Not 0. ● Failure: 0. 	
Note	Login the device with high level security. CLIENT_LoginEx2 can still be used, but there are security risks, so it is highly recommended to use the latest interface CLIENT_LoginWithHighLevelSecurity to log in to the device.	

Table 17-6 Error code and meaning

Error code	Meaning
1	Wrong password.
2	The user name does not exist.
3	Login timeout.
4	The account has logged in.
5	The account has been locked.
6	The account is in the blocklist.
7	The device resource is insufficient and the system is busy.
8	Sub connection failed.
9	Main connection failed.
10	Exceeds the maximum allowed number of user connections.
11	Lack avnetsdk or the dependent libraries of avnetsdk.
12	USB flash disk is not inserted into device, or the USB flash disk information error.
13	The IP at client is not authorized for login.

17.2.2 CLIENT_Logout

Table 17-7 Log out

Item	Description	
Name	Logout the device.	
Function	<pre> BOOL CLIENT_Logout(LLONG ILoginID); </pre>	
Parameter	[in] ILoginID	Return value of CLIENT_LoginWithHighLevelSecurity.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.3 Real-time Monitoring

17.3.1 CLIENT_RealPlayEx

Table 17-8 Start the real-time monitoring

Item	Description	
Name	Open the real-time monitoring.	
Function	LLONG CLIENT_RealPlayEx(LLONG ILoginID, int nChannelID, HWND hWnd, DH_RealPlayType rType);	
Parameter	[in]ILoginID	Return value of CLIENT_LoginWithHighLevelSecurity.
	[in]nChannelID	Video channel number is a round number starting from 0.
	[in]hWnd	Window handle valid only under Windows system.
	[in]rType	Preview type.
Return value	Success: Not 0. Failure: 0.	
Note	Windows system: When hWnd is valid, the corresponding window displays picture. When hWnd is NULL, get the video data through setting a callback and send to user for treatment.	

Table 17-9 Live view type and meaning

Preview type.	Meaning
DH_RType_Realplay	Real-time preview.
DH_RType_Multiplay	Multi-picture preview.
DH_RType_Realplay_0	Real-time monitoring—main stream, equivalent to DH_RType_Realplay.
DH_RType_Realplay_1	Real-time monitoring—sub stream 1.
DH_RType_Realplay_2	Real-time monitoring—sub stream 2.
DH_RType_Realplay_3	Real-time monitoring—sub stream 3.
DH_RType_Multiplay_1	Multi-picture preview—1 picture.
DH_RType_Multiplay_4	Multi-picture preview—4 pictures.
DH_RType_Multiplay_8	Multi-picture preview—8 pictures.
DH_RType_Multiplay_9	Multi-picture preview—9 pictures.
DH_RType_Multiplay_16	Multi-picture preview—16 pictures.
DH_RType_Multiplay_6	Multi-picture preview—6 pictures.
DH_RType_Multiplay_12	Multi-picture preview—12 pictures.
DH_RType_Multiplay_25	Multi-picture preview—25 pictures.

17.3.2 CLIENT_StopRealPlayEx

Table 17-10 Stop the real-time monitoring

Item	Description	
Name	Stop the real-time monitoring.	
Function	BOOL CLIENT_StopRealPlayEx(LLONG IRealHandle);	
Parameter	[in] IRealHandle	Return value of CLIENT_RealPlayEx.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.3.3 CLIENT_SaveRealData

Table 17-11 Save the real-time monitoring data as file

Item	Description	
Name	Save the real-time monitoring data as file.	
Function	BOOL CLIENT_SaveRealData(LLONG IRealHandle, const char *pchFileName);	
Parameter	[in] IRealHandle	Return value of CLIENT_RealPlayEx.
	[in] pchFileName	Save path.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.3.4 CLIENT_StopSaveRealData

Table 17-12 Stop saving the real-time monitoring data as file

Item	Description	
Name	Stop saving the real-time monitoring data as file.	
Function	BOOL CLIENT_StopSaveRealData(LLONG IRealHandle);	
Parameter	[in] IRealHandle	Return value of CLIENT_RealPlayEx.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.3.5 CLIENT_SetRealDataCallbackEx

Table 17-13 Set the callback of real-time monitoring data

Item	Description	
Name	Set the callback of real-time monitoring data.	

Item	Description	
Function	BOOL CLIENT_SetRealDataCallBackEx(LLONG IRealHandle, fRealDataCallBackEx cbRealData, LDWORD dwUser, DWORD dwFlag);	
Parameter	[in] IRealHandle	Return value of CLIENT_RealPlayEx.
	[in] cbRealData	Callback of monitoring data flow.
	[in] dwUser	Parameter of callback for monitoring data flow.
	[in] dwFlag	Type of monitoring data in callback. The type is EM_REALDATA_FLAG and supports OR operation.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

Table 17-14 dwFlag type and parameter

dwFlag	Meaning
0x00000001	The original data of device.
0x00000004	YUV data.

17.4 Subscribing to Intelligent Event

17.4.1 CLIENT_RealLoadPictureEx

Table 17-15 Subscribe intelligent event interface

Item	Description	
Name	Subscribe intelligent event interface.	
Function	LLONG CLIENT_RealLoadPictureEx(LLONG ILoginID, int nChannelID, DWORD dwAlarmType, BOOL bNeedPicFile, fAnalyzerDataCallBack cbAnalyzerData, LDWORD dwUser, void* Reserved);	
Parameter	[in] ILoginID	Login handle.
	[in] nChannelID	Channel number.
	[in] dwAlarmType	Alarm type.
	[in] bNeedPicFile	Whether to subscribe picture file, 1-yes, return intelligent picture information in the callback function; 0-no, do not return intelligent picture information(in this case, it can reduce the network flow).

Item	Description	
	[in] cbAnalyzerData	Intelligent data analysis callback function.
	[in] dwUser	The user parameters.
	[in] Reserved	Reserve parameter.
Return value	Success: the subscribe handle of LLONG type. Failure: 0.	
Note	If the interface return failed, call CLIENT_GetLastError to get error code.	

Table 17-16 Preview type and meaning

Preview Type	Meaning
EVENT_IVS_FACEDETECT	Target detection.
EVENT_IVS_FACERECOGNITION	Object recognition.
EVENT_IVS_TRAFFICJUNCTION	Traffic junction event.
EVENT_IVS_TRAFFICJAM	Traffic jam event.
EVENT_IVS_TRAFFIC_OVERSPEED	Over speed event.
EVENT_IVS_TRAFFIC_UNDERSPEED	Low speed.
EVENT_IVS_TRAFFIC_FLOWSTATE	Vehicle flow event.
EVENT_IVS_HUMANTRAIT	Body detection event.
EVENT_IVS_CROSSLINEDETECTION	Tripwire event.
EVENT_IVS_CROSSREGIONDETECTION	Intrusion event.
EVENT_IVS_NUMBERSTAT	People counting event.
EVENT_IVS_MAN_NUM_DETECTION	People counting in event in the area.
EVENT_IVS_ACCESS_CTL	Access control event.

17.4.2 CLIENT_StopLoadPic

Table 17-17 Stop subscribing intelligent event.

Item	Description	
Name	Stop subscribing intelligent event.	
Function	<pre> BOOL CLIENT_StopLoadPic(LLONG IAnalyzerHandle); </pre>	
Parameter	[in] IAnalyzerHandle	Event subscribing handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.5 Searching for and Downloading Intelligent Video and Picture

17.5.1 CLIENT_FindFileEx

Table 17-18 Search files by conditions

Item	Description
Name	Search files by conditions.

Item	Description	
Function	<pre> LLONG CLIENT_FindFileEx(LLONG ILoginID, EM_FILE_QUERY_TYPE emType, void* pQueryCondition, void* reserved, int waittime); </pre>	
Parameter	[in] ILoginID	Login handle.
	[in] emType	Searched file type.
	[in] pQueryCondition	Searching conditions.
	[in] reserved	Reserve bytes.
	[in] waittime	Waiting time.
Return value	Success: the searching handle of LLONG type. Failure: 0.	
Note	None.	

17.5.2 CLIENT_GetTotalFileCount

Table 17-19 Get the number of searched files

Item	Description	
Name	Get the number of searched files.	
Function	<pre> BOOL CLIENT_GetTotalFileCount(LLONG IFindHandle, int* pTotalCount, void * reserved, int waittime); </pre>	
Parameter	[in] IFindHandle	Searching handle.
	[out] pTotalCount	The searched number.
	[in] reserved	Reserve bytes.
	[in] waittime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.5.3 CLIENT_FindNextFileEx

Table 17-20 Search a file

Item	Description
Name	Search a file.

Item	Description	
Function	<pre>int CLIENT_FindNextFileEx(LLONG IFindHandle, int nFilecount, void* pMediaFileInfo, int maxlen, void* reserved, int waittime);</pre>	
Parameter	[in] IFindHandle	Searching handle.
	[in] nFilecount	The searched file number.
	[out] pMediaFileInfo	File cache area.
	[in] maxlen	Search for cache size of file array.
	[in] reserved	Reserve bytes.
	[in] waittime	Timeout.
Return value	Success: File number. Failure: -1. Return 0 means search complete.	
Note	None.	

17.5.4 CLIENT_FindCloseEx

Table 17-21 Stop searching files

Item	Description	
Name	Stop searching files.	
Function	<pre>BOOL CLIENT_FindCloseEx(LLONG IFindHandle);</pre>	
Parameter	[in] IFindHandle	Searching handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.5.5 CLIENT_PlayBackByTimeEx2

Table 17-22 Start playing back the video

Item	Description	
Name	Start playing back the video.	
Function	<pre>LLONG CLIENT_PlayBackByTimeEx2(LLONG ILoginID, Int nChannelID, NET_IN_PLAY_BACK_BY_TIME_INFO* pstNetIn, NET_OUT_PLAY_BACK_BY_TIME_INFO* pstNetOut);</pre>	
Parameter	[in] ILoginID	Login handle.
	[in] nChannelID	Channel number.
	[in] pstNetIn	Playback input parameter.
	[out] pstNetOut	Playback output parameter.
Return value	Success: the playback handle of LLONG type. Failure: 0.	

Item	Description
Note	None.

17.5.6 CLIENT_StopPlayBack

Table 17-23 Stop playing back the video

Item	Description		
Name	Stop playing back the video.		
Function	<pre> BOOL CLIENT_StopPlayBack(LLONG IPlayHandle); </pre>		
Parameter	<table border="1"> <tr> <td>[in] IPlayHandle</td><td>Playback handle.</td></tr> </table>	[in] IPlayHandle	Playback handle.
[in] IPlayHandle	Playback handle.		
Return value	Success: TRUE. Failure: FALSE.		
Note	None.		

17.5.7 CLIENT_DownloadByTimeEx

Table 17-24 Start downloading video

Item	Description
Name	Start downloading video.
Function	<pre> LLONG CLIENT_DownloadByTimeEx(LLONG ILoginID, int nChannelId, int nRecordFileType, LPNET_TIME tmStart, LPNET_TIME tmEnd, char* sSavedFileName, fTimeDownLoadPosCallBack cbTimeDownLoadPos, LDWORD dwUserData, fDataCallBack fDownloadDataCallBack, LDWORD dwDataUser, void* pReserved = NULL) </pre>
Parameter	[in] ILoginID
	Login handle.
	[in] nChannelId
	Channel number.
	[in] nRecordFileType
	Video type.
	[in] tmStart
	Start time of video download
	[in] tmEnd
	End time of video download
	[in] sSavedFileName
	Storage path of specific video data. if the path is empty, the video data will not be saved.
	[int] cbTimeDownLoadPos
	Callback function of video download progress.
	[in] dwUserData
	Callback user data of video download progress.
	[in] fDownloadDataCallBa
	ck
	Callback function of video download data.
	[in] dwDataUser
	Callback user data of video download data.

Item	Description	
	[in] pReserved	Reserved parameter.
Return value	Success: the playback handle of LLONG type. Failure: 0.	
Note	None.	

17.5.8 CLIENT_StopDownload

Table 17-25 Stop downloading the video

Item	Description	
Name	Stop downloading the video.	
Function	<pre> BOOL CLIENT_StopDownload(LLONG IFileHandle); </pre>	
Parameter	[in] IFileHandle	Download handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.5.9 CLIENT_DownloadRemoteFile

Table 17-26 Download files by file name

Item	Description	
Name	Download files by file name.	
Function	<pre> BOOL CLIENT_DownloadRemoteFile(LLONG ILoginID, const DH_IN_DOWNLOAD_REMOTE_FILE* pInParam, DH_OUT_DOWNLOAD_REMOTE_FILE* pOutParam, int nWaitTime); </pre>	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Download file input parameter.
	[out] pOutParam	Download file output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.6 Subscribing to Face Event

For subscribing face event, see "16.4 Subscribing to Intelligent Event."

17.7 Adding/Deleting/Modifying/Searching the Face Library

17.7.1 CLIENT_OperateFaceRecognitionGroup

Table 17-27 Add, delete and modify the face library

Item	Description	
Name	Add, delete and modify the face library.	
Function	BOOL CLIENT_OperateFaceRecognitionGroup(LLONG ILoginID, const NET_IN_OPERATE_FACERECONGNITION_GROUP* pstInParam, NET_OUT_OPERATE_FACERECONGNITION_GROUP *pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.7.2 CLIENT_FindGroupInfo

Table 17-28 Searching the of face library

Item	Description	
Name	Searching the of face library.	
Function	BOOL CLIENT_FindGroupInfo(LLONG ILoginID, const NET_IN_FIND_GROUP_INFO* pstInParam, NET_OUT_FIND_GROUP_INFO *pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8 Adding/Deleting/Modifying/Searching for People Face

17.8.1 CLIENT_OperateFaceRecognitionDB

Table 17-29 Add, delete and modify the people face

Item	Description	
Name	Add, delete and modify the people face.	
Function	BOOL CLIENT_OperateFaceRecognitionDB(LLONG ILoginID, const NET_IN_OPERATE_FACERECONGNITIONDB* pstInParam, NET_OUT_OPERATE_FACERECONGNITIONDB *pstOutParam, Int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.2 CLIENT_OperateFaceRecognitionDB

Table 17-30 Set the searching conditions of people face

Item	Description	
Name	Set the searching conditions of people face.	
Function	BOOL CLIENT_StartFindFaceRecognition(LLONG ILoginID, const NET_IN_STARTFIND_FACERECONGNITION* pstInParam, NET_OUT_STARTFIND_FACERECONGNITION *pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.3 CLIENT_DoFindFaceRecognition

Table 17-31 Search the face information

Item	Description
Name	Search the face information.

Item	Description	
Function	BOOL CLIENT_DoFindFaceRecognitionRecord(const NET_IN_DOFIND_FACERECONGNITIONRECORD* pstInParam, NET_OUT_DOFIND_FACERECONGNITIONRECORD *pstOutParam, int nWaitTime);	
Parameter	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.4 CLIENT_StopFindFaceRecognition

Table 17-32 Stop searching face information

Item	Description	
Name	Stop searching face information.	
Function	BOOL CLIENT_StopFindFaceRecognition(LLONG IFindHandle);	
Parameter	[in] IFindHandle	Searching handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.5 CLIENT_FaceRecognitionPutDisposition

Table 17-33 Arm by library

Item	Description	
Name	Arm by library.	
Function	BOOL CLIENT_FaceRecognitionPutDisposition(LLONG ILoginID, const NET_IN_FACE_RECOGNITION_PUT_DISPOSITION_INFO* pstInParam, NET_OUT_FACE_RECOGNITION_PUT_DISPOSITION_INFO *pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.6 CLIENT_FaceRecognitionDelDisposition

Table 17-34 Disarm by library

Item	Description	
Name	Disarm by library.	
Function	BOOL CLIENT_FaceRecognitionDelDisposition(LLONG ILoginID, const NET_IN_FACE_RECOGNITION_DEL_DISPOSITION_INFO* pstInParam, NET_OUT_FACE_RECOGNITION_DEL_DISPOSITION_INFO *pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.7 CLIENT_SetGroupInfoForChannel

Table 17-35 Arm by channel

Item	Description	
Name	Arm by channel.	
Function	BOOL CLIENT_SetGroupInfoForChannel(LLONG ILoginID, const NET_IN_SET_GROUPINFO_FOR_CHANNEL * pstInParam, NET_OUT_SET_GROUPINFO_FOR_CHANNEL *pstOutParam, int WaitTime);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.8.8 CLIENT_AttachFaceFindState

Table 17-36 Subscribe searching progress of people face

Item	Description
Name	Subscribe searching progress of people face.

Item	Description	
Function	<pre> LLONG CLIENT_AttachFaceFindState(LLONG ILoginID, const NET_IN_FACE_FIND_STATE* pstInParam, NET_OUT_FACE_FIND_STATE *pstOutParam, Int nWaitTime); </pre>	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pstOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: Face progress handle. Failure: 0.	
Note	None.	

17.8.9 CLIENT_DetachFaceFindState

Table 17-37 Cancel subscribing the searching progress of people face

Item	Description	
Name	Cancel Subscribing the searching progress of people face.	
Function	<pre> BOOL CLIENT_DetachFaceFindState(LLONG IAttachHandle); </pre>	
Parameter	[in] IAttachHandle	The handle returned by CLIENT_AttachFaceFindState.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.9 Body Detection

17.9.1 CLIENT_DownloadRemoteFile

Table 17-38 Download the picture

Item	Description	
Name	Download the picture.	
Function	<pre> BOOL CLIENT_DownloadRemoteFile(LLONG ILoginID, const DH_IN_DOWNLOAD_REMOTE_FILE* pInParam, DH_OUT_DOWNLOAD_REMOTE_FILE* pOutParam, int nWaitTime = 1000); </pre>	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input parameter.
	[out] pOutParam	Output parameter.
	[in] nWaitTime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	

Item	Description
Note	None.

17.10 People Flow Statistics

17.10.1 CLIENT_AttachVideoStatSummary

Table 17-39 Subscribe flow statistics event

Item	Description
Name	Subscribe flow statistics event.
Function	<pre>LLONG CLIENT_AttachVideoStatSummary(LLONG ILoginID, const NET_IN_ATTACH_VIDEOSTAT_SUM* pInParam, NET_OUT_ATTACH_VIDEOSTAT_SUM* pOutParam, int nWaitTime);</pre>
Parameter	[in] ILoginID Login handle.
	[in] pInParam Subscribe input parameter of people flow.
	[out] pOutParam Subscribe output parameter of people flow.
	[in] nWaitTime Timeout.
Return value	Flow statistics subscribing handle.
Note	None.

17.10.2 CLIENT_DetachVideoStatSummary

Table 17-40 Cancel subscribing flow statistics event

Item	Description
Name	Cancel subscribing flow statistics event.
Function	<pre>BOOL CLIENT_DetachVideoStatSummary(LLONG IAttachHandle);</pre>
Parameter	[in] IAttachHandle Flow statistics subscribing handle.
Return value	Success: TRUE. Failure: FALSE.
Note	None.

17.10.3 CLIENT_StartFindNumberStat

Table 17-41 Start searching people history data (set searching conditions)

Item	Description
Name	Start searching people history data (set searching conditions).
Function	<pre>LLONG CLIENT_StartFindNumberStat(LLONG ILoginID, NET_IN_FINDNUMBERSTAT* pstInParam, NET_OUT_FINDNUMBERSTAT* pstOutParam);</pre>

Item	Description	
Parameter	[in] ILoginID	Login handle.
	[in] pstInParam	Input searching conditions.
	[out] pstOutParam	Output query result.
Return value	Searching handle.	
Note	None.	

17.10.4 CLIENT_DoFindNumberStat

Table 17-42 Start searching people history data (Set searching conditions)

Item	Description	
Name	Start searching people history data (Set searching conditions).	
Function	<pre>int CLIENT_DoFindNumberStat(LLONG IFindHandle, NET_IN_DOFINDNUMBERSTAT* pstInParam, NET_OUT_DOFINDNUMBERSTAT* pstOutParam);</pre>	
Parameter	[in] ILoginID	Login handle.
	[in] plnParam	Searching input parameter.
	[out] pstOutParam	Searching output parameter.
Return value	Searching number.	
Note	None.	

17.10.5 CLIENT_StopFindNumberStat

Table 17-43 Stop searching history data

Item	Description	
Name	Stop searching history data.	
Function	<pre>BOOL CLIENT_StopFindNumberStat(LLONG IFindHandle);</pre>	
Parameter	[in] IFindHandle	Searching handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.11 Intelligent Traffic

17.11.1 CLIENT_FindRecord

Table 17-44 Start searching data (Set searching conditions)

Item	Description
Name	Start searching data (Set searching conditions).

Item	Description	
Function	BOOL CLIENT_FindRecord(LLONG ILoginID, NET_IN_FIND_RECORD_PARAM* pInParam, NET_OUT_FIND_RECORD_PARAM* pOutParam, int waittime=1000);	
Parameter	[in] ILoginID	Login handle.
	[in] pInParam	Input searching conditions.
	[out] pOutParam	Output query result.
Return value	Searching handle.	
Note	None.	

17.11.2 CLIENT_QueryRecordCount

Table 17-45 Search the total number of data

Item	Description	
Name	Search the total number of data.	
Function	BOOL CLIENT_QueryRecordCount(NET_IN_QUEYT_RECORD_COUNT_PARAM* pInParam, NET_OUT_QUEYT_RECORD_COUNT_PARAM* pOutParam, int waittime=1000);	
Parameter	[in] pInParam	Searching input parameter.
	[out] pOutParam	Searching output parameter.
	[in] waittime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.11.3 CLIENT_FindNextRecord

Table 17-46 Search the data of specified number

Item	Description	
Name	Search the data of specified number.	
Function	int CLIENT_FindNextRecord(NET_IN_FIND_NEXT_RECORD_PARAM* pInParam, NET_OUT_FIND_NEXT_RECORD_PARAM* pOutParam, int waittime=1000);	
Parameter	[in] pstInParam	Searching input parameter.
	[out] pstOutParam	Searching output parameter.
	[in] waittime	Timeout.
Return value	Searching number.	
Note	None.	

17.11.4 CLIENT_FindRecordClose

Table 17-47 Stop searching vehicle flow

Item	Description
Name	Stop searching vehicle flow.
Function	BOOL CLIENT_FindRecordClose(LLONG IFindHandle);
Parameter	[in] IFindHandle Searching handle.
Return value	Success: TRUE. Failure: FALSE.
Note	None.

17.11.5 CLIENT_OperateTrafficList

Table 17-48 Add, delete and modify the blocklist and allowlist

Item	Description
Name	Add, delete and modify the blocklist and allowlist.
Function	BOOL CLIENT_OperateTrafficList(LLONG ILoginID , NET_IN_OPERATE_TRAFFIC_LIST_RECORD* pstInParam , NET_OUT_OPERATE_TRAFFIC_LIST_RECORD *pstOutParam , int waittime)
Parameter	[in] ILoginID Login handle.
	[in] pstInParam Input parameter for blocklist and allowlist operation.
	[out] pstOutParam Output parameter for blocklist and allowlist operation.
	[in] waittime Timeout.
Return value	Success: TRUE. Failure: FALSE.
Note	NET_TRAFFIC_LIST_INSERT// Add record NET_TRAFFIC_LIST_UPDATE// Update record NET_TRAFFIC_LIST_REMOVE// Delete record

17.11.6 CLIENT_DownloadMultiFile

Table 17-49 Download files in batches

Item	Description
Name	Download files in batches.
Function	BOOL CLIENT_DownloadMultiFile(LLONG ILoginID, NET_IN_DOWNLOAD_MULTI_FILE *pstInParam, NET_OUT_DOWNLOAD_MULTI_FILE *pstOutParam, int waittime=1000);
Parameter	[in] ILoginID Login handle.
	[in] pstInParam Searching input parameter.
	[out] pstOutParam Searching output parameter.

Item	Description	
	[in] waittime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.11.7 CLIENT_StopLoadMultiFile

Table 17-50 Stop downloading files in batches

Item	Description	
Name	Stop downloading files in batches.	
Function	<pre> BOOL CLIENT_StopLoadMultiFile(LLONG IDownloadHandle); </pre>	
Parameter	[in] IDownloadHandle	Download handle in batches.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.12 Access Control

17.12.1 CLIENT_FindRecord

Table 17-51 Start searching data (Set searching conditions)

Item	Description	
Name	Start searching data (Set searching conditions).	
Function	<pre> BOOL CLIENT_FindRecord(LLONG ILoginID, NET_IN_FIND_RECORD_PARAM* pInParam, NET_OUT_FIND_RECORD_PARAM* pOutParam, int waittime=1000); </pre>	
Parameter	[in] ILoginID	Login handle.
	[int] pInParam	Input searching conditions.
	[out] pOutParam	Output query result.
Return value	Searching handle.	
Note	None.	

17.12.2 CLIENT_FindNextRecord

Table 17-52 Search the data of specified number

Item	Description
Name	Search the data of specified number.

Item	Description	
Function	<pre>int CLIENT_FindNextRecord(NET_IN_FIND_NEXT_RECORD_PARAM* pInParam, NET_OUT_FIND_NEXT_RECORD_PARAM* pOutParam, int waittime=1000);</pre>	
Parameter	[in] pstInParam	Searching input parameter.
	[out] pstOutParam	Searching output parameter.
	[in] waittime	Timeout.
Return value	Searching number.	
Note	None.	

17.12.3 CLIENT_FindRecordClose

Table 17-53 Stop searching

Item	Description	
Name	Stop searching.	
Function	<pre>BOOL CLIENT_FindRecordClose(LLONG IFindHandle);</pre>	
Parameter	[in] IFindHandle	Searching handle.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.12.4 CLIENT_FindRecordClose

Table 17-54 Operate records of personnel and access control

Item	Description	
Name	You can add, delete, modify, search and clean up the people information. You can also delete and clean up access control record.	
Function	<pre>BOOL CLIENT_ControlDevice(LLONG ILoginID , CtrlType type , void *param , int waittime = 1000);</pre>	
Parameter	[in] ILoginID	Login handle.
	[in] type	Control type.
	[in] param	Control parameter, according to different types.
	[in] waittime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.12.5 CLIENT_FindRecordClose

Table 17-55 Add, delete, modify and clean up the information of face picture

Item	Description	
Name	Add, delete, modify and clean up the information of face picture.	
Function	BOOL CLIENT_FaceInfoOpreate(LLONG ILoginID, EM_FACEINFO_OPREATE_TYPE emType, void* pInParam, void* pOutParam, int nWaitTime = 1000);	
Parameter	[in] ILoginID	Login handle.
	[in] emType	Control type.
	[in] pInParam	Control parameter, select different structures according to different types.
	[out] pOutParam	Return parameter, select different structures according to different types.
	[in] waittime	Timeout.
Return value	Success: TRUE. Failure: FALSE.	
Note	None.	

17.13 Parking Space Detection

17.13.1 Subscribing to designated parking space picture

Table 17-56 Subscribe to designated parking space picture

Item	Description	
Name	Subscribe to designated parking space picture	
Function	LLONG CLIENT_AttachParkingSpaceData(LLONG ILoginID, NET_IN_ATTACH_PARKINGSPACE* pstInParam, NET_OUT_ATTACH_PARKINGSPACE* pstOutParam,);	
Parameter	[in] ILoginID	Login handle
	[in] pInParam	Input parameter
	[out] pOutParam	Output parameter
Return value	Subscription handle of parking space picture	
Note	—	

17.13.2 Unsubscribing from Designated Parking Space Picture

Table 17-57 Unsubscribe from designated parking space picture

Item	Description	
Name	Unsubscribe from designated parking space picture	
Function	BOOL CLIENT_DetachParkingSpaceData(NET_IN_DETACH_PARKINGSPACE* pstInParam, NET_OUT_DETACH_PARKINGSPACE* pstOutParam,);	
Parameter	[in] pstInParam	Unsubscribe from the input parameter of designated parking space picture
	[out] pstOutParam	Unsubscribe from the output parameter of designated parking space picture
Return value	Return TRUE for success, and return FALSE for failure	
Note	—	

17.14 Heat Map

17.14.1 Subscribing to Heat Map Data

Table 17-58 Subscribe to heat map data

Item	Description	
Name	Subscribe to heat map data	
Function	LLONG CLIENT_AttachHeatMapRawStream (LLONG ILoginID, const NET_IN_RAWSTREAM_ATTACH_INFO *pInParam, NET_OUT_RAWSTREAM_ATTACH_INFO *pOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle
	[in] pInParam	Subscribe to input parameter of heat map
	[in] nWaitTime	Timeout duration
	[out] pOutParam	Subscribe to output parameter of heat map
Return value	Heat map subscription handle	
Description	—	

17.14.2 Unsubscribing from Heat Map Data

Table 17-59 Unsubscribe from heat map data

Item	Description
Name	Unsubscribe from heat map data

Item	Description	
Function	BOOL CLIENT_DetachHeatMapRawStream (LLONG IAttachHandle,);	
Parameter	[in] IAttachHandle	Subscription handle
Return value	Return TRUE for success, and return FALSE for failure	
Note	—	

17.14.3 Subscribing to Gray Map Data

Table 17-60 Subscribe to gray map data

Item	Description	
Description	Subscribe to grayscale data of heat map	
Function	LLONG CLIENT_AttachHeatMapGrayInfo(LLONG ILoginID, const NET_IN_GRAY_ATTACH_INFO *pInParam, NET_OUT_GRAY_ATTACH_INFO *pOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle
	[in] pInParam	Subscribe to input parameter of heat map grayscale data
	[in] nWaitTime	Timeout duration
	[out] pOutParam	Subscribe to output parameter of heat map grayscale data
Return value	Subscription handle of heat map grayscale data	
Note	—	

17.14.4 Unsubscribing from Gray Map Data

Table 17-61 Unsubscribe from gray map data

Item	Description	
Name	Unsubscribe from heat map grayscale data	
Function	BOOL CLIENT_DetachHeatMapGrayInfo(LLONG IAttachHandle,);	
Parameter	[in] IAttachHandle	Subscription handle
Return value	Return TRUE for success, and return FALSE for failure	
Note	—	

17.15 Stereo Analysis

17.15.1 Subscribing to Video Statistics

Table 17-62 Subscribe to video statistics

Item	Description	
Name	Subscribe to video statistics	
Function	LLONG CLIENT_AttachVideoStatistics(LLONG ILoginID, const NET_IN_ATTACH_VIDEO_STATISTICS* pstInParam, NET_OUT_ATTACH_VIDEO_STATISTICS* pstOutParam, int nWaitTime);	
Parameter	[in] ILoginID	Login handle
	[in] pstInParam	Subscribe to input parameter
	[in] nWaitTime	Timeout duration
	[out] pstOutParam	Subscribe to output parameter
Return value	Subscription handle of video statistics	
Description	—	

17.15.2 Unsubscribing from Video Statistics

Table 17-63 Unsubscribe from video statistics

Item	Description	
Name	Unsubscribe from video statistics	
Function	BOOL CLIENT_DetachVideoStatistics(LLONG IAttachHandle,);	
Parameter	[in] IAttachHandle	Subscription handle
Return value	Return TRUE for success, and return FALSE for failure	
Note	—	

18 Callback Function Definition

18.1 fDisConnect

Table 18-1 Disconnection callback

Item	Description	
Name	Disconnection callback.	
Function	typedef void (CALLBACK *fDisConnect)(LLONG ILoginID, char* pchDVRIP, LONG nDVRPort, LDWORD dwUser);	
Parameter	[out] ILoginID	Return value of CLIENT_LoginWithHighLevelSecurity.
	[out] pchDVRIP	Device IP.
	[out] nDVRPort	Device port.
	[out] dwUser	User parameter of callback.
Return value	None.	
Note	None.	

18.2 fHaveReConnect

Table 18-2 Reconnection callback

Item	Description	
Name	Reconnection callback.	
Function	typedef void (CALLBACK *fHaveReConnect)(LLONG ILoginID, char* pchDVRIP, LONG nDVRPort, LDWORD dwUser);	
Parameter	[out] ILoginID	Return value of CLIENT_LoginWithHighLevelSecurity.
	[out] pchDVRIP	Device IP.
	[out] nDVRPort	Device port.
	[out] dwUser	User parameter of callback.
Return value	None.	
Note	None.	

18.3 fRealDataCallbackEx

Table 18-3 Callback of real-time monitoring data

Item	Description	
Name	The callback of real-time monitoring data.	
Function	<pre>typedef void (CALLBACK *fRealDataCallbackEx)(LONG IRealHandle, DWORD dwDataType, BYTE* pBuffer, DWORD dwBufSize, LONG param, LDWORD dwUser);</pre>	
Parameter	[out] IRealHandle	Return value of CLIENT_RealPlayEx.
	[out] dwDataType	Data type, 0-original data, 2-YUV data.
	[out] pBuffer	Monitor block data address.
	[out] dwBufSize	Monitor the length of block data, unit: byte.
	[out] param	Callback data parameter structure, the types are different according to different value of dwDataType. <ul style="list-style-type: none"> When dwDataType is 0, param is the empty pointer. When dwDataType is 2, param is the tagCBYUVDataParam structure pointer.
	[out] dwUser	User parameter of callback.
Return value	None.	
Note	None.	

18.4 fAnalyzerDataCallback

Table 18-4 Intelligent event callback

Item	Description	
Name	Intelligent event callback.	
Function	<pre>typedef int (CALLBACK *fAnalyzerDataCallback)(LONG IAnalyzerHandle, DWORD dwAlarmType, void* pAlarmInfo, BYTE* pBuffer, DWORD dwBufSize, LDWORD dwUser, int nSequence, void* reserved);</pre>	
Parameter	[out] IAnalyzerHandle	Return value of CLIENT_RealLoadPictureEx.
	[out] dwAlarmType	Intelligent event type.
	[out] pAlarmInfo	Event information cache.

Item	Description	
	[out] pBuffer	Picture cache.
	[out] dwBufSize	Picture cache size.
	[out] dwUser	User data.
	[out] nSequence	<ul style="list-style-type: none"> • nSequence is 0, it means the first time appears. • nSequence is 2, it means only appears once or the last time appears. • nSequence is 1, it means after this time there is always one or multiple times.
	[out] reserved	Reserve.
Return value	None.	
Note	None.	

18.5 fDownloadPosCallBack

Table 18-5 Callback of playback and download by file

Item	Description	
Name	Playback or download progress calling back function by files.	
Function	<pre>typedef void (CALLBACK *fDownloadPosCallBack)(LLONG IPlayHandle, DWORD dwTotalSize, DWORD dwDownloadSize, LDWORD dwUser);</pre>	
Parameter	[out]IPlayHandle	Playback or download return value of interface.
	[out]dwTotalSize	Total size, unit: KB.
	[out]dwDownloadSize	Downloaded size, unit: KB. <ul style="list-style-type: none"> • -1: This time playback complete. • -2: Write file failed.
	[out]dwUser	User data.
Return value	None.	
Note	None.	

18.6 fDataCallBack

Table 18-6 Callback of playback and download data

Item	Description
Name	Playback or download data callback function.

Item	Description	
Function	<pre>typedef int (CALLBACK *fDataCallBack)(LLONG IRealHandle, DWORD dwDataType, BYTE *pBuffer, DWORD dwBufSize, LDWORD dwUser);</pre>	
Parameter	[out] IPlayHandle	Playback or download return value of interface.
	[out] dwDataType	Is 0 (original data).
	[out] pBuffer	Data cache.
	[out] dwBufSize	cache length, unit: byte.
	[out] dwUser	User data.
Return value	None.	
Note	None.	

18.7 fFaceFindState

Table 18-7 Callback of face searching progress

Item	Description	
Name	Face searching progress callback function.	
Function	<pre>typedef void (CALLBACK *fFaceFindState)(LLONG ILoginID, LLONG IAttachHandle, NET_CB_FACE_FIND_STATE* pstStates, int nStateNum, LDWORD dwUser);</pre>	
Parameter	[out] ILoginID	Return login handle.
	[out] IAttachHandle	Event subscribing handle.
	[out] pstStates	Status information of searching the people face.
	[out] nStateNum	Searching progress of people face.
	[out] dwUser	User data.
Return value	None.	
Note	None.	

18.8 fVideoStatSumCallBack

Table 18-8 Callback of subscribing people flow event

Item	Description
Name	People flow event subscribing callback.

Item	Description	
Function	<pre>typedef void (CALLBACK *fVideoStatSumCallBack) (LLONG IAttachHandle, NET_VIDEOSTAT_SUMMARY* pBuf, DWORD dwBufLen, LDWORD dwUser);</pre>	
Parameter	[out] IAttachHandle	Flow statistics subscribing handle.
	[out] pBuf	Flow statistics return data.
	[out] dwBufLen	Data Length returned.
	[out] dwUser	User data.
Return value	None.	
Note	None.	

18.9 fMultiFileDownloadPosCB

Table 18-9 Download file progress callback function in batches.

Item	Description	
Name	Download file progress callback function in batches.	
Function	<pre>typedef void (CALLBACK *fMultiFileDownloadPosCB)(LLONG IDownloadHandle, DWORD dwID, DWORD dwFileTotalSize, DWORD dwDownloadSize, int nError, LDWORD dwUser, void* pReserved);</pre>	
Parameter	[out] IDownloadHandle	Download file handle in batches.
	[out] dwID	ID is dwFileID set by user.
	[out] dwFileTotalSize	Total size of downloaded file.
	[out] dwDownloadSize	File size that have downloaded, when this value is the max value, it means download completed.
	[out] nError	Download error: 1-cache lack, 2-check error for return data, 3-download the current file failed, 4-create file failed.
	[out] dwUser	User data.
	[out] pReserved	Reserve bytes.
Return value	None.	
Note	None.	

18.10 fNotifySnapData

Table 18-10 Callback function of designated parking space picture

Item	Description	
Name	Callback function of designated parking space picture	
Function	<pre>typedef int (CALLBACK *fNotifySnapData) (LLONG IParkingHandle, NET_CB_PARKINGSPACE_DATA* pDiagnosisInfo, void* pBuf, int nBufLen, LDWORD dwUser);</pre>	
Parameter	[out] IParkingHandle	Subscription handle of parking space picture
	[out] pDiagnosisInfo	Returned data of parking space picture
	[out] pBuf	Data buffer
	[out] nBufLen	Buffer length
	[out] dwUser	User data
Return value	—	
Note	—	

18.11 fRawStreamCallBack

Table 18-11 Callback function of original heat map data

Item	Description	
Name	Callback function of original heat map data	
Function	<pre>typedef void (CALLBACK *fRawStreamCallBack) (LLONG IAttachHandle, NET_RAWSTREAM_NOTIFY_INFO* pBuf, DWORD dwBufLen, LDWORD dwUser);</pre>	
Parameter	[out] IAttachHandle	Subscription handle of original heat map data
	[out] pBuf	Return the original heat map data
	[out] dwBufLen	Length of returned data
	[out] dwUser	User data
Return value	—	
Note	—	

18.12 fHeatMapGrayCallBack

Table 18-12 Callback function of heat map grayscale data

Item	Description
Name	Callback function of heat map grayscale data

Item	Description	
Function	<pre>typedef void(CALLBACK *fHeatMapGrayCallBack) (LLONG IAttachHandle, NET_CB_HEATMAP_GRAY_INFO* pstGrayInfo, LDWORD dwUser);</pre>	
Parameter	[out] IAttachHandle	Subscription handle of heat map grayscale data
	[out] pstGrayInfo	Return the grayscale data
	[out] dwUser	User data
Return value	—	
Note	—	

18.13 fVideoStatisticsInfoCallBack

Table 18-13 Callback function of the video statistics

Item	Description	
Name	Callback function of video statistics	
Function	<pre>typedef void(CALLBACK *fVideoStatisticsInfoCallBack) (LLONG IAttachHandle, NET_EM_VS_TYPE emType, void* pBuf, DWORD nBufLen, LDWORD dwUser);</pre>	
Parameter	[out] IAttachHandle	Subscription handle of video statistics
	[out] emType	Business type
	[out] pBuf	Data buffer
	[out] nBufLen	Buffer length
	[out] dwUser	User data
Return value	—	
Note	—	

Appendix 1 Cybersecurity Recommendations

The necessary measures to ensure the basic cyber security of the platform:

1. Use Strong Passwords

Please refer to the following suggestions to set passwords:

- The length should not be less than 8 characters.
- Include at least two types of characters; character types include upper and lower case letters, numbers and symbols.
- Do not contain the account name or the account name in reverse order.
- Do not use continuous characters, such as 123, abc, etc.
- Do not use overlapped characters, such as 111, aaa, etc.

2. Customize the Answer to the Security Question

The security question setting should ensure the difference of answers, choose different questions and customize different answers (all questions are prohibited from being set to the same answer) to reduce the risk of security question being guessed or cracked.

Recommendation measures to enhance platform cyber security:

1. Enable Account Binding IP/MAC

It is recommended to enable the account binding IP/MAC mechanism, and configure the IP/MAC of the terminal where the commonly used client is located as allowlist to further improve access security.

2. Change Password Regularly

We suggest that you change passwords regularly to reduce the risk of being guessed or cracked.

3. Turn On Account Lock Mechanism

The account lock function is enabled by default at the factory, and it is recommended to keep it on to protect the security of your account. After the attacker has failed multiple password attempts, the corresponding account and source IP will be locked.

4. Reasonable Allocation of Accounts and Permissions

According to business and management needs, reasonably add new users, and reasonably allocate a minimum set of permissions for them.

5. Close Non-essential Services and Restrict the Open Form of Essential Services

If not needed, it is recommended to turn off NetBIOS (port 137, 138, 139), SMB (port 445), remote desktop (port 3389) and other services under Windows, and Telnet (port 23) and SSH (port 22) under Linux. At the same time, close the database port to the outside or only open to a specific IP address, such as MySQL (port 3306), to reduce the risks faced by the platform.

6. Patch the Operating System/Third Party Components

It is recommended to regularly detect security vulnerabilities in the operating system and third-party components, and apply official patches in time.

7. Security Audit

- Check online users: It is recommended to check online users irregularly to identify whether there are illegal users logging in.
- View the platform log: By viewing the log, you can get the IP information of the attempt to log in to the platform and the key operation information of the logged-in user.

8. The Establishment of a Secure Network Environment

In order to better protect the security of the platform and reduce cyber security risks, it is recommended that:

- Follow the principle of minimization, restrict the ports that the platform maps externally by firewalls or routers, and only map ports that are necessary for services.
- Based on actual network requirements, separate networks: if there is no communication requirement between the two subnets, it is recommended to use VLAN, gatekeeper, etc. to divide the network to achieve the effect of network isolation.

Appendix 2 Intelligent Events

Type	Code	Remarks
EVENT_IVS_ALL	0x00000001	subscription all event
EVENT_IVS_CROSSLINEDETECTION	0x00000002	cross line event(Corresponding to DEV_EVENT_CROSSLINE_INFO)
EVENT_IVS_CROSSREGIONDETECTION	0x00000003	cross region event(Corresponding to DEV_EVENT_CROSSREGION_INFO)
EVENT_IVS_PASTEDETECTION	0x00000004	past event(Corresponding to DEV_EVENT_PASTE_INFO)
EVENT_IVS_LEFTDETECTION	0x00000005	left event(Corresponding to DEV_EVENT_LEFT_INFO)
EVENT_IVS_STAYDETECTION	0x00000006	stay event(Corresponding to DEV_EVENT_STAY_INFO)
EVENT_IVS_WANDERDETECTION	0x00000007	wander event(Corresponding to DEV_EVENT_WANDER_INFO)
EVENT_IVS_PRESERVATION	0x00000008	reservation event(Corresponding to DEV_EVENT_PRESERVATION_INFO)
EVENT_IVS_MOVEDETECTION	0x00000009	move event(Corresponding to DEV_EVENT_MOVE_INFO)
EVENT_IVS_TAILDETECTION	0x0000000A	tail event(Corresponding to DEV_EVENT_TAIL_INFO)
EVENT_IVS_RIOTERDETECTION	0x0000000B	rioter event(Corresponding to DEV_EVENT_RIOTERL_INFO)
EVENT_IVS_FIREDETECTION	0x0000000C	fire event(Corresponding to DEV_EVENT_FIRE_INFO)
EVENT_IVS_SMOKEDETECTION	0x0000000D	smoke event(Corresponding to DEV_EVENT_SMOKE_INFO)
EVENT_IVS_FIGHTDETECTION	0x0000000E	fight event(Corresponding to DEV_EVENT_FIGHT_INFO)
EVENT_IVS_FLOWSTAT	0x0000000F	flow stat event(Corresponding to DEV_EVENT_FLOWSTAT_INFO)
EVENT_IVS_NUMBERSTAT	0x00000010	number stat event(Corresponding to DEV_EVENT_NUMBERSTAT_INFO)
EVENT_IVS_CAMERACOVERDDETECTION	0x00000011	camera cover event
EVENT_IVS_CAMERAMOVEDDETECTION	0x00000012	camera move event
EVENT_IVS_VIDEOABNORMALDETECTION	0x00000013	video abnormal event(Corresponding to DEV_EVENT_VIDEOABNORMALDETECTION_INFO)
EVENT_IVS_VIDEOBADDETECTION	0x00000014	video bad event
EVENT_IVS_TRAFFICCONTROL	0x00000015	traffic control event(Corresponding to DEV_EVENT_TRAFFICCONTROL_INFO)
EVENT_IVS_TRAFFICACCIDENT	0x00000016	traffic accident event(Corresponding to DEV_EVENT_TRAFFICACCIDENT_INFO)
EVENT_IVS_TRAFFICJUNCTION	0x00000017	traffic junction event(Corresponding to DEV_EVENT_TRAFFICJUNCTION_INFO)
EVENT_IVS_TRAFFICGATE	0x00000018	traffic gate event(Corresponding to

		DEV_EVENT_TRAFFICGATE_INFO)
EVENT_TRAFFICSNAPSHOT	0x00000019	traffic snapshot(Corresponding to DEV_EVENT_TRAFFICSNAPSHOT_INFO)
EVENT_IVS_FACEDETECT	0x0000001A	target detection(Corresponding to DEV_EVENT_FACEDETECT_INFO)
EVENT_IVS_TRAFFICJAM	0x0000001B	traffic-Jam(Corresponding to DEV_EVENT_TRAFFICJAM_INFO)
EVENT_IVS_TRAFFIC_NONMOTORINMOTORROUTE	0x0000001C	Non-motor vehicles occupy the lanes(Corresponding to DEV_EVENT_TRAFFIC_NONMOTORINMOTORROUTE_INFO)
EVENT_IVS_TRAFFIC_RUNREDLIGHT	0x000000100	traffic-RunRedLight(Corresponding to DEV_EVENT_TRAFFIC_RUNREDLIGHT_INFO)
EVENT_IVS_TRAFFIC_OVERLINE	0x000000101	traffic-Overline(Corresponding to DEV_EVENT_TRAFFIC_OVERLINE_INFO)
EVENT_IVS_TRAFFIC_RETROGRADE	0x000000102	traffic-Retrograde(Corresponding to DEV_EVENT_TRAFFIC_RETROGRADE_INFO)
EVENT_IVS_TRAFFIC_TURNLEFT	0x000000103	traffic-TurnLeft(Corresponding to DEV_EVENT_TRAFFIC_TURNLEFT_INFO)
EVENT_IVS_TRAFFIC_TURNRIGHT	0x000000104	traffic-TurnRight(Corresponding to DEV_EVENT_TRAFFIC_TURNRIGHT_INFO)
EVENT_IVS_TRAFFIC_UTURN	0x000000105	traffic-Uturn(Corresponding to DEV_EVENT_TRAFFIC_UTURN_INFO)
EVENT_IVS_TRAFFIC_OVERSPEED	0x000000106	traffic-Overspeed(Corresponding to DEV_EVENT_TRAFFIC_OVERSPEED_INFO)
EVENT_IVS_TRAFFIC_UNDERSPEED	0x000000107	traffic-Underspeed(Corresponding to DEV_EVENT_TRAFFIC_UNDERSPEED_INFO)
EVENT_IVS_TRAFFIC_PARKING	0x000000108	traffic-Parking(Corresponding to DEV_EVENT_TRAFFIC_PARKING_INFO)
EVENT_IVS_TRAFFIC_WRONGROUTE	0x000000109	traffic-WrongRoute(Corresponding to DEV_EVENT_TRAFFIC_WRONGROUTE_INFO)
EVENT_IVS_TRAFFIC_CROSSLANE	0x00000010A	traffic-CrossLane(Corresponding to DEV_EVENT_TRAFFIC_CROSSLANE_INFO)
EVENT_IVS_TRAFFIC_OVERYELLOWLINE	0x00000010B	traffic-OverYellowLine(Corresponding to DEV_EVENT_TRAFFIC_OVERYELLOWLINE_INFO)
EVENT_IVS_TRAFFIC_DRIVINGONSHOULDER	0x00000010C	traffic-DrivingOnShoulder(Corresponding to DEV_EVENT_TRAFFIC_DRIVINGONSHOULDER_INFO)
EVENT_IVS_TRAFFIC_YELLOWPLATEINLANE	0x00000010E	traffic-YellowPlateInLane(Corresponding to DEV_EVENT_TRAFFIC_YELLOWPLATEINLANE_INFO)
EVENT_IVS_TRAFFIC_PEDESTRAINPRIORITY	0x00000010F	Traffic offense-Pedestrian has higher priority at the crosswalk(Corresponding to DEV_EVENT_TRAFFIC_PEDESTRAINPRIORITY_INFO)
EVENT_IVS_ELECTROSPARKDETECTION	0x000000110	ElectroSpark event(Corresponding to DEV_EVENT_ELECTROSPARK_INFO)
EVENT_IVS_TRAFFIC_NOPASSING	0x000000111	no passing(Corresponding to DEV_EVENT_TRAFFIC_NOPASSING_INFO)

EVENT_IVS_ABNORMALRUNDETECTION	0x00000112	abnormal run(Corresponding to DEV_EVENT_ABNORMALRUNDETECTION_INFO)
EVENT_IVS_RETROGRADEDETECTION	0x00000113	retrograde(Corresponding to DEV_EVENT_RETROGRADEDETECTION_INFO)
EVENT_IVS_INREGIONDETECTION	0x00000114	in region detection(Corresponding to DEV_EVENT_INREGIONDETECTION_INFO)
EVENT_IVS_TAKENAWAYDETECTION	0x00000115	taking away things(Corresponding to DEV_EVENT_TAKENAWAYDETECTION_INFO)
EVENT_IVS_PARKINGDETECTION	0x00000116	parking(Corresponding to DEV_EVENT_PARKINGDETECTION_INFO)
EVENT_IVS_FACERECOGNITION	0x00000117	Target recognition(Corresponding to DEV_EVENT_FACERECOGNITION_INFO)
EVENT_IVS_TRAFFIC_MANUALSNAP	0x00000118	manual snap(Corresponding to DEV_EVENT_TRAFFIC_MANUALSNAP_INFO)
EVENT_IVS_TRAFFIC_FLOWSTATE	0x00000119	traffic flow state(Corresponding to DEV_EVENT_TRAFFIC_FLOW_STATE)
EVENT_IVS_TRAFFIC_STAY	0x0000011A	traffic stay(Corresponding to DEV_EVENT_TRAFFIC_STAY_INFO)
EVENT_IVS_TRAFFIC_VEHICLEINROUTE	0x0000011B	traffic vehicle route(Corresponding to DEV_EVENT_TRAFFIC_VEHICLEINROUTE_INFO)
EVENT_ALARM_MOTIONDETECT	0x0000011C	motion detect(Corresponding to DEV_EVENT_ALARM_INFO)
EVENT_ALARM_LOCALALARM	0x0000011D	local alarm(Corresponding to DEV_EVENT_ALARM_INFO)
EVENT_IVS_PSRISEDETECTION	0x0000011E	rise detect(Corresponding to DEV_EVENT_PSRISEDETECTION_INFO)
EVENT_IVS_CROSSFENCEDECTION	0x0000011F	cross fence(Corresponding to DEV_EVENT_CROSSFENCEDECTION_INFO)
EVENT_IVS_TRAFFIC_TOLLGATE	0x00000120	traffic tollgate(Corresponding to DEV_EVENT_TRAFFICJUNCTION_INFO)
EVENT_IVS_DENSITYDETECTION	0x00000121	density detection of persons(Corresponding to DEV_EVENT_DENSITYDETECTION_INFO)
EVENT_IVS_VIDEODIAGNOSIS	0x00000122	video diagnosis result(Corresponding to NET_VIDEODIAGNOSIS_COMMON_INFO and NET_REAL_DIAGNOSIS_RESULT)
EVENT_IVS_QUEUEDECTION	0x00000123	queue detection(Corresponding to DEV_EVENT_QUEUEDECTION_INFO)
EVENT_IVS_TRAFFIC_VEHICLEINBUSROUTE	0x00000124	take up in bus route(Corresponding to DEV_EVENT_TRAFFIC_VEHICLEINBUSROUTE_INFO)
EVENT_IVS_TRAFFIC_BACKING	0x00000125	illegal astern(Corresponding to DEV_EVENT_IVS_TRAFFIC_BACKING_INFO)
EVENT_IVS_AUDIO_ABNORMALDETECTION	0x00000126	audio abnormality(Corresponding to DEV_EVENT_IVS_AUDIO_ABNORMALDETECTION_INFO)
EVENT_IVS_TRAFFIC_RUNYELLOWLIGHT	0x00000127	run yellow light(Corresponding to DEV_EVENT_TRAFFIC_RUNYELLOWLIGHT_INFO)

EVENT_IVS_CLIMBDETECTION	0x00000128	climb detection(Corresponding to DEV_EVENT_IVS_CLIMB_INFO)
EVENT_IVS_LEAVEDETECTION	0x00000129	leave detection(Corresponding to DEV_EVENT_IVS_LEAVE_INFO)
EVENT_IVS_TRAFFIC_PARKINGON YELLOWBOX	0x0000012A	parking on yellow box(Corresponding to DEV_EVENT_TRAFFIC_PARKINGONYELLOWBOX_INFO)
EVENT_IVS_TRAFFIC_PARKINGSPACEPARKING	0x0000012B	parking space parking(Corresponding to DEV_EVENT_TRAFFIC_PARKINGSPACEPARKING_INFO)
EVENT_IVS_TRAFFIC_PARKINGSPACENOPARKING	0x0000012C	parking space no parking(Corresponding to DEV_EVENT_TRAFFIC_PARKINGSPACENOPARKING_INFO)
EVENT_IVS_TRAFFIC_PEDESTRAIN	0x0000012D	passerby(Corresponding to DEV_EVENT_TRAFFIC_PEDESTRAIN_INFO)
EVENT_IVS_TRAFFIC_THROW	0x0000012E	throw(Corresponding to DEV_EVENT_TRAFFIC_THROW_INFO)
EVENT_IVS_TRAFFIC_IDLE	0x0000012F	idle(Corresponding to DEV_EVENT_TRAFFIC_IDLE_INFO)
EVENT_ALARM_VEHICLEACC	0x00000130	Vehicle ACC power outage alarm events(Corresponding to DEV_EVENT_ALARM_VEHICLEACC_INFO)
EVENT_ALARM_VEHICLE_TURNOVER	0x00000131	Vehicle rollover alarm events(Corresponding to DEV_EVENT_VEHICLE_ALARM_INFO)
EVENT_ALARM_VEHICLE_COLLISION	0x00000132	Vehicle crash alarm events(Corresponding to DEV_EVENT_VEHICLE_ALARM_INFO)
EVENT_ALARM_VEHICLE_LARGE_ANGLE	0x00000133	On-board camera large Angle turn events
EVENT_IVS_TRAFFIC_PARKINGSPACEOVERLINE	0x00000134	Parking line pressing events(Corresponding to DEV_EVENT_TRAFFIC_PARKINGSPACEOVERLINE_INFO)
EVENT_IVS_MULTISCENESWITCH	0x00000135	Many scenes switching events(Corresponding to DEV_EVENT_IVS_MULTI_SCENE_SWICH_INFO)
EVENT_IVS_TRAFFIC_RESTRICTED_PLATE	0x00000136	Limited license plate event(Corresponding to DEV_EVENT_TRAFFIC_RESTRICTED_PLATE)
EVENT_IVS_TRAFFIC_OVERSTOPLINE	0x00000137	Cross stop line event(Corresponding to DEV_EVENT_TRAFFIC_OVERSTOPLINE)
EVENT_IVS_TRAFFIC_WITHOUT_SAFEBELT	0x00000138	Traffic unfasten seat belt event(Corresponding to DEV_EVENT_TRAFFIC_WITHOUT_SAFEBELT)
EVENT_IVS_TRAFFIC_DRIVER_SMOKING	0x00000139	Driver smoking event(Corresponding to DEV_EVENT_TRAFFIC_DRIVER_SMOKING)
EVENT_IVS_TRAFFIC_DRIVER_CALLING	0x0000013A	Driver call event(Corresponding to DEV_EVENT_TRAFFIC_DRIVER_CALLING)
EVENT_IVS_TRAFFIC_PEDESTRAIN_RUNREDLIGHT	0x0000013B	Pedestrian red light(Corresponding to DEV_EVENT_TRAFFIC_PEDESTRAIN_RUNREDLIGHT_INFO)
EVENT_IVS_TRAFFIC_PASSNOTINORDER	0x0000013C	Pass not in order(corresponding

RDER		DEV_EVENT_TRAFFIC_PASSNOTINORDER_INFO)
EVENT_IVS_OBJECT_DETECTION	0x00000141	Object feature detection event(Corresponding to DEV_EVENT_TRAFFIC_OBJECT_DETECTION)
EVENT_ALARM_ANALOGALARM	0x00000150	Analog alarm channel alarm event(correspondingDEV_EVENT_ALARM_ANALOGALARM_INFO)
EVENT_IVS_CROSSLINEDETECTION_EX	0x00000151	Warning lineexpansion event(Corresponding to DEV_EVENT_CROSSLINE_INFO_EX)
EVENT_ALARM_COMMON	0x00000152	Normal Record
EVENT_ALARM_VIDEOBLIND	0x00000153	Video tampering event(Corresponding to DEV_EVENT_ALARM_VIDEOBLIND)
EVENT_ALARM_VIDEOLOSS	0x00000154	Video loss event
EVENT_IVS_GETOUTBEDDETECTIO N	0x00000155	Event of getting out bed detection(Corresponding to DEV_EVENT_GETOUTBED_INFO)
EVENT_IVS_PATROLDETECTION	0x00000156	Event of patrol detection(Corresponding to DEV_EVENT_PATROL_INFO)
EVENT_IVS_ONDUTYDETECTION	0x00000157	Event of on duty detection(Corresponding to DEV_EVENT_ONDUTY_INFO)
EVENT_IVS_NOANSWERCALL	0x00000158	Event of VTO do not answer calling request
EVENT_IVS_STORAGENOTEXIST	0x00000159	Event of storage do not exist
EVENT_IVS_STORAGEELOWSPACE	0x0000015A	Event of storage space low
EVENT_IVS_STORAGEFAILURE	0x0000015B	Event of storage failure
EVENT_IVS_PROFILEALARMTRANS MIT	0x0000015C	Event of profile alarm transmit(corresponding to DEV_EVENT_PROFILE_ALARM_TRANSMIT_INFO)
EVENT_IVS_VIDEOSTATIC	0x0000015D	Event of static video detect(corresponding DEV_EVENT_ALARM_VIDEOSTATIC_INFO)
EVENT_IVS_VIDEOTIMING	0x0000015E	Event of video timing detect(corresponding DEV_EVENT_ALARM_VIDEOTIMING_INFO)
EVENT_IVS_HEATMAP	0x0000015F	Heat map (Corresponding to)
EVENT_IVS_CITIZENIDCARD	0x00000160	ID info reading event (Corresponding to DEV_EVENT_ALARM_CITIZENIDCARD_INFO)
EVENT_IVS_PICINFO	0x00000161	Image info event(Corresponding to DEV_EVENT_ALARM_PIC_INFO)
EVENT_IVS_NETPLAYCHECK	0x00000162	NetPlayCheck event(corresponding DEV_EVENT_ALARM_NETPLAYCHECK_INFO)
EVENT_IVS_TRAFFIC_JAM_FORBID _INTO	0x00000163	Jam Forbid into event(corresponding DEV_EVENT_ALARM_JAMFORBIDINTO_INFO)
EVENT_IVS_SNAPBYTIME	0x00000164	Snap by time event(corresponding DEV_EVENT_SNAPBYTIME)
EVENT_IVS_PTZ_PRESET	0x00000165	PTZ turn to preset event(corresponding to DEV_EVENT_ALARM_PTZ_PRESET_INFO)
EVENT_IVS_RFID_INFO	0x00000166	Event of infrared detect info(corresponding to DEV_EVENT_ALARM_RFID_INFO)
EVENT_IVS_STANDUPDETECTION	0x00000167	Event of standing up detection
EVENT_IVS_QSYTRAFFICCARWEIG HT	0x00000168	Event of QSYTrafficCarWeight (corresponding to DEV_EVENT_QSYTRAFFICCARWEIGHT_INFO)

EVENT_IVS_TRAFFIC_COMPAREPLATE	0x00000169	Event of compare plate(corresponding to DEV_EVENT_TRAFFIC_COMPAREPLATE_INFO)
EVENT_IVS_SHOOTINGSCORERECOGNITION	0x0000016A	Event of shooting score recognition(corresponding to CFG_IVS_SHOOTINGSCORERECOGNITION_INFO)
EVENT_IVS_TRAFFIC_FCC	0x0000016B	Event of refuel gas (corresponding to DEV_EVENT_TRAFFIC_FCC_INFO)
EVENT_IVS_TRAFFIC_TRANSFINITE	0x0000016C	Event of traffic transfinite (corresponding to DEV_EVENT_TRAFFIC_TRANSFINITE_INFO)
EVENT_IVS_SCENE_CHANGE	0x0000016D	Event of scene change (corresponding to DEV_ALARM_SCENECCHANGE_INFO,CFG_VIDEOABNORMALDETECTION_INFO)
EVENT_IVS_LETRACK	0x0000016E	Event of simple track(no event data)
EVENT_IVS_OBJECT_ACTION	0x0000016F	Event of object action detection(no event data)
EVENT_IVS_TRAFFIC_ANALYSE_PRESNAP	0x00000170	Event of presnap analyse(corresponding to DEV_EVENT_TRAFFIC_ANALYSE_PRESNAP_INFO)
EVENT_ALARM_EQSTATE	0x00000171	Event of electrical power quality state of smart switch (no event data)
EVENT_IVS_ALARM_IPC	0x00000172	Event of IPC used by DVR/NVR(corresponding to DEV_EVENT_ALARM_IPC_INFO)
EVENT_IVS_POS_RECORD	0x00000173	Event of POS record
EVENT_IVS_NEAR_DISTANCE_DETECTION	0x00000174	Event of near distance detection(corresponding to DEV_EVENT_NEAR_DISTANCE_DETECTION_INFO)
EVENT_IVS_OBJECTSTRUCTLIZE_PERSON	0x00000175	Event of person feature detect(corresponding to DEV_EVENT_OBJECTSTRUCTLIZE_PERSON_INFO)
EVENT_IVS_OBJECTSTRUCTLIZE_NONMOTOR	0x00000176	Event of nonmotor feature detect(corresponding to DEV_EVENT_OBJECTSTRUCTLIZE_NONMOTOR_INFO)
EVENT_IVS_TUMBLE_DETECTION	0x00000177	Event of tumble detection(corresponding to DEV_EVENT_TUMBLE_DETECTION_INFO)
EVENT_IVS_TRAFFIC_ALL	0x000001FF	All event start with [TRAFFIC]
EVENT_IVS_VIDEOANALYSE	0x00000200	All IVS events (Corresponding to)
EVENT_IVS_LINKSD	0x00000201	LinkSD events(Corresponding to)
EVENT_IVS_VEHICLEANALYSE	0x00000202	Vehicle Analyse (Corresponding to DEV_EVENT_VEHICLEANALYSE)
EVENT_IVS_FLOWRATE	0x00000203	Flow rate events(Corresponding to DEV_EVENT_FLOWRATE_INFO)
EVENT_IVS_ACCESS_CTL	0x00000204	Access control events (Corresponding to DEV_EVENT_ACCESS_CTL_INFO)
EVENT_IVS_SNAPMANUAL	0x00000205	SnapManual events(Corresponding to DEV_EVENT_SNAPMANUAL)
EVENT_IVS_TRAFFIC_ELETAGINFO	0x00000206	RFID electronic tag events(Corresponding to DEV_EVENT_TRAFFIC_ELETAGINFO_INFO)
EVENT_IVS_TRAFFIC_TIREDPHYSIOLOGICAL	0x00000207	physiological fatigue driving events(Corresponding to DEV_EVENT_TIREDPHYSIOLOGICAL_INFO)
EVENT_IVS_TRAFFIC_BUSSHARPTURN	0x00000208	bus sharp turn events(Corresponding to DEV_EVENT_BUSSHARPTURN_INFO)
EVENT_IVS_CITIZEN_PICTURE_CO	0x00000209	Event of comparison with ID and card

MPARE		picture(corresponding to DEV_EVENT_CITIZEN_PICTURE_COMPARE_INFO)
EVENT_IVS_TRAFFIC_TIREDLLOWER HEAD	0x0000020A	Event of driver lower head(Corresponding to DEV_EVENT_TIREDLLOWERHEAD_INFO)
EVENT_IVS_TRAFFIC_DRIVERLOOK AROUND	0x0000020B	Event of driver look around(Corresponding to DEV_EVENT_DRIVERLOOKAROUND_INFO)
EVENT_IVS_TRAFFIC_DRIVERLEAVE POST	0x0000020C	Event of driver leave post(Corresponding to DEV_EVENT_DRIVERLEAVEPOST_INFO)
EVENT_IVS_MAN_STAND_DETECTI ON	0x0000020D	stereo standing event(Corresponding to DEV_EVENT_MANSTAND_DETECTION_INFO)
EVENT_IVS_MAN_NUM_DETECTIO N	0x0000020E	Event of regional population statistics (Corresponding to DEV_EVENT_MANNUM_DETECTION_INFO)
EVENT_IVS_STEREO_NUMBERSTAT	0x0000020F	Event of passenger flow statistics(no event data)
EVENT_IVS_TRAFFIC_DRIVERYAW N	0x00000210	Event of driver yawn(Corresponding to DEV_EVENT_DRIVERYAWN_INFO)
EVENT_IVS_NUMBERSTAT_PLAN	0x00000211	Passenger flow statistics plan(no event data,Speed Dome use,Corresponding to rule config CFG_NUMBERSTAT_INFO)
EVENT_IVS_HEATMAP_PLAN	0x00000212	Heat map plan(no event data,Speed Dome use,Corresponding to rule config CFG_IVS_HEATMAP_INFO)
EVENT_IVS_CALLNOANSWERED	0x00000213	Event of call no answered
EVENT_IVS_IGNOREINVITE	0x00000214	Event of ignore invite
EVENT_IVS_HUMANTRAIT	0x00000215	Event of human trait(Corresponding to DEV_EVENT_HUMANTRAIT_INFO)
EVENT_ALARM_LE_HEADETECTI ON	0x00000216	alarm of Head detection(Corresponding to DEV_EVENT_LE_HEADETECTION_INFO)
EVENT_IVS_FACEANALYSIS	0x00000217	Event of target analysis(no event data)
EVENT_IVS_TRAFFIC_TURNLEFTAF TERSTRAIGHT	0x00000218	Event of turn left not give precedence to straight(Corresponding to DEV_EVENT_TURNLEFTAFTERSTRAIGHT_INFO)
EVENT_IVS_TRAFFIC_BIGBENDSM ALLTURN	0x00000219	Event of small turn on big bend(Corresponding to DEV_EVENT_BIGBENDSMALLTURN_INFO)
EVENT_IVS_ROAD_CONSTRUCTIO N	0x0000021A	Event of road construction inspection (Corresponding to DEV_EVENT_ROAD_CONSTRUCTION_INFO)
EVENT_IVS_ROAD_BLOCK	0x0000021B	Event of road block detection (Corresponding to DEV_EVENT_ROAD_BLOCK_INFO)
EVENT_IVS_TRAFFIC_QUEUEJUMP	0x0000021C	Event of car jump a queue(Corresponding to DEV_EVENT_TRAFFIC_QUEUEJUMP_INFO)
EVENT_IVS_VEHICLE_SUSPICIOUSC AR	0x0000021D	Event of Suspicious Car(Corresponding to DEV_EVENT_VEHICLE_SUSPICIOUSCAR_INFO)
EVENT_IVS_TRAFFIC_TURNRIGHTA FTERSTRAIGHT	0x0000021E	Turn right to make a straight event(Corresponding to DEV_EVENT_TURNRIGHTAFTERSTRAIGHT_INFO)
EVENT_IVS_TRAFFIC_TURNRIGHTA FTERPEOPLE	0x0000021F	Turn right and go straight to pedestrians(Corresponding to DEV_EVENT_TURNRIGHTAFTERPEOPLE_INFO)

EVENT_IVS_INSTALL_CARDREADER	0x00000220	Install card reader event(Corresponding to DEV_EVENT_INSTALL_CARDREADER_INFO)
EVENT_ALARM_YALE_DROPBOX_BADTOKEN	0x00000221	event of Yale token bad
EVENT_IVS_ACC_OFF_SNAP	0x00000222	Vehicle equipment's ACC off snap event(Corresponding to DEV_EVENT_ACC_OFF_SNAP_INFO)
EVENT_IVS_XRAY_DETECTION	0x00000223	X ray detection(Corresponding to DEV_EVENT_XRAY_DETECTION_INFO)
EVENT_IVS_NOTCLEARCAR	0x00000224	Not clear car alarm(Corresponding to DEV_EVENT_NOTCLEARCAR_INFO)
EVENT_IVS_SOSALEART	0x00000225	SOS alert(Corresponding to DEV_EVENT_SOSALEART_INFO)
EVENT_IVS_OVERLOAD	0x00000226	Overload snap picture(Corresponding to DEV_EVENT_OVERLOAD_INFO)
EVENT_IVS_NONWORKINGTIME	0x00000227	Non-working time alarm(Corresponding to DEV_EVENT_NONWORKINGTIME_INFO)
EVENT_IVS_TRAFFIC_HIGH_BEAM	0x00000228	Event of traffic high beam(Corresponding to DEV_EVENT_TRAFFIC_HIGH_BEAM_INFO)
EVENT_IVS_TRAFFIC_TRUCKFORBID	0x00000229	truck forbid Event(Corresponding to DEV_EVENT_TRAFFICTRUCKFORBID_INFO)
EVENT_IVS_DRIVINGWITHOUTCARD	0x0000022A	Event of Driving without card(Corresponding to DEV_EVENT_DRIVINGWITHOUTCARD_INFO)
EVENT_IVS_HIGHSPEED	0x0000022B	Event of high speed(Corresponding to DEV_EVENT_HIGHSPEED_INFO)
EVENT_IVS_CROWDDETECTION	0x0000022C	Event of crowd detection(Corresponding to DEV_EVENT_CROWD_DETECTION_INFO)
EVENT_IVS_TRAFFIC_CARDISTANCESHORT	0x0000022D	Event of car distance short(Corresponding to DEV_EVENT_TRAFFIC_CARDISTANCESHORT_INFO)
EVENT_IVS_PEDESTRIAN_JUNCTION	0x00000230	Pedestrian Junction Event(Corresponding to DEV_EVENT_PEDESTRIAN_JUNCTION_INFO)
EVENT_IVS_VEHICLE_RECOGNITION	0x00000231	Vehicle recognition alarm(Corresponding to DEV_EVENT_VEHICLE_RECOGNITION_INFO)
EVENT_IVS_PASS_CHANGE	0x00000232	Event of Pass change (Corresponding to DEV_EVENT_PASS_CHANGE_INFO)
EVENT_IVS_TRAFFIC_PARKING_SPACEDETECTION	0x00000233	Event of Space detection
EVENT_IVS_TRAFFIC_WAITINGAREA	0x00000234	Event of Traffic Waiting Area(Corresponding to DEV_EVENT_TRAFFIC_WAITINGAREA_INFO)
EVENT_IVS_TRAFFIC_BAN	0x00000235	Event of Traffic(Corresponding to DEV_EVENT_TRAFFIC_BAN_INFO)
EVENT_IVS_POS_EXCHANGE	0x00000236	Event of POS Exchange (Corresponding to DEV_EVENT_POS_EXCHANGE_INFO)
EVENT_IVS_STEREO_FIGHTDETECTION	0x00000237	Stereoscopic behavior analysis of fighting/strenuous motion(only used for configuration rules, alarm event is EVENT_IVS_FIGHTDETECTION)

EVENT_IVS_STEREO_DISTANCE_DETECTION	0x00000238	Stereoscopic behavior analysis of distance anomaly/personnel close(only used for configuration rules, alarm event is EVENT_IVS_DISTANCE_DETECTION)
EVENT_IVS_STEREO_STEREOFALLDETECTION	0x00000239	Stereoscopic behavior analysis of fall detection(only used for configuration rules, alarm event is EVENT_IVS_TUMBLE_DETECTION)
EVENT_IVS_STEREO_STAYDETECTION	0x0000023A	Stereoscopic behavior analysis of stay detection(only used for configuration rules, alarm event is EVENT_IVS_STAYDETECTION)
EVENT_IVS_BANNER_DETECTION	0x0000023B	Event of Banner detection(Corresponding to DEV_EVENT_BANNER_DETECTION_INFO)
EVENT_IVS_NORMAL_FIGHTDETECTION	0x0000023C	Event of normal fight(only be used to rule of normal fight, the alarm event is same as EVENT_IVS_FIGHTDETECTION)
EVENT_IVS_ELEVATOR_ABNORMAL	0x0000023D	Event of elevator abnormal(Corresponding to DEV_EVENT_ELEVATOR_ABNORMAL_INFO)
EVENT_IVS_NONMOTORDETECT	0x0000023E	Event of Non-Motor detect (Corresponding to DEV_EVENT_NONMOTORDETECT_INFO)
EVENT_IVS_VEHICLEDETECT	0x0000023F	Event of Vehicle detect (only be used to rule of vehicledetect,the alarm event is same as EVENT_IVS_TRAFFICJUNCTION)
EVENT_IVS_TRAFFIC_PARKING_B	0x00000240	Event of Class B traffic-Parking(Corresponding to DEV_EVENT_TRAFFIC_PARKING_B_INFO)
EVENT_IVS_TRAFFIC_PARKING_C	0x00000241	Event of Class C traffic-Parking(Corresponding to DEV_EVENT_TRAFFIC_PARKING_C_INFO)
EVENT_IVS_TRAFFIC_PARKING_D	0x00000242	Event of Class D traffic-Parking(Corresponding to DEV_EVENT_TRAFFIC_PARKING_D_INFO)
EVENT_IVSS_FACEATTRIBUTE	0x00000243	Event of IVSS target attribute(no event data)
EVENT_IVSS_FACECOMPARE	0x00000244	Event of IVSS target compare(no event data)
EVENT_IVS_FIREWARNING	0x00000245	Event of FireWarning(Corresponding to DEV_EVENT_FIREWARNING_INFO)
EVENT_IVS_SHOPPRESENCE	0x00000246	Event of ShopPresence(Corresponding to DEV_EVENT_SHOPPRESENCE_INFO)
EVENT_IVS_WASTEDUMPED	0x00000247	Event of WasteDumped(Corresponding to DEV_EVENT_WASTEDUMPED_INFO)
EVENT_IVS_SPILLED MATERIAL_DETECTION	0x00000248	Event of spilled material detection(Corresponding to DEV_EVENT_SPILLED MATERIAL_DETECTION_INFO)
EVENT_IVS_STEREO_MANNUM_DETECTION	0x00000249	Stereoscopic behavior analysis of mannum detection(only used for configuration rules, alarm event is EVENT_IVS_MAN_NUM_DETECTION)
EVENT_IVS_DISTANCE_DETECTION	0x0000024A	Event of distance detection(Corresponding to DEV_EVENT_DISTANCE_DETECTION_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR_OVERLOAD	0x0000024B	Event of non-motor overload (Corresponding to DEV_EVENT_TRAFFIC_NONMOTOR_OVERLOAD_INFO)

EVENT_IVS_TRAFFIC_NONMOTOR_WITHOUTSAFEHAT	0x0000024C	Event of non-motor without safehat (Corresponding to DEV_EVENT_TRAFFIC_NONMOTOR_WITHOUTSAFEHAT_INFO)
EVENT_IVS_TRAFFIC_JAM_STOP_ON_ZEBRACROSSING	0x0000024D	Event of TrafficJamStopOnZebraCrossing(Corresponding to DEV_EVENT_TRAFFIC_JAM_STOP_ON_ZEBRACROSSING_INFO)
EVENT_IVS_FLOWBUSINESS	0x0000024E	Event of flowBusiness (Corresponding to DEV_EVENT_FLOWBUSINESS_INFO)
EVENT_IVS_CITY_MOTORPARKING	0x0000024F	Event of CityMotorParking (Corresponding to DEV_EVENT_CITY_MOTORPARKING_INFO)
EVENT_IVS_CITY_NONMOTORPARKING	0x00000250	Event of CityNonMotorParking (Corresponding to EV_EVENT_CITY_NONMOTORPARKING_INFO)
EVENT_IVS_LANEDEPARTURE_WARNING	0x00000251	Lane Departure warning(Corresponding to DEV_EVENT_LANEDEPARTURE_WARNING_INFO)
EVENT_IVS_FORWARDCOLLISION_WARNING	0x00000252	Forward Collision Warning(Corresponding to DEV_EVENT_FORWARDCOLLISION_WARNING_INFO)
EVENT_IVS_MATERIALSSTAY	0x00000253	Event of MaterialsStay(Corresponding to DEV_EVENT_MATERIALSSTAY_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR_HOLDUMBRELLA	0x00000254	Event of NonMotor hold umbrella (Corresponding to DEV_EVENT_TRAFFIC_NONMOTOR_HOLDUMBRELLA_INFO)
EVENT_IVS_JABLOTRON_ALARM	0x00000255	Event of Jablotron alarm
EVENT_IVS_VIDEOUNFOCUS_ALARM	0x00000256	Event of video unfocus(Corresponding to DEV_EVENT_EVENT_VIDEOUNFOCUS_INFO)
EVENT_IVS_FLOATINGOBJECT_DETECTION	0x00000257	Event of FloatingObject detection (Corresponding to DEV_EVENT_FLOATINGOBJECT_DETECTION_INFO)
EVENT_IVS_SHIP_DETECTION	0x00000258	Event of Ship detection (Corresponding to DEV_EVENT_SHIP_DETECTION_INFO)
EVENT_IVS_AIRPLANE_DETECTION	0x00000259	Event of AirPlaneDetection(Corresponding to DEV_EVENT_AIRPLANE_DETECTION_INFO)
EVENT_IVS_PHONECALL_DETECT	0x0000025A	Event of phone call detect(Corresponding to DEV_EVENT_PHONECALL_DETECT_INFO)
EVENT_IVS_SMOKING_DETECT	0x0000025B	Event of Smoking Detection(Corresponding to DEV_EVENT_SMOKING_DETECT_INFO)
EVENT_IVS_RADAR_SPEED_LIMIT_ALARM	0x0000025C	Event of Radar speed limit alarm(Corresponding to DEV_EVENT_RADAR_SPEED_LIMIT_ALARM_INFO)
EVENT_IVS_WATER_LEVEL_DETECTION	0x0000025D	Event of Water level detection (Corresponding to DEV_EVENT_WATER_LEVEL_DETECTION_INFO)
EVENT_IVS_HOLD_UMBRELLA	0x0000025E	Event of Hold umbrella detection(Corresponding to DEV_EVENT_HOLD_UMBRELLA_INFO)
EVENT_IVS_GARBAGE_EXPOSURE	0x0000025F	Event of Garbage Exposure detection (Corresponding to DEV_EVENT_GARBAGE_EXPOSURE_INFO)
EVENT_IVS_DUSTBIN_OVER_FLOW	0x00000260	Event of Dustbin Overflow detection (Corresponding

		to DEV_EVENT_DUSTBIN_OVER_FLOW_INFO)
EVENT_IVS_DOOR_FRONT_DIRTY	0x00000261	Event of Door Front Dirty detection(Corresponding to DEV_EVENT_DOOR_FRONT_DIRTY_INFO)
EVENT_IVS_QUEUESTAY_DETECTION	0x00000262	Event of Queue Stay Detection (Corresponding to DEV_EVENT_QUEUESTAY_DETECTION_INFO)
EVENT_IVS_QUEUENUM_DETECTION	0x00000263	Event of Queue Num Detection(Corresponding to DEV_EVENT_QUEUENUM_DETECTION_INFO)
EVENT_IVS_GENERATEGRAPH_DETECTION	0x00000264	Event of Generate Graph Detection(Corresponding to DEV_EVENT_GENERATEGRAPH_DETECTION_INFO)
EVENT_IVS_TRAFFIC_PARKING_MANUAL	0x00000265	Event of manual traffic-Parking(Corresponding to DEV_EVENT_TRAFFIC_PARKING_MANUAL_INFO)
EVENT_IVS_HELMET_DETECTION	0x00000266	Event of helmet detection (Corresponding to DEV_EVENT_HELMET_DETECTION_INFO)
EVENT_IVS_DEPOSIT_DETECTION	0x00000267	Event of deposit detection (Corresponding to DEV_EVENT_DEPOSIT_DETECTION_INFO)
EVENT_IVS_HOTSPOT_WARNING	0x00000268	Event of Hot spot warning(Corresponding to DEV_EVENT_HOTSPOT_WARNING_INFO)
EVENT_IVS_WEIGHING_PLATFORM_DETECTION	0x00000269	Event of weighing platform detection(Corresponding to DEV_EVENT_WEIGHING_PLATFORM_DETECTION_INFO)
EVENT_IVS_CLASSROOM_BEHAVIOR	0x0000026A	Event of classroom behavior detection(Corresponding to DEV_EVENT_CLASSROOM_BEHAVIOR_INFO)
EVENT_IVS_VEHICLE_DISTANCE_NEAR	0x0000026B	Event of safe driving vehicle distance near alarm(Corresponding to DEV_EVENT_VEHICLE_DISTANCE_NEAR_INFO)
EVENT_IVS_TRAFFIC_DRIVER_ABNORMAL	0x0000026C	Event of traffic driver abnormal alarm(Corresponding to DEV_EVENT_TRAFFIC_DRIVER_ABNORMAL_INFO)
EVENT_IVS_TRAFFIC_DRIVER_CHANGE	0x0000026D	Event of traffic driver change alarm(Corresponding to DEV_EVENT_TRAFFIC_DRIVER_CHANGE_INFO)
EVENT_IVS_WORKCLOTHES_DETECTION	0x0000026E	Event of work clothes(helmet/clothes)detection(Corresponding to DEV_EVENT_WORKCLOTHES_DETECT_INFO)
EVENT_IVS_SECURITYGATE_PERSONALARM	0x0000026F	Event of security gate person alarm(Corresponding to DEV_EVENT_SECURITYGATE_PERSONALARM_INFO)
EVENT_IVS_STAY_ALONE_DETECTION	0x00000270	Event of stay alone detection (Corresponding to DEV_EVENT_STAY_ALONE_DETECTION_INFO)
EVENT_IVS_TRAFFIC_ROAD_BLOCK	0x00000271	Event of traffic road block(Corresponding to DEV_EVENT_TRAFFIC_ROAD_BLOCK_INFO)
EVENT_IVS_TRAFFIC_ROAD_CONSTRUCTION	0x00000272	Event of traffic road construction(Corresponding to DEV_EVENT_TRAFFIC_ROAD_CONSTRUCTION_INFO)
EVENT_IVS_XRAY_DETECT_BYOBJECT	0x00000273	Rule config of X ray detection, event is EVENT_IVS_XRAY_DETECTION
EVENT_IVS_WORKSTATDETECTION	0x00000274	Event of work stat detection(Corresponding to DEV_EVENT_WORKSTATDETECTION_INFO)

EVENT_IVS_INFRAREDBLOCK	0x00000275	Event of infrared block(Corresponding to DEV_EVENT_INFRAREDBLOCK_INFO)
EVENT_IVS_FEATURE_ABSTRACT	0x00000276	Event of feature abstract(Corresponding to DEV_EVENT_FEATURE_ABSTRACT_INFO)
EVENT_IVS_INTELLI_SHELF	0x00000277	Event of intelligent replenishment(Corresponding to DEV_EVENT_INTELLI_SHELF_INFO)
EVENT_IVS_PANORAMA_SHOT	0x00000278	Event of Panoramic snapshot(Corresponding to DEV_EVENT_PANORAMA_SHOT_INFO)
EVENT_ALARM_SMARTMOTION_HUMAN	0x00000279	Event of smart motion detection(human),(Corresponding to DEV_EVENT_SMARTMOTION_HUMAN_INFO)
EVENT_ALARM_SMARTMOTION_VEHICLE	0x0000027A	Event of smart motion detection(vehicle),(Corresponding to DEV_EVENT_SMARTMOTION_VEHICLE_INFO)
EVENT_IVS_CAR_DRIVING_IN_OUT	0x0000027B	Event of car driving in or out(Corresponding to DEV_EVENT_CAR_DRIVING_IN_OUT_INFO)
EVENT_IVS_PARKINGSPACE_STATUS	0x0000027C	Event of parking spcae status(Corresponding to DEV_EVENT_PARKINGSPACE_STATUS_INFO)
EVENT_IVS_VIOLENT_THROW_DETECTION	0x0000027D	Event violent throw detection (Corresponding to DEV_EVENT_VIOLENT_THROW_DETECTION_INFO)
EVENT_IVS_TRAMCARSECTIONS_DETECTION	0x0000027E	Event of TramCarSectionsDetection (Corresponding to DEV_EVENT_TRAMCARSECTIONS_DETECTION_INFO)
EVENT_IVS_ALARM_BOX_ALARM	0x0000027F	Event of alarm box, only for mobile push
EVENT_IVS_FACE_COMPARISON	0x00000280	Event of target comparision, only for mobile push
EVENT_IVS_FACEBODY_DETECT	0x00000281	Event of target body detect (Corresponding to DEV_EVENT_FACEBODY_DETECT_INFO)
EVENT_IVS_FACEBODY_ANALYSE	0x00000282	Event of target body analyse (Corresponding to DEV_EVENT_FACEBODY_ANALYSE_INFO)
EVENT_IVS_GASSTATION_VEHICLE_DETECT	0x00000283	Event of gas station vehicle detection(Corresponding to DEV_EVENT_GASSTATION_VEHICLE_DETECT_INFO)
EVENT_IVS_CONGESTION_DETECTION	0x00000284	Event of congestion detection(Corresponding to DEV_EVENT_CONGESTION_DETECTION_INFO)
EVENT_IVS_VEHICLELIMIT_DETECTION	0x00000285	Event of vehicle limit detection(Corresponding to DEV_EVENT_VEHICLELIMIT_DETECTION_INFO)
EVENT_IVS_ANIMAL_DETECTION	0x00000286	Event of animal detection(Corresponding to DEV_EVENT_ANIMAL_DETECTION_INFO)
EVENT_IVS_SHOP_WINDOW_POST	0x00000287	Event of shop window post(Corresponding to DEV_EVENT_SHOP_WINDOW_POST_INFO)
EVENT_IVS_SHOP_SIGN_ABNORMAL	0x00000288	Event of shop sign abnormal (Corresponding to DEV_EVENT_SHOP_SIGN_ABNORMAL_INFO)
EVENT_IVS_BREED_DETECTION	0x00000289	Event of breed detection (Corresponding to DEV_EVENT_BREED_DETECTION_INFO)
EVENT_IVS_AIRPORT_VEHICLE_DETECT	0x0000028A	Event of airport-vehicle detection (Corresponding to DEV_EVENT_AIRPORT_VEHICLE_DETECT)

EVENT_IVS_PIG_TEMPERATURE_DETECT	0x0000028B	Event of pig body temperature detection (it is only used to analyse rule config)
EVENT_IVS_MAN_CAR_COEXISTANCE	0x0000028C	Event of man and cars exist at the same time (Corresponding to DEV_EVENT_MAN_CAR_COEXISTANCE_INFO)
EVENT_IVS_HIGH_TOSS_DETECT	0x0000028D	Event of high toss detection(Corresponding to DEV_EVENT_HIGH_TOSS_DETECT_INFO)
EVENT_IVS_ELECTRIC_GLOVE_DETECT	0x0000028E	Event of electric glove detection(Corresponding to DEV_EVENT_ELECTRIC_GLOVE_DETECT_INFO)
EVENT_IVS_ELECTRIC_LADDER_DETECT	0x0000028F	Event of electric ladder detection(Corresponding to DEV_EVENT_ELECTRIC_LADDER_DETECT_INFO)
EVENT_IVS_ELECTRIC_CURTAIN_DETECT	0x00000290	Event of electric curtain detection(Corresponding to DEV_EVENT_ELECTRIC_CURTAIN_DETECT_INFO)
EVENT_IVS_ELECTRIC_FENCE_DETECT	0x00000291	Event of electric fence detection(Corresponding to DEV_EVENT_ELECTRIC_FENCE_DETECT_INFO)
EVENT_IVS_ELECTRIC_SIGNBOARD_DETECT	0x00000292	Event of electric signboard detection(Corresponding to DEV_EVENT_ELECTRIC_SIGNBOARD_DETECT_INFO)
EVENT_IVS_ELECTRIC_BELT_DETECT	0x00000293	Event of electric belt detection(Corresponding to DEV_EVENT_ELECTRIC_BELT_DETECT_INFO)
EVENT_IVS_RADAR_LINE_DETECTION	0x00000294	Event of Radar cross line detection(Corresponding to DEV_EVENT_RADAR_LINE_DETECTION_INFO)
EVENT_IVS_RADAR_REGION_DETECTION	0x00000295	Event of Radar cross region detection(Corresponding to DEV_EVENT_RADAR_REGION_DETECTION_INFO)
EVENT_IVS_AUDIO_INTENSITY	0x00000296	Event of audio intensity (Corresponding to DEV_EVENT_AUDIO_INTENSITY_INFO)
EVENT_IVS_PARKING_LOT_STATUS_DETECTION	0x00000297	Event of parking lot status detection (Corresponding to DEV_EVENT_PARKING_LOT_STATUS_DETECTION_INFO)
EVENT_IVS_VEHICLE_COMPARE	0x00000298	
EVENT_IVS_DREGS_UNCOVERED	0x00000299	Event of loading test not covered by muck truck(Corresponding to DEV_EVENT_DREGS_UNCOVERED_INFO)
EVENT_IVS_WALK_DETECTION	0x0000029A	Event of walk detection (Corresponding to DEV_EVENT_WALK_DETECTION_INFO)
EVENT_IVS_BACK_TO_DETECTION	0x0000029B	Event of back to dection (corresponding to DEV_EVENT_BACK_TO_DETECTION_INFO)
EVENT_IVS_WRITE_ON_THE_BOARD_DETECTION	0x0000029C	Event of write on the board detection(corresponding to DEV_EVENT_WRITE_ON_THE_BOARD_DETECTION_INFO)
EVENT_IVS_SMART_KITCHEN_CLOTHES_DETECTION	0x0000029D	Event of Smart kitchen wearing detection(alarm for not wearing mask, chef's clothes whose color does not meet the requirements)(Corresponding to DEV_EVENT_SMART_KITCHEN_CLOTHES_DETECTION)

		_INFO)
EVENT_IVS_SLEEP_DETECT	0x0000029E	Event of sleep detect(Corresponding to DEV_EVENT_SLEEP_DETECT_INFO)
EVENT_IVS_WALK_AROUND_DETECT	0x0000029F	Event of walk around detect(Corresponding to DEV_EVENT_WALK_AROUND_DETECT_INFO)
EVENT_IVS_PLAY_MOBILEPHONE	0x00000300	Event of play mobile phone(Corresponding to DEV_EVENT_PLAY_MOBILEPHONE_INFO)
EVENT_IVS_FINANCE_CONTRABAND_DETECT	0x00000301	Event of finance contraband detect(Corresponding to DEV_EVENT_FINANCE_CONTRABAND_DETECT_INFO)
EVENT_IVS_FINANCE_CASH_TRANSACTION	0x00000302	Event of finance cash transaction(Corresponding to DEV_EVENT_FINANCE_CASH_TRANSACTION_INFO)
EVENT_IVS_ANATOMY_TEMP_DETECT	0x00000303	Event of Intelligent detection of human body temperature(Corresponding to DEV_EVENT_ANATOMY_TEMP_DETECT_INFO)
EVENT_IVS_ACTIVITY_ANALYSE	0x00000304	Event of active analyse (just for video analyse rule configuration)
EVENT_IVS_DOOR_STATUS	0x00000305	Event of door status detect(Corresponding to DEV_EVENT_DOOR_STATUS_INFO)
EVENT_IVS_DHOP_CUSTOM	0x00000306	Event of DHOP custom(start/stop, Corresponding to DEV_EVENT_DHOP_CUSTOM_INFO)
EVENT_IVS_DHOP_CUSTOM_ONCE	0x00000307	Event of DHOP custom once(Pulse, Corresponding to DEV_EVENT_DHOP_CUSTOM_INFO)
EVENT_IVS_FOG_DETECTION	0x00000308	Event of fog detect(Corresponding to DEV_EVENT_FOG_DETECTION)
EVENT_IVS_TRAFFIC_VEHICLE_BC	0x00000309	Event of bc (Corresponding to DEV_EVENT_TRAFFIC_VEHICLE_BC)
EVENT_IVS_TRAFFIC_MOTOR_OVERLOAD	0x0000030A	Event of traffic motor overload(Corresponding to DEV_EVENT_TRAFFIC_MOTOR_OVERLOAD_INFO)
EVENT_IVS_TRAFFIC_PLATE_OCCLUSION	0x0000030B	Event of traffic plate occlusion(Corresponding to DEV_EVENT_TRAFFIC_PLATE_OCCLUSION_INFO)
EVENT_IVS_NONMOTOR_ENTRYING	0x0000030C	Event of non-motor vehicle access elevator(Corresponding to DEV_EVENT_NONMOTOR_ENTRYING_INFO)
EVENT_IVS_WATER_STAGE_MONITOR	0x0000030D	Event of water stage monitor, only used for task intelligent analysis (Corresponding to DEV_EVENT_WATER_STAGE_MONITOR_INFO)
EVENT_IVS_TRAFFIC_ROAD_ALERT	0x0000030E	Event of traffic road alert(Corresponding to DEV_EVENT_TRAFFIC_ROAD_ALERT_INFO)
EVENT_IVS_BREAK_RULE_BUILDING_DETECTION	0x0000030F	Event of break rule building detection(Corresponding to DEV_EVENT_BREAK_RULE_BUILDING_DETECTION_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR_RUN_REDLIGHT	0x00000310	Event of traffic non-motor run redlight (Corresponding to DEV_EVENT_TRAFFIC_NONMOTOR_RUN_REDLIGHT_INFO)

EVENT_IVS_TRAFFIC_VEHICLE_IN_EMERGENCY_LANE	0x00000311	Event of vehicle in emergency lane (Corresponding to DEV_EVENT_TRAFFIC_VEHICLE_IN_EMERGENCY_LANE_INFO)
EVENT_IVS_PRAM_DETECTION	0x00000312	Event of pram detection (Corresponding to DEV_EVENT_PRAM_DETECTION_INFO)
EVENT_IVS_STEREO_PRAM_DETECTION	0x00000313	Event of stereo pram detection (For rule configuration only)
EVENT_IVS_BIG_BAGGAGE_DETECTION	0x00000314	Event of big baggage detection (Corresponding to DEV_EVENT_BIG_BAGGAGE_DETECTION_INFO)
EVENT_IVS_STEREO_BIG_BAGGAGE_DETECTION	0x00000315	Event of stereo big baggage detection (For rule configuration only)
EVENT_IVS_TICKET_EVADE_DETECTION	0x00000316	Event of ticket evade detection (Corresponding to DEV_EVENT_TICKET_EVADE_DETECTION_INFO)
EVENT_IVS_STEREO_TICKET_EVADE_DETECTION	0x00000317	Event of stereo ticket evade detection (For rule configuration only)
EVENT_IVS_POWERLINE_FOREIGN_DETECTION	0x00000318	Event of PowerLine Foreign Detection (Corresponding to DEV_EVENT_POWERLINE_FOREIGN_DETECTION_INFO)
EVENT_IVS_TRAFFIC_OVER_GUIDE_LINE	0x00000319	Event of Over Guide line (Corresponding to DEV_EVENT_TRAFFIC_OVER_GUIDE_LINE_INFO)
EVENT_IVS_TRAFFIC_CAR_MEASUREMENT	0x00000320	Event of Traffic checkpoint measurement (vehicle length, width, height, weight, etc.) (Corresponding to DEV_EVENT_TRAFFIC_CAR_MEASUREMENT_INFO)
EVENT_IVS_TRAFFIC_WRONG_TURN_LIGHT	0x00000321	Event of Traffic wrong turn light (Corresponding to DEV_EVENT_TRAFFIC_WRONG_TURN_LIGHT_INFO)
EVENT_IVS_TRAFFIC_REAREND_ACCIDENT	0x00000322	Event of Traffic rearend accident (Corresponding to DEV_EVENT_TRAFFIC_REAREND_ACCIDENT_INFO)
EVENT_IVS_DO_TALK_ACTION	0x00000323	Event of Talk action (Corresponding to DEV_EVENT_DO_TALK_ACTION_INFO)
EVENT_IVS_FIRE_LANE_DETECTION	0x00000324	Event of Fire lane detection (Corresponding to DEV_EVENT_FIRE_LANE_DETECTION_INFO)
EVENT_IVS_PARKING_DETECTION_FOR_PRMA	0x00000325	Event of parking detection for PRMA (For rule configuration only)
EVENT_IVS_TRAFFIC_JAM_FOR_PRMA	0x00000326	Event of traffic jam for PRMA (For rule configuration only)
EVENT_IVS_TRAFFIC_ACCIDENT_FOR_PRMA	0x00000327	Event of traffic accident for PRMA (For rule configuration only)
EVENT_IVS_TRAFFIC_NON_MOTOR_RETROGRADE	0x00000328	Event of traffic non motor retrograde (Corresponding to DEV_EVENT_TRAFFIC_NON_MOTOR_RETROGRADE_INFO)
EVENT_IVS_TRAFFIC_NON_MOTOR_OVER_STOP_LINE	0x00000329	Event of traffic non motor over stop line (Corresponding to DEV_EVENT_TRAFFIC_NON_MOTOR_OVER_STOP_LINE_INFO)
EVENT_IVS_CAR_DRIVING_IN	0x00000330	Event of Cardriving in (Corresponding to

		DEV_EVENT_CAR_DRIVING_IN_INFO)
EVENT_IVS_CAR_DRIVING_OUT	0x00000331	Event of Car driving out(Corresponding to DEV_EVENT_CAR_DRIVING_OUT_INFO)
EVENT_IVS_PORTRAIT_DETECTION	0x00000332	Event of portrait detection (For rule configuration only)
EVENT_IVS_TRAFFIC_SPECIAL_VEHICLE_DETECT	0x00000333	Event of special vehicle detect (For rule configuration only)
EVENT_IVS_TRAFFIC_HEAD_LAMP_OFF	0x00000334	Event of traffic head lamp off(Corresponding to DEV_EVENT_TRAFFIC_HEAD_LAMP_OFF_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR	0x00000335	Event of traffic nonmotor detector(Corresponding to DEV_EVENT_TRAFFIC_NONMOTOR_INFO)
EVENT_IVS_TRAFFIC_BOARD	0x00000336	Event of traffic board detection (Corresponding to DEV_EVENT_TRAFFIC_BOARD_INFO)
EVENT_IVS_TRAFFIC_VISIBILITY	0x00000337	Event of traffic visibility detection (Corresponding to DEV_EVENT_TRAFFIC_VISIBILITY_INFO)
EVENT_IVS_TRAFFIC_VEHICLE_CLEANLINESS	0x00000338	Event of traffic vehicle cleanliness detection (Corresponding to DEV_EVENT_TRAFFIC_VEHICLE_CLEANLINESS_INFO)
EVENT_IVS_TRAFFICFLOW_FOR_PRMA	0x00000339	Event of Panorama TrafficFlow (For rule configuration only)
EVENT_IVS_TRUCKNOTCLEAN_FOR_PRMA	0x0000033A	Event of Trunk no clean Corresponding to DEV_EVENT_TRUCKNOTCLEAN_FOR_PRMA_INFO
EVENT_IVS_ROADOCCUPATION_BY_FOREIGNOBJECT	0x0000033B	Event of Road Occupation By ForeignObject (Corresponding to DEV_EVENT_ROADOCCUPATION_BY_FOREIGNOBJECT_INFO)
EVENT_IVS_TRAFFICFLOW_OVER	0x0000033C	Event of Trafficflow over (Corresponding to DEV_EVENT_TRAFFICFLOW_OVER_INFO)
EVENT_IVS_GOODS_DETECTION	0x0000033D	Event of Goods Detection (Corresponding to DEV_EVENT_GOODS_DETECTION_INFO)
EVENT_IVS_CONVEYORBLOCK_DETECTION	0x0000033E	Event of Conveyor obstruction alarm (Corresponding to DEV_EVENT_CONVEYORBLOCK_DETECTION_INFO)
EVENT_IVS_ANYTHING_DETECT	0x0000033F	Event of Anything Detection (Corresponding to DEV_EVENT_ANYTHING_DETECT_INFO)
EVENT_IVS_OBJECT_ABNORMAL	0x00000340	Event of Object abnormal(Corresponding to DEV_EVENT_OBJECT_ABNORMAL_INFO)
EVENT_IVS_DRIVE_ASSISTANT	0x00000341	Assisted driving (only for rule configuration)
EVENT_IVS_DRIVE_ACTION_ANALYSIS	0x00000342	Driving behavior analysis (only for rule configuration)
EVENT_IVS_DRIVE_HANDSOFF_STEERING_WHEEL	0x00000343	Driving behavior analysis detection of hand leaving the steering wheel (Corresponding to DEV_EVENT_DRIVE_HANDSOFF_STEERING_WHEEL_INFO)
EVENT_IVS_DRIVE_BLIND_SPOT	0x00000344	Driving behavior analysis Blind spot detection (only for rule configuration)
EVENT_IVS_ARTICLE_DETECTION	0x00000345	Article Detection for rule configuration,

		corresponding to EVENT_IVS_LEFTDETECTION or EVENT_IVS_TAKENAWAYDETECTION
EVENT_IVS_TRAFFIC_PARKINGSPACE_MANUALSNAP	0x00000346	Manual capture of roadside parking spaces (Corresponding to DEV_EVENT_PARKINGSPACE_MANUALSNAP_INFO)
EVENT_IVS_STREET_SUNCURE	0x00000347	Sun drying incident along the street (Corresponding to DEV_EVENT_STREET_SUNCURE_INFO)
EVENT_IVS_OUTDOOR_ADVERTISEMENT	0x00000348	Outdoor advertising event (Corresponding to DEV_EVENT_OUTDOOR_ADVERTISEMENT_INFO)
EVENT_IVS_HUDDLE_MATERIAL	0x00000349	Random material detection event (Corresponding to DEV_EVENT_HUDDLE_MATERIAL_INFO)
EVENT_IVS_FIRE_LINE_DETECTION	0x0000034A	Detection of access to fire fighting (Corresponding to DEV_EVENT_FIRE_LINE_DETECTION_INFO)
EVENT_IVS_OCCUPY_BUS_LANE	0x0000034B	Illegal occupation of bus lanes(Corresponding to DEV_EVENT_OCCUPY_BUS_LANE_INFO)
EVENT_IVS_DISTRESS_DETECTION	0x0000034C	Detection of Distress (Corresponding to DEV_EVENT_DISTRESS_DETECTION_INFO)
EVENT_IVS_TRAFFIC_ASSISTANT_WITHOUT_SAFELELT	0x0000034D	Event of assistant without safebelt (Corresponding to DEV_EVENT_TRAFFIC_ASSISTANT_WITHOUT_SAFELELT)
EVENT_IVS_TRAFFIC_SPEED_CHANGE_DETECTION	0x0000034E	Traffic Speed Change Detection(Corresponding DEV_EVENT_TRAFFIC_SPEED_CHANGE_DETECTION_INFO)
EVENT_IVS_FOLLOW_CAR_ALARM	0x0000034F	Car following alarm(Corresponding to DEV_EVENT_FOLLOW_CAR_ALARM_INFO)
EVENT_IVS_CONVEYER_BELT_DETECT	0x00000350	Conveyor detection, only for rule configuration, corresponding to rule structure CFG_CONVEYER_BELT_DETECT_INFO, Corresponding to event
EVENT_IVS_CONVEYER_BELT_BULK	0x00000351	Large foreign body detection event of conveyor belt(Corresponding to DEV_EVENT_CONVEYER_BELT_BULK_INFO)
EVENT_IVS_CONVEYER_BELT_NONLOAD	0x00000352	Non load detection event of conveyor belt(Corresponding to DEV_EVENT_CONVEYER_BELT_NONLOAD_INFO)
EVENT_IVS_CONVEYER_BELT_RUNOFF	0x00000353	Run off detection event of conveyor belt(Corresponding to DEV_EVENT_CONVEYER_BELT_RUNOFF_INFO)
EVENT_IVS_CONVEYER_BELT_COAL_RATIO	0x00000354	Conveyor belt coal detection(Corresponding to DEV_EVENT_CONVEYER_BELT_COAL_RATIO_INFO)
EVENT_IVS_PACKBROKEN_DETECTION	0x00000355	Event of pack broken detection (Corresponding to DEV_EVENT_PACKBROKEN_DETECTION_INFO)
EVENT_IVS_PACKLAND_DETECTION	0x00000356	Event of pack land detection (Corresponding to DEV_EVENT_PACKLAND_DETECTION_INFO)
EVENT_IVS_PACKOPEN_DETECTION	0x00000357	Event of pack open (Corresponding to DEV_EVENT_PACKOPEN_DETECTION_INFO)

EVENT_IVS_TRAFFIC_TURN_RIGHT_NO_STOP	0x00000358	Event of Right turn and start again (Corresponding to DEV_EVENT_TRAFFIC_TURN_RIGHT_NO_STOP_INFO)
EVENT_IVS_TRAFFIC_CROSSING_GUARDRAIL	0x00000359	Event of crossing guardrail (Corresponding to DEV_EVENT_TRAFFIC_CROSSING_GUARDRAIL_INFO)
EVENT_IVS_EMERGENCY	0x0000035A	The emergency triggered by human is usually handled by the linkage of external communication function and request for help(Corresponding to DEV_EVENT_EMERGENCY_INFO)
EVENT_IVS_TRAFFIC_PARKING_STATISTICS	0x0000035B	event to Parking statistics(Corresponding to DEV_EVENT_TRAFFIC_PARKING_STATISTICS_INFO)
EVENT_IVS_HEAT_IMAGING_TEMP	0x0000035C	Thermal temperature abnormal event alarm(Corresponding to DEV_EVENT_HEAT_IMAGING_TEMPER_INFO)
EVENT_IVS_SCALPER_ALARM	0x0000035D	Event of Scalper Alarm (Corresponding to DEV_EVENT_SCALPER_ALARM_INFO)
EVENT_IVS_ROAD_OBSTACLE_DETECTION	0x0000035E	event to Road obstacle detection (Corresponding to DEV_EVENT_ROAD_OBSTACLE_DETECTION_INFO)
EVENT_IVS_TRAFFIC_BRIGANDAGE_CAR	0x0000035F	Event of brigandage car (Corresponding to DEV_EVENT_TRAFFIC_BRIGANDAGE_CAR_INFO)
EVENT_IVS_TRAFFIC_COUNTERFEIT_PLATE_CAR	0x00000360	Event of counterfeit plate car (Corresponding to DEV_EVENT_TRAFFIC_COUNTERFEIT_PLATE_CAR_INFO)
EVENT_IVS_TRAFFIC_FAKE_PLATE_CAR	0x00000361	Event of fake plate car(Corresponding to DEV_EVENT_TRAFFIC_FAKE_PLATE_CAR_INFO)
EVENT_IVS_SEWAGE_DETECTION	0x00000362	Pollution detection event(Corresponding to DEV_EVENT_SEWAGE_DETECTION_INFO)
EVENT_IVS_WATERCOLOR_DETECTION	0x00000363	Water color event(Corresponding to DEV_EVENT_WATERCOLOR_DETECTION_INFO)
EVENT_IVS_TRAFFIC_MOTORCYCLE_FORBID	0x00000364	Event of traffic motorcycle forbid (Corresponding to DEV_EVENT_TRAFFIC_MOTORCYCLE_FORBID)
EVENT_IVS_VIDEO_NORMAL_DETECTION	0x00000365	Video normal events,At the end of the video diagnostic detection cycle, the diagnostic items that have not reported errors are reported to the normal events DEV_EVENT_VIDEO_NORMAL_DETECTION_INFO
EVENT_IVS_TRAFFIC_TURN_RIGHT_OVER_LINE	0x00000366	Right turn line pressing event(Corresponding to DEV_EVENT_TRAFFIC_TURN_RIGHT_OVER_LINE_INFO)
EVENT_IVS_MANUAL_ALARM	0x00000367	Manual alarm event (Corresponding to DEV_EVENT_MANUAL_ALARM_INFO)
EVENT_IVS_TRAFFIC_DRIVE_ON_LINE	0x00000368	Line-riding incident(Corresponding to DEV_EVENT_TRAFFIC_DRIVE_ON_LINE_INFO)
EVENT_IVS_OBJECT_PLACEMENT_DETECTION	0x00000369	Item placement detection event(Corresponding to DEV_EVENT_OBJECT_PLACEMENT_DETECTION_INFO)
EVENT_IVS_OBJECT_REMOVAL_DETECTION	0x0000036A	Item removal detection event(Corresponding to DEV_EVENT_OBJECT_REMOVAL_DETECTION_INFO)

EVENT_IVS_FIRE_DOOR_DETECTION	0x0000036B	Fire door detection event(Corresponding to DEV_EVENT_FIRE_DOOR_DETECTION_INFO)
EVENT_IVS_FIRE_EXTINGUISHER_DETECTION	0x0000036C	Fire extinguisher detection event(Corresponding to DEV_EVENT_FIRE_EXTINGUISHER_DETECTION_INFO)
EVENT_IVS_DOOR_NOT_CLOSE	0x0000036D	The door is not closed event(Corresponding to DEV_EVENT_DOOR_NOT_CLOSE_INFO)
EVENT_IVS_VEHICLE_PERIPHERAL_ALARM	0x0000036E	Vehicle peripheral event(Corresponding to DEV_EVENT_VEHICLE_PERIPHERAL_ALARM_INFO)
EVENT_IVS_PARKING_LIMIT_DETECTION	0x0000036F	Parking threshold change alarm(Corresponding to DEV_EVENT_PARKING_LIMIT_DETECTION_INFO)
EVENT_IVS_PARKING_STATUS_CHANGE_DETECTION	0x00000370	Parking status change alarm(Corresponding to DEV_EVENT_PARKING_STATUS_CHANGE_DETECTION_INFO)
EVENT_IVS_DIALRECOGNITION	0x00000371	Dial recognition alarm(Corresponding to DEV_EVENT_DIALRECOGNITION_INFO)
EVENT_IVS_ELECTRICFAULT_DETECTION	0x00000372	Electric defect detection alarm(Corresponding to DEV_EVENT_ELECTRICFAULTDETECT_INFO)
EVENT_IVS_TRASH_WITHOUT_COVER_DETECTION	0x00000373	Detection event of uncovered trash can(Corresponding to DEV_EVENT_TRASH_WITHOUT_COVER_DETECTION_INFO)
EVENT_IVS_SECURITY_INSPECTOR_LOOKAROUND	0x00000374	Security Inspector Look Around event(Corresponding to DEV_EVENT_SECURITY_INSPECTOR_LOOKAROUND_INFO)
EVENT_IVS_SECURITY_INSPECTOR_LOWERHEAD	0x00000375	Security Inspector Lower Head event(Corresponding to DEV_EVENT_SECURITY_INSPECTOR_LOWERHEAD_INFO)
EVENT_IVS_SECURITY_INSPECTOR_TIRED	0x00000376	Security Inspector Tired event(Corresponding to DEV_EVENT_SECURITY_INSPECTOR_TIRED_INFO)
EVENT_IVS_SECURITY_INSPECTOR_YAWN	0x00000377	Security Inspector Yawn event(Corresponding to DEV_EVENT_SECURITY_INSPECTOR_YAWN_INFO)
EVENT_IVS_AI_PICK_DETECT	0x00000378	AI Pick Detect Event(Corresponding to DEV_EVENT_AI_PICK_DETECT_INFO)
EVENT_IVS_TRAFFIC_LIGHT_FAULT	0x00000379	Traffic light failure alarm event(Corresponding to DEV_EVENT_TRAFFIC_LIGHT_FAULT_INFO)
EVENT_IVS_TRAFFIC_LANE_INDICATOR_FAULT	0x0000037A	Traffic lane change sign failure event(Corresponding to DEV_EVENT_TRAFFIC_LANE_INDICATOR_FAULT_INFO)
EVENT_IVS_WATER_SPEED_DETECTION	0x0000037B	Water velocity detection event (Corresponding to DEV_EVENT_WATER_SPEED_DETECTION_INFO)
EVENT_IVS_TRAFFIC_PARKING_BACKING	0x0000037C	PARKING BACKING Event (Corresponding to DEV_EVENT_TRAFFIC_PARKING_BACKING_INFO)#define EVENT_IVS_VALVE_ABNORMAL0x0000037D

EVENT_IVS_VALVE_ABNORMAL	0x0000037D	Valve abnormality detection event (corresponding to DEV_EVENT_VALVE_ABNORMAL_INFO)
EVENT_IVS_DISCHARGE_PORT_ABNORMAL	0x0000037E	Outlet abnormal detection event(Corresponding to DEV_EVENT_DISCHARGE_PORT_ABNORMAL_INFO)
EVENT_IVS_TRAFFIC_MOTOR_STRICTLY_PROHIBIT_MANNED	0x0000037F	Illegal manned event (Corresponding to DEV_EVENT_TRAFFIC_MOTOR_STRICTLY_PROHIBIT_MANNED_INFO)
EVENT_IVS_BARELAND_DETECTION	0x00000380	Bare soil detection event(DEV_EVENT_BARELAND_DETECTION_INFO)
EVENT_IVS_CONSUMPTION_EVENT	0x00000381	Consumption event (Corresponding to DEV_EVENT_CONSUMPTION_EVENT_INFO)
EVENT_IVS_TOUCH_ELECTROSTATIC_BALL	0x00000382	touch electrostatic ball(corresponding to DEV_EVENT_TOUCH_ELECTROSTATIC_BALL_INFO)
EVENT_IVS_OXYGEN_CYLINDER_DETECTION	0x00000383	oxygen cylinder detection(corresponding to DEV_EVENT_OXYGEN_CYLINDER_DETECTION_INFO)
EVENT_IVS_XRAY_UNPACKING_CHECK	0x00000384	XRAY Unpacking Check(corresponding to DEV_EVENT_XRAY_UNPACKING_CHECK_INFO)
EVENT_IVS_GENERAL_ATTITUDE_DETECTION	0x00000385	General Attitude Detection(corresponding to DEV_EVENT_GENERAL_ATTITUDE_DETECTION_INFO)
EVENT_IVS_SNAP_TASK	0x00000386	Snap Task Event(corresponding to DEV_EVENT_SNAP_TASK_INFO)
EVENT_IVS_TRAFFIC_CHANGE_LANE_CONTINUES	0x00000387	Traffic Change Lane Continues Event(corresponding to DEV_EVENT_TRAFFIC_CHANGE_LANE_CONTINUES_INFO)
EVENT_IVS_SPRAY_DETECT	0x00000388	Spray Detect Event(corresponding to DEV_EVENT_SPRAY_DETECT_INFO)
EVENT_IVS_SILICON_FIRE_DETECTION	0x00000389	Prick detection event(corresponding to DEV_EVENT_SILICON_FIRE_DETECTION_INFO)
EVENT_IVS_TRAFFIC_PARKING_ON_RIGHT_TURN_ROUTE	0x0000038A	Traffic Parking On Right Turn Route(Corresponding to DEV_EVENT_TRAFFIC_PARKING_ON_RIGHT_TURN_ROUTE_INFO)
EVENT_IVS_PERSONNEL_COEXISTENCE	0x0000038B	Personnel Coexistence Event(Corresponding to DEV_EVENT_PERSONNEL_COEXISTENCE_INFO)
EVENT_IVS_GENEAL_ATTITUDE	0x0000038C	Geneal Attitude Event(Corresponding to DEV_EVENT_GENEAL_ATTITUDE_INFO)
EVENT_IVS_CASH_BOX_STATE	0x0000038D	Cash Box State Event(Corresponding to DEV_EVENT_CASH_BOX_STATE_INFO)
EVENT_IVS_LEAKAGE_DETECTION	0x0000038E	Leakage detection event(Corresponding to DEV_EVENT_LEAKAGE_DETECTION_INFO)
EVENT_IVS_LADLE_NO_DETECTION	0x0000038F	Ladle number identification event(corresponding to DEV_EVENT_LADLE_NO_DETECTION_INFO)
EVENT_IVS_FISHING_DETECTION	0x00000390	Fishing Detection Event(corresponding to DEV_EVENT_FISHING_DETECTION_INFO)
EVENT_IVS_ALARM_ACCESSORY	0x00000391	Alarm accessory events(Corresponding to DEV_EVENT_ALARM_ACCESSORY_INFO)

EVENT_IVS_WIRELESS_DEV_LOWP OWER	0x00000392	Wireless device low battery event(Corresponding to DEV_EVENT_WIRELESS_DEV_LOWPOWER_INFO)
EVENT_IVS_SENSOR_ABNORMAL	0x00000393	Detector abnormal alarm(Corresponding to DEV_EVENT_SENSOR_ABNORMAL_INFO)
EVENT_IVS_MODULE_LOST	0x00000394	Module disconnection alarm event(Corresponding to DEV_EVENT_MODULE_LOST_INFO)
EVENT_IVS_CROWD_LEVEL_DETEC TION	0x00000395	Congestion detection event(Corresponding to DEV_EVENT_CROWD_LEVEL_DETECTION_INFO)
EVENT_IVS_REGION_DEFENSE_DE TECTION	0x00000396	Regional prevention events(Corresponding to DEV_EVENT_REGION_DEFENSE_DETECTION_INFO)
EVENT_IVS_DUSTBIN_DETECTION	0x00000397	Trash can detection event(Corresponding to DEV_EVENT_DUSTBIN_DETECTION_INFO)
EVENT_IVS_DIALRECOGNITION_EX	0x00000398	Dial recognition alarm(Corresponding to DEV_EVENT_DIALRECOGNITION_INFO)
EVENT_IVS_OCR_DETECTION	0x00000399	OCR detection event(Corresponding to DEV_EVENT_OCR_DETECTION_INFO)
EVENT_IVS_ROAD_CONDITIONS_D ETECTION	0x0000039A	Road detection event(Corresponding to DEV_EVENT_ROAD_CONDITIONS_DETECTION_INFO)
EVENT_IVS_NEAR_OBJECT_DETEC T	0x0000039B	Near object detect event(Corresponding to DEV_EVENT_NEAR_OBJECT_DETECT_INFO)
EVENT_IVS_OBJECT_NUM_DETECT ION	0x0000039C	Object number detection event(Corresponding to DEV_EVENT_OBJECT_NUM_DETECTION_INFO, note that this event does not support separate use, only as a rule in "Open Intelligent Event (EVENT_IVS_OPEN_INTELLI)")
EVENT_IVS_OPEN_INTELLI	0x0000039D	Open Intelligent Event(Corresponding to DEV_EVENT_OPEN_INTELLI_INFO)
EVENT_IVS_CRANE_LOAD_STAY_ DETECTION	0x0000039E	Crane load stay detection event(Corresponding to DEV_EVENT_CRANE_LOAD_STAY_DETECTION_INFO)
EVENT_IVS_TRAFFIC_DRIVER_NO_ BELT	0x00000400	Not wearing seat belt alarm event(Corresponding to DEV_EVENT_TRAFFIC_DRIVER_NO_BELT_INFO)
EVENT_IVS_RIDING_MOTOR_CYCL E	0x00000401	Motorcycle straddle detection event(Corresponding to DEV_EVENT_RIDING_MOTOR_CYCLE_INFO)
EVENT_IVS_CONVEYOR_ARTICLE_ TYPE	0x00000402	Conveyor Article Type event(Corresponding to DEV_EVENT_CONVEYOR_ARTICLE_TYPE_INFO)
EVENT_IVS_BLIND_ALARM	0x00000403	Blind Alarm event(Corresponding to DEV_EVENT_BLIND_ALARM_INFO)
EVENT_IVS_TRAFFIC_SPEED_DROP _SHARPLY	0x00000404	Traffic Speed Drop Sharply event(Corresponding to DEV_EVENT_TRAFFIC_SPEED_DROP_SHARPLY_INFO)
EVENT_IVS_GESTURE_DETECTION	0x00000405	Gesture detection event(Corresponding to DEV_EVENT_GESTURE_DETECTION_INFO)
EVENT_IVS_HEAD_LIFT_DETECTIO N	0x00000406	Head Lift Detection(Corresponding to DEV_EVENT_HEAD_LIFT_DETECTION_INFO)
EVENT_IVS_DRAINING_DETECTION	0x00000407	Draining Detection event(Corresponding to DEV_EVENT_DRAINING_DETECTION_INFO)
EVENT_IVS_TRAFFIC_CROSSING_S	0x00000408	Zebra crossing non deceleration event

PEEDY		(Corresponding to DEV_EVENT_TRAFFIC_CROSSING_SPEEDY_INFO)
EVENT_IVS_TRAFFIC_LARGE_CAR_NO_STOP	0x00000409	Event of crane turning right without stopping(Corresponding to DEV_EVENT_TRAFFIC_LARGE_CAR_NO_STOP_INFO)
EVENT_IVS_TRAFFIC_OVERTAKE_ONRIGHT	0x0000040A	Right overtaking event(Corresponding to DEV_EVENT_TRAFFIC_OVERTAKE_ONRIGHT_INFO)
EVENT_IVS_TRAFFIC_TRUCK_OCCUPIED	0x0000040B	Truck occupation event(Corresponding to DEV_EVENT_TRAFFIC_TRUCK_OCCUPIED_INFO)
EVENT_IVS_TRAFFIC_SMALL_DISPLACEMENT	0x0000040C	Vehicle slow motion event(Corresponding to DEV_EVENT_TRAFFIC_SMALL_DISPLACEMENT_INFO)
EVENT_IVS_GREEN_BELT	0x0000040D	Green belt alarm event(Corresponding to DEV_EVENT_GREEN_BELT_ALARM_INFO)
EVENT_IVS_ROAD_DAMAGE	0x0000040E	Road damage alarm event(Corresponding to DEV_EVENT_ROAD_DAMAGE_ALARM_INFO)
EVENT_IVS_TRAFFIC_SERPENTINE_CHANGE_LANE	0x0000040F	Traffic Serpentine Change Lane event(Corresponding to DEV_EVENT_TRAFFIC_SERPENTINE_CHANGE_LANE_I NFO)
EVENT_IVS_PERSON_TRANS_DETECTION	0x00000410	Personnel transmission detection event(Corresponding to DEV_EVENT_IVS_PERSON_TRANS_DETECTION_INFO)
EVENT_IVS_HUMAN_ANIMAL_COEXISTENCE	0x00000411	Human Animal Coexistence Event(Corresponding to DEV_EVENT_HUMAN_ANIMAL_COEXISTENCE_INFO)
EVENT_IVS_TANK_CAPACITY_DETECTION	0x00000412	Tank Capacity Detection Event(Corresponding to DEV_EVENT_IVS_TANK_CAPACITY_DETECTION_INFO)
EVENT_IVS_TANK_DUMPING_DETECTION	0x00000413	Tank Dumping Detection Event(Corresponding to DEV_EVENT_IVS_TANK_DUMPING_DETECTION_INFO)
EVENT_IVS_TANK_OVERFLOW_DETECTION	0x00000414	Tank Overflow Detection Event(Corresponding to DEV_EVENT_IVS_TANK_OVERFLOW_DETECTION_INF O)
EVENT_IVS_DUSTBIN_RETREAT	0x00000416	Trash can evacuation detection event(Corresponding to NET_DEV_EVENT_DUSTBIN_RETREAT_INFO)
EVENT_IVS_SCRAPSTEEL_DETECT	0x00000417	Scrapsteel Detect Event(Corresponding to NET_DEV_EVENT_SCRAPSTEEL_DETECT_INFO)
EVENT_IVS_SIGNAL_LIGHT_ON_DETECTION	0x00000418	Detection event of signal light on at the same time(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHT_ON_DETECTION_IN FO)
EVENT_IVS_SIGNAL_LIGHT_FLASH_YELLOW_DETECTION	0x00000419	Monochrome light yellow flash detection event(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHT_FLASHYELLOW_DE TECTION_INFO)
EVENT_IVS_SIGNAL_LIGHT_NOBRIGHT_DETECTION	0x0000041A	Detection event of insufficient brightness of signal lamp(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHT_NOBRIGHT_DETECT

		ION_INFO)
EVENT_IVS_SIGNAL_LIGHT_BLOCKING_DETECTION	0x0000041B	Signal light occlusion detection event(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHT_BLOCKING_DETECTION_INFO)
EVENT_IVS_SIGNAL_LIGHT_ALWAYS_ON_DETECTION	0x0000041C	Detection event of monochrome light always on(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHT_ALWAYS_ON_DETECTION_INFO)
EVENT_IVS_LEDSCREEN_NOTLIGHTUP_DETECTION	0x0000041D	Detection event of LED screen not on(Corresponding to NET_DEV_EVENT_LEDSCREEN_NOTLIGHTUP_DETECTION_INFO)
EVENT_IVS_SIGNAL_LIGHTOFF_DETECTION	0x0000041E	Detection event when the signal light is not on(Corresponding to NET_DEV_EVENT_SIGNAL_LIGHTOFF_DETECTION_INFO)
EVENT_IVS_PIC_COMPARE_DETECTION	0x0000041F	The picture cannot be compared with the alarm event(Corresponding to NET_DEV_EVENT_PIC_COMPARE_DETECTION_INFO)
EVENT_IVS_CHANGE_DETECTION	0x00000420	Change detection(Corresponding to NET_DEV_EVENT_CHANGE_DETECTION_INFO)
EVENT_IVS_LINK_AGE_RADAR_ALARM	0x00000421	Linkage event (sent as detail event and linkage ball machine)(Corresponding to NET_DEV_EVENT_LINK_AGE_RADAR_ALARM_INFO)
EVENT_IVS_VEHICLE_STATE	0x00000422	Vehicle status event(Corresponding to NET_DEV_EVENT_VEHICLE_STATE_INFO)
EVENT_IVS_PASS_WINDOW_DELIVERY	0x00000423	Pass Window Delivery event(Corresponding to NET_DEV_EVENT_PASS_WINDOW_DELIVERY_INFO)
EVENT_IVS_DOOR_STATE_DETECTION	0x00000424	Door opening and closing detection event(Corresponding to NET_DEV_EVENT_DOOR_STATE_DETECTION_INFO)
EVENT_IVS_WASTE_MIXED_INVEST	0x00000425	Mixed waste disposal event(Corresponding to NET_DEV_EVENT_WASTE_MIXED_INVEST_INFO)
EVENT_IVS_UNBROKEN_TRASHBAG	0x00000426	Detection event of garbage bag not broken(Corresponding to NET_DEV_EVENT_UNBROKEN_TRASHBAG_INFO)
EVENT_IVS_PERSON_CARRY_TRASHBAG	0x00000427	Bag carrying alarm event(Corresponding to NET_DEV_EVENT_PERSON_CARRY_TRASHBAG_INFO)
EVENT_IVS_UNATTENDED_DETENTION	0x00000428	Unattended detection event(Corresponding to NET_DEV_EVENT_UNATTENDED_DETENTION_INFO)
EVENT_IVS_DROP_DETECTION	0x00000429	Drip detection event(Corresponding to NET_DEV_EVENT_DROP_DETECTION_INFO)
EVENT_IVS_TEMPERATURE_ALARM	0x0000042A	Temperature Alarm(Corresponding to NET_DEV_EVENT_TEMPERATURE_ALARM_INFO)
EVENT_IVS_HUMIDITY_ALARM	0x0000042B	Humidity Alarm(Corresponding to

		NET_DEV_EVENT_HUMIDITY_ALARM_INFO)
EVENT_IVS_POWER_SWITCHER_ALARM	0x0000042C	Power Switcher Alarm(Corresponding to NET_DEV_EVENT_POWER_SWITCHER_ALARM_INFO)
EVENT_IVS_GAS_TANK_DETECTION	0x0000042D	Gas tank detection event(Corresponding to NET_DEV_EVENT_GAS_TANK_DETECTION_INFO)
EVENT_IVS_TRAFFIC_OCCUPYING_THEPATH	0x0000042E	Occupying The Path event(Corresponding to NET_EVENT_TRAFFIC_OCCUPYING_THEPATH_INFO)
EVENT_IVS_ILLEGAL_CARRIAGE	0x0000042F	Illegal Carriage event(Corresponding to NET_DEV_EVENT_ILLEGAL_CARRIAGE_INFO)
EVENT_IVS_SILICON_FIREPUTTER_DETECTION	0x00000430	Silicon Fire Putter Detection event(Corresponding to NET_EVENT_SILICON_FIREPUTTER_DETECTION_INFO)
EVENT_IVS_ALARM_METHANE_ALARM	0x00000431	Methane alarm event(Corresponding to NET_DEV_EVENT_METHANE_ALARM_INFO)
EVENT_IVS_ALARM_TORPEDO_DETECT	0x00000432	Torpedo detect alarm event(Corresponding to NET_DEV_EVENT_TORPEDO_DETECT_ALARM_INFO)
EVENT_IVS_FINANCIAL_CABINET_ALARM_EVENT	0x00000433	Financial Cabinet Alarm event(Corresponding to NET_DEV_EVENT_FINANCIAL_CABINET_ALARM_EVENT_INFO)
EVENT_IVS_SCRAPSTEEL_DANGER_DETECT	0x00000434	Scrap dangerous goods incident(Corresponding to NET_DEV_EVENT_SCRAPSTEEL_DANGER_DETECT_INFO)
EVENT_IVS_TRAFFIC_RUNASTOP_SIGN	0x00000435	Breaking stop sign event(Corresponding to NET_DEV_EVENT_TRAFFIC_RUNASTOP_SIGN_INFO)
EVENT_IVS_IRCUT_MODESWITCH_ALARM_EVENT	0x00000436	IRCUT mode switch event(Corresponding to NET_DEV_EVENT_IRCUT_MODESWITCH_ALARM_INFO)
EVENT_IVS_TRAFFIC_VEHICLE_OVERLOAD	0x00000437	Vehicle overload event(Corresponding to NET_DEV_EVENT_TRAFFIC_VEHICLE_OVERLOAD_INFO)
EVENT_IVS_REMOTE_APPROVAL_ALARM	0x00000438	Financial remote approval event(Corresponding to NET_DEV_EVENT_REMOTE_APPROVAL_ALARM_INFO)
EVENT_IVS_ANTI_COUNTERFEIT	0x00000439	Anti-counterfeiting detection event(Corresponding to NET_DEV_EVENT_ANTI_COUNTERFEIT_INFO)
EVENT_IVS_TRAFFIC_SPEED_DIFFPREWARNING	0x0000043A	Traffic Speed Diff Prewarning event(Corresponding to NET_DEV_EVENT_TRAFFIC_SPEED_DIFFPREWARNING_INFO)
EVENT_IVS_TRAFFIC_DRIVER_IDENTIFIED	0x0000043B	Driver comparison success notification event(Corresponding to NET_DEV_EVENT_TRAFFIC_DRIVER_IDENTIFIED_INFO)
EVENT_IVS_TRAFFIC_QUEUE_OVERFLOW	0x0000043C	Queue overflow event (corresponding to net_dev_event_traffic_queue_overflow_info)
EVENT_IVS_TRAFFIC_QUEUE_TIMEOUT	0x0000043D	The queue exceeds the threshold (corresponding to NET_DEV_EVENT_TRAFFIC_QUEUE_TIMEOUT_INFO)
EVENT_IVS_RAILING_PASS_DETECTOR	0x0000043E	passing things across the fence event(corresponding

TION		to NET_DEV_EVENT_RAILING_PASS_DETECTION_INFO)
EVENT_IVS_MULTI_MAN_NUM_DETECTION	0x0000043F	Interrogation room number alarm event (corresponding to NET_DEV_EVENT_MULTI_MAN_NUM_DETECTION_INFO)
EVENT_IVS_OBJECT_QUANTITY_DETECTION	0x00000440	Target type and quantity detection alarm events (corresponding to NET_DEV_EVENT_OBJECT_QUANTITY_DETECTION_INFO)
EVENT_IVS_USERMANAGER_FOR_TWSDK	0x00000441	User information reporting event(corresponding to NET_DEV_EVENT_USERMANAGER_FOR_TWSDK_INFO)
EVENT_IVS_DRIVE_ASSISTANT_ALARM	0x00000442	Drive Assistant Alarm(corresponding to NET_DEV_EVENT_DRIVE_ASSISTANT_ALARM_INFO)
EVENT_IVS_CROSSLINE_STAT	0x00000443	Crossline stat(corresponding to NET_DEV_EVENT_CROSSLINE_STAT_INFO),(note that this event does not support separate use, only as a rule in "Open Intelligent Event (EVENT_IVS_OPEN_INTELLI)")
EVENT_IVS_REGIONNUM_STAT	0x00000444	Regionnum stat(corresponding to NET_DEV_EVENT_REGIONNUM_STAT_INFO),(note that this event does not support separate use, only as a rule in "Open Intelligent Event (EVENT_IVS_OPEN_INTELLI)")
EVENT_IVS_FISH_STATE_DETECTION	0x00000445	Fish State Detection event(corresponding to NET_DEV_EVENT_FISH_STATE_DETECTION_INFO)
EVENT_IVS_CONVEYOR_NTH_TO_STH_DETECTION	0x00000446	Conveyor Nothing To Something Detection(corresponding to NET_DEV_EVENT_CONVEYOR_NTH_TO_STH_DETECTION_INFO)
EVENT_IVS_POSITION_SNAP	0x00000447	Snap with position(corresponding to NET_DEV_EVENT_POSITION_SNAP_INFO)
EVENT_IVS_DRIVER_MISMATCH_CERTIFICATE	0x00000448	Driver Mismatch Certificate event(corresponding to NET_DEV_EVENT_DRIVER_MISMATCH_CERTIFICATE_INFO)
EVENT_IVS_DRIVER_MISMATCH_VEHICLE	0x00000449	Driver Mismatch Vehicle event(corresponding to NET_DEV_EVENT_DRIVER_MISMATCH_VEHICLE_INFO)
EVENT_IVS_CERTIFICATE_MISMATCH_VEHICLE	0x0000044A	Certificate Mismatch Vehicle event(corresponding to NET_DEV_EVENT_CERTIFICATE_MISMATCH_VEHICLE_INFO)
EVENT_IVS_XRAY_DETECT_PACKAGE	0x0000044B	XRay Detect Package event (Corresponding to NET_EVENT_XRAY_DETECT_PACKAGE_INFO)
EVENT_IVS_CONVEYOR_STH_TO_NTH_DETECTION	0x0000044C	Conveyor Something To Nothing Detection(corresponding to NET_DEV_EVENT_CONVEYOR_STH_TO_NTH_DETECTION_INFO)

		ON_INFO)
EVENT_IVS_CO_LOW_ALARM	0x0000044D	CO low concentration alarm event(corresponding to NET_DEV_EVENT_CO_LOW_ALARM_INFO)
EVENT_IVS_CO_HIGH_ALARM	0x0000044E	CO high concentration alarm event(corresponding to NET_DEV_EVENT_CO_HIGH_ALARM_INFO)
EVENT_IVS_TIMECHANGE_FOR_TWSDK	0x0000044F	System time modified alarm event(Timewatch)(corresponding to NET_DEV_EVENT_TIMECHANGE_FOR_TWSDK_INFO)
EVENT_IVS_CIGARETTE_CASE_DETECTION	0x00000450	Cigarette case detection event(corresponding to NET_DEV_EVENT_CIGARETTE_CASE_DETECTION_INFO)
EVENT_IVS_CONVEYOR_BELT_STATUS	0x00000451	Conveyor Belt Status event(corresponding to NET_DEV_EVENT_CONVEYOR_BELT_STATUS_INFO)
EVENT_IVS_HOSPITAL_TALK_CONTROL	0x00000452	hospital talk control(corresponding to NET_DEV_EVENT_HOSPITAL_TALK_CONTROL_INFO)
EVENT_IVS_PASSENGER_FLOW_ALARM	0x00000453	Passenger Flow Alarm(corresponding to NET_DEV_EVENT_PASSENGER_FLOW_ALARM_INFO)
EVENT_IVS_SCHOOL_BUS_SWIPE_CARD	0x00000454	Student Card swiping Event (corresponding to NET_DEV_EVENT_SCHOOL_BUS_SWIPE_CARD_INFO)
EVENT_IVS_COLD_SPOT_WARNING	0x00000455	Cold spot warning (corresponding to NET_DEV_EVENT_COLD_SPOT_WARNING_INFO)
EVENT_IVS_TRAFFIC_PLATE_ABNORMAL	0x00000456	Traffic plate abnormal(corresponding to NET_DEV_EVENT_TRAFFIC_PLATE_ABNORMAL_INFO)
EVENT_IVS_TRAFFIC_ACCELERATION_RAPID	0x00000457	Traffic Acceleration Rapid Event(corresponding to NET_DEV_EVENT_TRAFFIC_ACCELERATION_RAPID_INFO)
EVENT_IVS_TRAFFIC_TURN_SHARP	0x00000458	Traffic Turn Sharp Event(corresponding to NET_DEV_EVENT_TRAFFIC_TURN_SHARP_INFO)
EVENT_IVS_GARBAGE_PLASTICBAG	0x00000459	Garbage Plasticbag Event(corresponding to NET_DEV_EVENT_GARBAGE_PLASTICBAG_INFO)
EVENT_IVS_DOOR_STATUS_FOR_BOX	0x0000045A	Smart cabinet door status report (corresponding to NET_DEV_EVENT_DOOR_STATUS_FOR_BOX_INFO)
EVENT_IVS_COLLISION_CONFLICT	0x0000045B	Collision Conflict Event(corresponding to NET_DEV_EVENT_COLLISION_CONFLICT_INFO)
EVENT_IVS_PHOTOGRAPH_DETECTION	0x0000045C	Photograph Detection Event(corresponding to NET_DEV_EVENT_PHOTOGRAPH_DETECTION_INFO)
EVENT_IVS_REFUELING_GUN_PERSON	0x0000045D	Gun lifting identification action event(corresponding to NET_DEV_EVENT_REFUELING_GUN_PERSON_INFO)
EVENT_IVS_AUDIO_MUTATION	0x0000045E	Audio Mutation event(corresponding to NET_DEV_EVENT_AUDIO_MUTATION_INFO),(note that this event does not support separate use, only as a rule in "Open Intelligent Event (EVENT_IVS_OPEN_INTELLI)")
EVENT_IVS_OBJECT_APPEAR_DETECTION	0x0000045F	Object Appear Detection event(corresponding to NET_DEV_EVENT_OBJECT_APPEAR_DETECTION_INFO)

)
EVENT_IVS_OBJECT_DISAPPEAR_DETECTION	0x00000460	Object Disappear Detection event(corresponding to NET_DEV_EVENT_OBJECT_DISAPPEAR_DETECTION_INFO)
EVENT_IVS_OBJECT_STATE_DETECTION	0x00000461	Object state event(corresponding to NET_DEV_EVENT_OBJECT_STATE_DETECTION_INFO)
EVENT_IVS_TRAPPED_IN_LIFT_DETECTION	0x00000462	Trapped In Lift Detection event(corresponding to NET_DEV_EVENT_TRAPPED_IN_LIFT_DETECTION_INFO)
EVENT_IVS_LIGHT_DETECTION	0x00000463	Video exception event(corresponding to NET_DEV_EVENT_LIGHT_DETECTION_INFO)
EVENT_IVS_COVERING_DETECTION	0x00000464	Covering Detection event(corresponding to NET_DEV_EVENT_COVERING_DETECTION_INFO)
EVENT_IVS_ATM_SMASH_DETECTION	0x00000465	ATM Smash Detection event(corresponding to NET_DEV_EVENT_ATM_SMASH_DETECTION_INFO)
EVENT_IVS_TRAFFIC_ASSISTANT_DRIVER_SMOKING	0x00000466	The co-pilot smoking events(corresponding to NET_DEV_EVENT_TRAFFIC_ASSISTANT_DRIVER_SMOKING_INFO)
EVENT_IVS_TRAFFIC_ASSISTANT_DRIVER_CALLING	0x00000467	The co-pilot call event(corresponding to NET_DEV_EVENT_TRAFFIC_ASSISTANT_DRIVER_CALLING_INFO)
EVENT_IVS_TRAFFIC_VEHICLEIN_AREA	0x00000468	Traffic Vehiclein Area Event(corresponding to NET_DEV_EVENT_TRAFFIC_VEHICLEIN_AREA_INFO)
EVENT_IVS_TRAFFIC_NOT_AFTER_HORIZONTAL_PEOPLE	0x00000469	Disobedience to pedestrians(corresponding to NET_DEV_EVENT_TRAFFIC_NOT_AFTER_HORIZONTAL_PEOPLE_INFO)
EVENT_IVS_WATER_LOGGED_DETECTION	0x0000046A	Water logged Detection Event(corresponding to NET_DEV_EVENT_WATER_LOGGED_DETECTION_INFO)
EVENT_IVS_TRAFFIC_DRIVER_SUN_VISOR	0x0000046B	Traffic Driver Sun Visor Detection Event(corresponding to NET_DEV_EVENT_TRAFFIC_DRIVER_SUN_VISOR_INFO)
EVENT_IVS_WORK_CHECK	0x0000046C	Automatic job search event(corresponding to NET_DEV_EVENT_WORK_CHECK_INFO)
EVENT_IVS_PERSON_PECCANCY	0x0000046D	Person peccancy event(corresponding to NET_DEV_EVENT_PERSON_PECCANCY_INFO)
EVENT_IVS_ACTION_COUNT	0x0000046E	Action Count event(corresponding to NET_DEV_EVENT_ACTION_COUNT_INFO)
EVENT_IVS_WADING_DETECTION	0x0000046F	Wading safety detection and water area monitoring alarm (corresponding to NET_DEV_EVENT_WADING_DETECTION_INFO)
EVENT_IVS_TRAFFIC_GO_STRAIGHT	0x00000470	Traffic go straight(corresponding to NET_DEV_EVENT_TRAFFIC_GO_STRAIGHT_INFO)
EVENT_IVS_PHONE_SECURITY_STICKER_DETECTION	0x00000471	Mobile security sticker detection event(corresponding to

		NET_DEV_EVENT_PHONE_SECURITY_STICKER_DETECTION_INFO)
EVENT_IVS_SAME_OBJECT_SEARCH_DETECT	0x00000472	Trace object detection events (corresponding to NET_DEV_EVENT_SAME_OBJECT_SEARCH_DETECTION_INFO)
EVENT_IVS_TRAFFIC_PEOPLE_IN_MOTOR_ROUTE	0x00000473	Traffic People In Motor Route event(corresponding to NET_DEV_EVENT_TRAFFIC_PEOPLE_IN_MOTOR_ROUTE_INFO)
EVENT_IVS_TRAFFIC_PEOPLE_IN_NON_MOTOR_ROUTE	0x00000474	Traffic People In Non Motor Route event(corresponding to NET_DEV_EVENT_TRAFFIC_PEOPLE_IN_NON_MOTOR_ROUTE_INFO)
EVENT_IVS_ALL_INTELLIGENCE	0x00000475	All intelligence event(Not all smart events are distributed, corresponding to smart video query)
EVENT_IVS_GATE_HEIGHT_DETECTION	0x00000476	Gate height event (corresponding to NET_DEV_EVENT_GATE_HEIGHT_DETECTION_INFO)
EVENT_IVS_TRAFFIC_BRIGHT_LIGHT	0x00000477	Illegal installation of bull eye lights event (corresponding to NET-DEV-EVENT_TRAFFIC.BRIGHTLIGHT_INFO)
EVENT_IVS_FREQUENT_CHANGE_LANE	0x00000478	Frequent Change Lane event(corresponding to NET_DEV_EVENT_FREQUENT_CHANGE_LANE_INFO)
EVENT_IVS_PYROTECHNIC_DETECTION	0x00000479	Pyrotechnic Detection event(corresponding to NET_DEV_EVENT_PYROTECHNIC_DETECTION_INFO)
EVENT_IVS_TRAFFIC_STRANGE_CAR	0x0000047A	Traffic Strange Car event(corresponding to NET_DEV_EVENT_TRAFFIC_STRANGE_CAR_INFO)
EVENT_IVS_SAME_OBJECT_SEARCH_COUNT	0x00000480	Event for counting items based on the chart (corresponding to NET-DEV-EVENT_SAME_OBJECT_SEARCH_COUNT_INFO)
EVENT_IVS_VIDEO_OUT_SNAP	0x00000481	Output port screen capture event (corresponding to NET_DEV_EVENT_IVS_VIDEO_OUT_SNAP_INFO)
EVENT_IVS_POSE_EXCEPTION_NUM_DETECTION	0x00000482	Pose Exception Num Detection event(corresponding to NET_DEV_EVENT_IVS_POSE_EXCEPTION_NUM_DETECTION_INFO)
EVENT_IVS_HOSPITAL_APPRAISE	0x00000483	Medical device report review event (Corresponding to NET_DEV_EVENT_HOSPITAL_APPRAISE_INFO)
EVENT_IVS_GRANARY_TRANS_ACTION_DETECTION	0x00000484	Report changes in grain levels detection event (Corresponding to NET_DEV_EVENT_GRANARY_TRANS_ACTION_DETECTION_INFO)
EVENT_IVS_REGION_PROPORTION_DETECTION	0x00000485	Area proportion detection event (Corresponding to NET_DEV_EVENT_REGION_PROPORTION_DETECTION_INFO)
EVENT_IVS_TRAFFIC_DANGEROUS_SPEED	0x00000486	Speeding event (Corresponding to NET_DEV_EVENT_TRAFFIC_DANGEROUS_SPEED_INFO)

EVENT_IVS_TRAFFIC_URUN_RED_LIGHT	0x00000487	U-turn and run a red light (Corresponding to NET_DEV_EVENT_TRAFFIC_URUN_RED_LIGHT_INFO)
EVENT_IVS_TRAFFIC_LEFT_WRONG_ROUTE	0x00000488	Drive without following the left direction arrow (Corresponding to NET_DEV_EVENT_TRAFFIC_LEFT_WRONG_ROUTE_INFO)
EVENT_IVS_TRAFFIC_RIGHT_WRONG_ROUTE	0x00000489	Drive without following the right direction arrow (Corresponding to NET_DEV_EVENT_TRAFFIC_RIGHT_WRONG_ROUTE_INFO)
EVENT_IVS_TRAFFIC_STRAIGHT_WRONG_ROUTE	0x0000048A	Drive without following the straight direction arrow (Corresponding to NET_DEV_EVENT_TRAFFIC_STRAIGHT_WRONG_ROUTE_INFO)
EVENT_IVS_TRAFFIC_UWRONG_ROUTE	0x0000048B	Drive without following the U-turn arrow (Corresponding to NET_DEV_EVENT_TRAFFIC_UWRONG_ROUTE_INFO)
EVENT_IVS_TRAFFIC_UNQUEUED	0x0000048C	Not waiting in line (Corresponding to NET_DEV_EVENT_TRAFFIC_UNQUEUED_INFO)
EVENT_IVS_POWER_LINE_CHANGE	0x0000048D	Line movement detection through snapshots (Corresponding to NET_DEV_EVENT_POWER_LINE_CHANGE_INFO)
EVENT_IVS_ILLEGAL_ADBOARD_DETECTION	0x0000048E	Illegal billboard detection and alarm (Corresponding to NET_DEV_EVENT_ILLEGAL_ADBOARD_DETECTION_INFO)
EVENT_IVS_GRAIN_HEIGHT_DETECTION	0x0000048F	Grain surface change detection event (Corresponding to NET_DEV_EVENT_GRAIN_HEIGHT_DETECTION_INFO)
EVENT_IVS_OBJECTCHANGE_DETECTION	0x00000490	Change event - Target change detection (Corresponding to NET_DEV_EVENT_OBJECTCHANGE_DETECTION_INFO)
EVENT_IVS_FIRE_AWAY_FROM_MAN_DETECTION	0x00000491	Detection of keeping fire away from people (Corresponding to NET_DEV_EVENT_FIRE_AWAY_FROM_MAN_DETECTION_INFO)
EVENT_IVS_TRUCK_NOT_CLEAN_FOR_PRMA	0x00000492	Unwashed engineering truck detection (Corresponding to NET_DEV_EVENT_TRUCK_NOT_CLEAN_FOR_PRMA_INFO)
EVENT_IVS_ELEVATOR_WORK_INFO	0x00000493	Elevator operation data reporting (Corresponding to NET_DEV_EVENT_ELEVATOR_WORK_INFO_INFO)
EVENT_IVS_ELEVATOR_ALARM	0x00000494	Abnormal elevator alarm (Corresponding to NET_DEV_EVENT_ELEVATOR_ALARM_INFO)
EVENT_IVS_MATERIAL_FLOW_STATUS_DETECTION	0x00000495	Material flow status detection (Corresponding to NET_DEV_EVENT_MATERIAL_FLOW_STATUS_DETECT

		ION_INFO)
EVENT_IVS_RULE_CORRELATION_ALARM	0x00000496	Behavior combination detection (Corresponding to NET_DEV_EVENT_RULE_CORRELATION_ALARM_INFO)
EVENT_IVS_TRAFFIC_OVER_PASSE RSBY_LINE	0x00000497	Block crosswalk (Corresponding to NET_DEV_EVENT_OVER_PASSERSBY_LINE_INFO)
EVENT_IVS_TRAFFIC_OVER_NONE _MOTOR_LINE	0x00000498	Block bicycle crosswalk (Corresponding to NET_DEV_EVENT_OVER_NONE_MOTOR_LINE_INFO)
EVENT_IVS_TRAFFIC_TRUST_CAR	0x00000499	Trusted vehicle event (Corresponding to NET_DEV_EVENT_TRAFFIC_TRUST_CAR_INFO)
EVENT_IVS_TRAFFIC_SUSPICIOUS_CAR	0x0000049A	Suspected vehicle detection (Corresponding to NET_DEV_EVENT_TRAFFIC_SUSPICIOUS_CAR_INFO)
EVENT_IVS_TRAFFIC_RAIN_DETE CTION	0x0000049B	Rainy weather detection (Corresponding to NET_DEV_EVENT_TRAFFIC_RAIN_DETECTION_INFO)
EVENT_IVS_TRAFFIC_SNOW_DETE CTION	0x0000049C	Snowy weather detection (Corresponding to NET_DEV_EVENT_TRAFFIC_SNOW_DETECTION_INFO)
EVENT_IVS_TRAFFIC_FOG_DETECT ION	0x0000049D	Foggy weather detection (Corresponding to NET_DEV_EVENT_TRAFFIC_FOG_DETECTION_INFO)
EVENT_IVS_CROP_DETECTION	0x0000049E	Crop detection (Corresponding to NET_DEV_EVENT_CROP_DETECTION_INFO)
EVENT_IVS_ENCLOSURE_ALARM	0x0000049F	Station exit data (Corresponding to NET_DEV_EVENT_ENCLOSURE_ALARM_INFO)
EVENT_IVS_HEAT_IMAGING_TEMP ER_INFO	0x000004A0	Abnormal temperature alarm for thermal temperature monitoring point (Corresponding to DEV_EVENT_HEAT_IMAGING_TEMPER_INFO_DETAIL)
EVENT_IVS_TRAFFIC_STRAIGHTRU N_REDLIGHT	0x000004A1	Run a red light and go straight (Corresponding to NET_DEV_EVENT_TRAFFIC_STRAIGHTRUN_REDLIGHT_INFO)
EVENT_IVS_TRAFFIC_LEFTRUN_RE DLIGHT	0x000004A2	Run a red light and turn left (Corresponding to NET_DEV_EVENT_TRAFFIC_LEFTRUN_REDLIGHT_INF O)
EVENT_IVS_TRAFFIC_RIGHTRUN_R EDLIGHT	0x000004A3	Run a red light and turn right (Corresponding to NET_DEV_EVENT_TRAFFIC_RIGHTRUN_REDLIGHT_IN FO)
EVENT_IVS_GROUND_THING_DET ECTION	0x000004A4	Recognition of objects on the ground (Corresponding to NET_DEV_EVENT_GROUND_THING_DETECTION_INF O)
EVENT_IVS_PERSON_SPACE_TRAC K	0x000004A5	Movement track of persons (Corresponding to NET_DEV_EVENT_PERSON_SPACE_TRACK_INFO)
EVENT_IVS_AUDIO_SAMPLE	0x000004A6	Timed audio data collection and reporting (Corresponding to NET_DEV_EVENT_AUDIO_SAMPLE_INFO). This event cannot be used alone temporarily. It is a rule in EVENT_IVS_OPEN_INTELLI.
EVENT_IVS_RADARANALYSE_DETE	0x000004A7	Radar linkage event (Corresponding to

CTION		NET_DEV_EVENT_RADARANALYSE_DETECTION_INFO)
EVENT_IVS_LADLE_HANGING_DETECTION	0x000004A8	Lug detection (Corresponding to NET_DEV_EVENT_LADLE_HANGING_DETECTION_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR_CALLING	0x000004A9	Use phone while driving non-motor vehicle (Corresponding to NET_DEV_EVENT_TRAFFIC_NONMOTOR_CALLING_INFO)
EVENT_IVS_TRAFFIC_NONMOTOR_IN_ZEBRA_CROSSING	0x000004AA	Non-motor vehicle blocking crosswalks (Corresponding to NET_DEV_EVENT_TRAFFIC_NONMOTOR_IN_ZEBRA_CROSSING_INFO)
EVENT_IVS_TRAFFIC_TURN_RIGHT_AFTER_PEOPLE_SUCCESS	0x000004AB	Yield to passengers going straight when turning right (Corresponding to NET_DEV_EVENT_TRAFFIC_TURN_RIGHT_AFTER_PEOPLE_SUCCESS_INFO)
EVENT_IVS_PSR_STAND_RISE_DETECTION	0x000004AC	Criminal stand-up detection (Corresponding to NET_DEV_EVENT_PSR_STAND_RISE_DETECTION_INFO)
EVENT_IVS_PSR_KEY_PEO_RISE_DETECTION	0x000004AD	Key person stand-up detection (Corresponding to NET_DEV_EVENT_PSR_KEY_PEO_RISE_DETECTION_INFO)
EVENT_IVS_NATURAL_DISASTER_DETECTION	0x000004AE	Natural disaster detection (Corresponding to NET_DEV_EVENT_NATURAL_DISASTER_DETECTION_INFO)
EVENT_IVS_PRE_HEAT_IMAGING_TEMPER	0x000004AF	Abnormal temperature pre-alarm for thermal temperature monitoring point (Corresponding to structure DEV_EVENT_PRE_HEAT_IMAGING_TEMPER_INFO)
EVENT_IVS_ILLEGAL_CYCLING_DETECTION	0x000004B0	Illegal riding detection (Corresponding to structure NET_DEV_EVENT_ILLEGAL_CYCLING_DETECTION_INFO)
EVENT_IVS_TRAFFIC_INTERSECTION_CONGESTION	0x000004B1	Intersection congestion detection (Corresponding to structure NET_DEV_EVENT_TRAFFIC_INTERSECTION_CONGESTION_INFO)
EVENT_IVS_MOSAIC	0x000004B2	Privacy masking (Corresponding to structure NET_DEV_EVENT_MOSAIC_INFO)
EVENT_IVS_FLAME_EXTINGUISH	0x000004B3	Flame extinguishing event (Corresponding to structure NET_DEV_EVENT_FLAME_EXTINGUISH_INFO)
EVENT_IVS_FLAME_OVER_SIZE	0x000004B4	Flame too high (Corresponding to structure NET_DEV_EVENT_FLAME_OVER_SIZE_INFO)
EVENT_IVS_FLAME_UNDER_SIZE	0x000004B5	Flame too low (Corresponding to structure NET_DEV_EVENT_FLAME_UNDER_SIZE_INFO)
EVENT_IVS_ATTENDANCE_RATE_D	0x000004B6	Attendance rate detection (Corresponding to

ETECTION		structure NET_DEV_EVENT_ATTENDANCE_RATE_DETECTION_I NFO)
EVENT_IVS_SEATING_RATE_DETECT TION	0x000004B7	Seat occupancy rate detection (Corresponding to structure NET_DEV_EVENT_SEATING_RATE_DETECTION_INFO)
EVENT_IVS_AUDIO_ANALYSIS	0x000004B8	Audio keyword recognition and alarm (Corresponding to structure NET_DEV_EVENT_AUDIO_ANALYSIS_INFO)
EVENT_IVS_STEAL_OIL	0x000004B9	Low fuel alarm (Corresponding to structure NET_DEV_EVENT_STEAL_OIL_INFO)
EVENT_IVS_VISION_LANG_MODEL _DETECTION	0x000004BA	Visual text model detection (Corresponding to structure NET_DEV_EVENT_VISION_LANG_MODEL_DETECTION _INFO)
EVENT_IVS_BETWEEN_RULES_TEM PERDIFF_ALARM	0x000004BB	Temperature difference alarm (NET_DEV_EVENT_BETWEEN_RULES_TEMPERDIFF_AL ARM_INFO)
EVENT_IVS_GAS_CLOUD_DETECTI ON	0x000004BC	Gas detection (NET_DEV_EVENT_GAS_CLOUD_DETECTION_INFO)
EVENT_IVS_INFUSION_DETECTION	0x000004BD	Infusion detection (Corresponding to structure NET_DEV_EVENT_INFUSION_DETECTION_INFO)
EVENT_IVS_PERSONNEL_CATEGOR Y_COUNT	0x000004BE	Person type statistics (Corresponding to structure NET_DEV_EVENT_PERSONNEL_CATEGORY_COUNT_I NFO)
EVENT_IVS_SMART_MOTION_EQUI PMEN	0x000004BF	SMD event (Corresponding to structure NET_DEV_EVENT_SMART_MOTION_EQUIPMENT_INF O)
EVENT_IVS_TARFFIC_NOT_IN_ALL OW_LIST	0x000004C0	Not on allowlist (Corresponding to structure NET_DEV_EVENT_TARFFIC_NOT_IN_ALLOW_LIST_INF O)
EVENT_IVS_STEREO_TAIL_DETECTI ON	0x000004C1	ATM booth tailing detection (Only add to the enumerated values and not implemented in CLIENT_RealLoadPictureEx)
EVENT_IVS_TARGET_PROPORTION	0x000004C2	Semantic segmentation event (Corresponding to structure NET_DEV_EVENT_TARGET_PROPORTION_INFO)
EVENT_IVS_OPEN_ALG_ARRANGE	0x000004C3	Open algorithm orchestration (Corresponding to structure NET_DEV_EVENT_OPEN_ALG_ARRANGE_INFO)
EVENT_IVS_AMW_PERSON_DETECT TION	0x000004C4	Human detection through millimeter wave walk-through metal detector. Serves as an event enumeration. Event implementation is not currently being carried out, and there is no corresponding structure available.
EVENT_IVS_SIGNAL_LIGHT_SKEW_ DETECTION	0x000004C5	Traffic light offset detection (Corresponding to structure

		NET_DEV_EVENT_SIGNAL_LIGHT_SKEW_DETECTION_INFO)
EVENT_IVS_PUBLIC_SECURITY_AB NORMAL	0x000004C6	Public safety abnormal behaviors detection (Corresponding to structure NET_DEV_EVENT_PUBLIC_SECURITY_ABNORMAL_INFO)
EVENT_IVS_PATROL_TASK_FAILURE	0x000004C7	Error sending inspection tasks (Corresponding to structure NET_DEV_EVENT_PATROL_TASK_FAILURE_INFO)
EVENT_IVS_TRAFFIC_FACILITIES_AB NORMAL	0x000004C8	Abnormal traffic safety facility detection (Corresponding to structure NET_DEV_EVENT_TRAFFIC_FACILITIES_ABNORMAL_INFO)
EVENT_HY_FIRE_DETECTION	0x01000001	Fire flame detection events (corresponding to DEV_EVENT_HY_FIRE_DETECTION_INFO)
EVENT_HY_SMOG_DETECTION	0x01000002	Fire smoke detection events (corresponding to DEV_EVENT_HY_SMOG_DETECTION_INFO)
EVENT_HY_INFIRE_PASSAGE_DETECTION	0x01000003	Fire evacuation passage blocking event (corresponding to DEV_EVENT_HY_INFIRE_PASSAGE_DETECTION_INFO)
EVENT_HY_OUTFIRE_PASSAGE_DETECTION	0x01000004	Occupation of fire-fighting vehicle passageway/occupation of fire-fighting climbing surface (corresponding to DEV_EVENT_HY_OUTFIRE_PASSAGE_DETECTION_INFO)
EVENT_HY_MAN_LEAVING_DETECTION	0x01000005	Personnel leaving the fire control room (corresponding to DEV_EVENT_HY_MAN_LEAVING_DETECTION_INFO)