

请输入标题

通用信息：

基础URL：/cgi-bin/entry.cgi

URL结构：/cgi-bin/entry.cgi/{api_category}/{resource}/{sub_resource}

认证方式: JWT Token (通过Cookie传递)

内容类型: application/json

错误响应：

当后端获取信息出错或者其他原因无法给出请求信息时按照以下格式给出响应

```
1  {  
2  "status": 0,      状态码，供用户排查  
3  "message": "error description", 出错时前端显示  
4  }
```

交互数据格式：

json数据KEY的命名：首字母代表VALUE数据类型，i代表整型数据，f代表浮点，s代表字符串，l代表列表，d代表字典，KEY采用驼峰命名

认证

1. 登录

URL:/system/system/login

请求方法: POST

响应格式: JSON

PUT示例

```
1  {
2    "sUserName": "admin",
3    "sPassword": "password"  sha256
4  }
```

响应示例:

```
1  {
2    "iStatus": 0,  Int 登录状态
3    "iAuth": 0,   Int 认证等级
4  }
```

认证Headers

```
1  Cookie: token={jwt_token}
```

一、 设备信息

1.基本信息：

URL：/system/device-info

请求方法：GET

响应格式：JSON

```
1  {  
2    "sSerialNumber": "RC1126B-20240101-001",    String 设备序列号  
3    "sFirmwareVersion": "V1.0.3_20250915", String 固件版本  
4    "sSensorModel": "SC850SL",                String sensor板型号  
5    "sBasePlateModel": "PoE",                  String 底板型号  
6  }
```

2.系统时间：

1) 功能：获取系统时间

URL: /system/time

请求方法：GET

响应格式: JSON

响应示例:

```
1  {
2    "iTimestamp": 13256456,   Int64 时间戳
3    "sTimezone": "Asia/Shanghai",   String 时区城市
4    "sTz": "UTC+8",           String 时区
5  }
6
```

2) 功能：更新系统时间

URL：/system/time

请求方法：PUT

响应格式：JSON

PUT示例：

```
1  {
2    "sMethod": "ntp/manual",     时区设置方式
3    "sTimezone": "Asia/Shanghai",  时区，当method字段为ntp时存在
4    "sTz": "UTC+8",             时区缩写当method字段为ntp时存在
5    "iTimestamp": 1699999999     时间戳，当method字段为manual时使用
6  }
```

响应示例：

```
1  {
2  "iTimestamp": 13214645643,(long int),    Int64 时间戳
3  "sTimezone": "Asia/Shanghai",    String 时间城市
4  "sTz": "UTC+8"                String 时区
5  }
```

3.系统状态

功能：获取CPU NPU占用率

URL：/system/resource-info

请求方法：GET

响应示例：

```
1  {
2  "iCpuUsage": 13, Int CPU使用率 范围0-100
3  "iNpuUsage": 20, Int NPU使用率 范围0-100
4  "iMemUsage": 30, Int NPU使用率 范围0-100
5  "iStorageUsage": 50 Int STORAGE使用率 范围0-100
6
7  }
```

备注：频繁请求，

4. 网络信息

功能：获取网络信息

URL：</network/wlan>

请求方法：GET

响应示例：

```
1  {
2    "dIpv4": {
3      "sV4Address": "192.168.1.100",  IPv4地址
4      "sV4Gateway": "192.168.1.1",    ipv4网关地址
5      "sV4Method": "manual",          ip设置方式
6      "sV4Netmask": "255.255.255.0"   ipv4子网掩码
7    },
8    "dLink": {
9      "sDNS1": "8.8.8.8",              DNS地址
10     "sDNS2": "8.8.4.4",              DNS地址
11     "sAddress": "aa:bb:cc:dd:ee:ff",  MAC地址
12     "sInterface": "eth0",            网卡设备
13     "iPower": 1,
14     "sNicSpeed": "100baseT/Full"     网络速度
15   }
16 }
17
```

功能：更新网络信息

URL：/network/wlan

请求方法：PUT

PUT示例：

```
1  {  
2    "sGetMethod": "DHCP", String IP获取方法 ["DHCP","STATIC"]  
3    "sIpAddress": "192.168.1.11", String IP地址，getmethod为DHCP时有值  
4    "sIpGateway": "192.168.1.1", String 网关地址  
5    "sIpNetmask": "255.255.255.0", String 子网掩码  
6    "sDNS0": "8.8.8.8", String DNS服务器地址  
7    "sDNS1": "1.1.1.1" String 备用DNS服务器地址  
8  }
```

5. WiFi

功能：查询WiFi状态

URL：/network/wifi

请求方法：GET

响应示例：

```
1  {  
2    "iPower": 1,  Int 是否处于开启状态  
3    "iId": 1,    Int  
4    "sType": "wifi" String 类型  
5  }
```

功能：获取WiFi信息

URL：/network/wifi-list

请求方法：GET

响应示例：


```
1  [
2    {
3      "sBssid": "9a:bb:99:12:1b:61",
4      "sFlags": "[WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]",
5      "iFrequency": 2412,
6      "iRssi": -21,
7      "sSsid": "Freeland2"
8    },
9    {
10     "sBssid": "f0:b4:29:53:e4:15",
11     "sFlags": "[WPA2-PSK-CCMP][ESS]",
12     "iFrequency": 2417,
13     "iRssi": -23,
14     "sSsid": "Xiaomi_Youth"
15   },
16   {
17     "sBssid": "70:4d:7b:95:47:c0",
18     "sFlags": "[WPA2-PSK-CCMP][ESS]",
19     "iFrequency": 2462,
20     "iRssi": -35,
21     "sSsid": "asus_test"
```

功能：扫描WiFi

URL：/network/wifi

请求方法：POST

Param：

参数名	值	备注
scan	3	表示返回前多少个WiFi信息

响应示例：

```
1  [
2    {
3      "sBssid": "9a:bb:99:12:1b:61",
4      "sFlags": "[WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]",
5      "iFrequency": 2412,
6      "iRssi": -21,
7      "sSsid": "Freeland2"
8    },
9    {
10     "sBssid": "f0:b4:29:53:e4:15",
11     "sFlags": "[WPA2-PSK-CCMP][ESS]",
12     "iFrequency": 2417,
13     "iRssi": -23,
14     "sSsid": "Xiaomi_Youth"
15   },
16   {
17     "sBssid": "70:4d:7b:95:47:c0",
18     "sFlags": "[WPA2-PSK-CCMP][ESS]",
19     "iFrequency": 2462,
20     "iRssi": -35,
21     "sSsid": "asus_test"
```

功能：WiFi网络连接/断开

URL：/network/wifi

请求方法：DELETE

Param：

参数名	值	备注
service	ssid	WiFi ssid

功能：WiFi电源控制

URL：/network/wifi

请求方法：POST

Param：

参数名	值	备注
power	on/off	on为开启电源，off为关闭电源

6. 组播

功能：获取当前组播配置

URL:/network/multicast

请求方法：GET

响应实例：

```
1  {  
2  "muticastAddress": "224.1.1.1",  String 组播地址  
3  "muticastPort": "8868"          String 组播端口  
4  }  
5
```

功能：组播设置

URL：/network/multicast

请求方法：POST

请求示例：

```
1  {  
2  "muticastAddress": "224.1.1.1",  String 组播地址  
3  "muticastPort": "8868"          String 组播端口  
4  }
```

7. HTTP

功能：获取当前HTTP API设置

URL：/web/setting

请求方法：GET

响应示例：

```
1  {  
2  "sApiPort": "8080",      String API端口  
3  "sApiKey": "adfsfgfvxcvb" String 访问密钥  
4  }
```

功能：HTTP API设置

URL：/web/setting

请求方法：POST

请求示例：

```
1  {  
2  "sApiPort": "8080",      String API端口  
3  "sApiKey": "adfsfgfvxcvb" String 访问密钥  
4  }
```

8. FTP服务

功能：获取当前FTP服务设置

URL：/ftp/setting

请求方法：POST

请求示例：

```
1  {
2  "sFtpPort": "1234",      String 端口号
3  "sFtpUser": "username",  String 用户名
4  "sFtpPassword": "adfafadfa" String SHA256哈希后的值
5  }
```

功能：FTP服务设置

URL：/ftp/setting

请求方法：POST

请求示例：

```
1  {
2  "sFtpPort": "1234",      String 端口号
3  "sFtpUser": "username",  String 用户名
4  "sFtpPassword": "adfafadfa" String SHA256哈希后的值
5  }
```

9. 配置管理

功能：导出配置

URL：/config/export

请求方法：GET

Content-Type: application/json

功能：导入配置

URL：/config/upload

请求方法：POST

Content-Type: application/json

10. 固件管理

功能：固件更新上传

URL：/system/firmware-upgrade

请求方法：POST

Param:

参数名	值	备注
upload-type	resumable/network	resumable为从本地上传附件network为从网络获取更新固件

upload-type = network

响应示例:


```
1  {
2      "iUpdateAvailable": 1,    Int 是否可以更新
3      "sCurrentVersion": "V1.0.0",  String 当前固件版本
4      "sLatestVersion": "V1.1.0",  String 最新固件版本
5      "sToken": "xxxxxxxxx"  String 短期验证令牌包
6  }
7
```

功能: 确认远程更新

URL: /system/firmware-network

请求方法 : POST

Content-Type: application/json

```
1  {
2      "sToken": "xxxxxxxxx",  String 验证令牌包
3      "sStart": "on/off"  String 是否更新
4  }
```

upload-type = resumable(前端二次确认)

Content-Type: multipart/form-data

11. 重启设备

功能：重启设备

URL：</system/reboot>

请求方法：POST

响应:无

12. 恢复出厂设置

功能：恢复出厂设置

URL：</system/factory-reset>

请求方法：POST

响应：无

13. 修改密码

功能：修改密码

URL:/system/password

请求方法：PUT

请求示例：

```
1  {  
2  "sOldPassword":"adfafgfga",  String 旧密码SHA256哈希值  
3  "sNewPassword":"adfadfzcv"   String 新密码SHA256哈希值  
4  }
```

二、实时画面

1. 视频设置

URL结构 : /video/{stream_id}/{specific}

stream_id

- 0：主码流
- 1：子码流

specific 分类

URL	说明
/video/0/encode	主码流视频编码配置
/video/0/osd-char	主码流视频osd 字符配置，设备名、通道，时间配置
/video/0/osd-inference	主码流视频osd 推理结果显示配置
/video/0/osd-mask	主码流视频osd 遮盖配置
/video/0/stream	主码流视频推流配置

1.1 视频编码

功能：获取视频流编码配置

URL：/video/{stream_id}/encode

请求方法：GET

响应示例：

```
1  {
2    "id": 0, // 0=主码流, 1=子码流
3    "sStreamType": "mainStream", //, 码流类型/逻辑名称 ( mainStream / subStream)
4    "sResolution": "3840*2160", //编码输出分辨率 ( 宽*高 )
5    "sOutputDataType": "H.265", // 编码输出类型、编码标准, H.264、H.265
6    "sFrameRate": "30", //帧率
7    "iMaxRate": 8192, // 最大码率 ( kbps )
8    "iGOP": 60, // GOP(Group of Pictues)长度, 即 I 帧间隔,
9    "sRCMode": "VBR", //Rate Control Mode , 码率控制模式 ; CBR: 固定码率, VBR: 可变码率
10   "sRCQuality": "highest", // 可变码率时 , 质量等级 highest / high / medium / low
11
12   //----- 以下配置预留-----
13   "iMinRate": 0, // 预留 , 最小码率(kbps)
14   "iStreamSmooth": 50, // 预留, 码流平滑度参数
15   "iTargetRate": 0, //预留, 目标码率 ( kbps )
16   "sFrameRateIn": "30", //预留, : 输入帧率
17   "sGOPMode": "normalP", //预留, GOP 模式
18   "sH264Profile": "high", //预留, H.264 Profile , 编码复杂度: baseline/main/high
19   "sSmart": "open", //预留, 智能码率/智能编码开关 ( open/close )
20 }
```

功能：配置视频流编码参数

URL：/video/{stream_id}/encode

请求方法：PUT

配置内容：

```
1  {
2      "id": 0, // 0=主码流, 1=子码流
3      "sStreamType": "mainStream", //, 码流类型/逻辑名称 ( mainStream / subStream)
4      "sResolution": "3840*2160", //编码输出分辨率 ( 宽*高 )
5      "sOutputDataType": "H.265", // 编码输出类型、编码标准, H.264、H.265
6      "sFrameRate": "30", //帧率
7      "iMaxRate": 8192, // 最大码率 ( kbps )
8      "iGOP": 60, // GOP(Group of Poctures)长度, 即 I 帧间隔,
9      "sRCMode": "VBR", //Rate Control Mode , 码率控制模式 ; CBR: 固定码率, VBR: 可变码率
10     "sRCQuality": "highest", // 可变码率时 , 质量等级  highest / high / medium / low
11 }
12
```

1.2 OSD设置

功能：获取视频流OSD 绘制参数

URL：/video/{stream_id}/osd-char

请求方法：GET

响应示例：

```
1  {
2    "attribute": {
3      "iOSDFontSize": 64, // 字体大小，取值:0, 16, 32, 48, 64, 其中0表示自适应
4      "sOSDFrontColor": "fff799", // 字体颜色，格式为RGB的十六进制表示法，例如：红色为"ff0000"
5      "sOSDFrontColorMode": "customize" // 字体颜色模式, 0=黑白自动, 1=自定义
6    },
7    "channelNameOverlay": {
8      "iEnabled": 1, // 0=关闭, 1=开启
9      "iPositionX": 528, // 字体位置X坐标，单位为像素
10     "iPositionY": 458, // 字体位置Y坐标，单位为像素
11     "sChannelName": "reCamera 1126B" // 通道名称
12   },
13   "dateTimeOverlay": {
14     "iEnabled": 1, // 0=关闭, 1=开启
15     "iDisplayWeekEnabled": 1, // 0=不显示星期, 1=显示星期
16     "iPositionX": 50, // 字体位置X坐标，单位为像素
17     "iPositionY": 244, // 字体位置Y坐标，单位为像素
18     "sDateStyle": "CHR-YYYY-MM-DD", // 日期格式, "CHR-YYYY-MM-DD", "CHR-DD-MM-YYYY", "CHR-MM-DD-YYYY",
19     "NUM-YYYY-MM-DD", "NUM-DD-MM-YYYY", "NUM-MM-DD-YYYY"
20     "sTimeStyle": "24hour" // 时间格式, "12hour", "24hour"
21   },
22 }
```


功能：配置OSD 绘制参数

URL：/video/{stream_id}/osd-char

请求方法：PUT

数据内容举例，全部配置

```
1  {
2    "attribute": {
3      "iOSDFontSize": 64, // 字体大小，取值:0, 16, 32, 48, 64, 其中0表示自适应
4      "sOSDFrontColor": "fff799", // 字体颜色，格式为RGB的十六进制表示法，例如：红色为"ff0000"
5      "sOSDFrontColorMode": "customize" // 字体颜色模式, 0=黑白自动, 1=自定义
6    },
7    "channelNameOverlay": {
8      "iEnabled": 1, // 0=关闭, 1=开启
9      "iPositionX": 528, // 字体位置X坐标，单位为像素
10     "iPositionY": 458, // 字体位置Y坐标，单位为像素
11     "sChannelName": "reCamera 1126B" // 通道名称
12   },
13   "dateTimeOverlay": {
14     "iEnabled": 1, // 0=关闭, 1=开启
15     "iDisplayWeekEnabled": 1, // 0=不显示星期, 1=显示星期
16     "iPositionX": 50, // 字体位置X坐标，单位为像素
17     "iPositionY": 244, // 字体位置Y坐标，单位为像素
18     "sDateStyle": "CHR-YYYY-MM-DD", // 日期格式, "CHR-YYYY-MM-DD", "CHR-DD-MM-YYYY", "CHR-MM-DD-YYYY",
19     "NUM-YYYY-MM-DD", "NUM-DD-MM-YYYY", "NUM-MM-DD-YYYY"
20     "sTimeStyle": "24hour" // 时间格式, "12hour", "24hour"
21   },
22 }
```

数据内容举例，部分修改

```
1  {  
2    "SNOOverlay": { // 设备读取自己的SN信息  
3      "iEnabled": 1, // 0=关闭, 1=开启  
4      "iPositionX": 50, // 字体位置X坐标，单位为像素  
5      "iPositionY": 244 // 字体位置Y坐标，单位为像素  
6    }  
7  }
```

备注说明:

1. 时间显示上，对应显示星期几是用中文还是英文，经过沟通之后，统一为英文
- 2.

1.3 AI 结果

功能：获取视频流AI 结果绘制配置

URL：/video/{stream_id}/osd-inference

请求方法：GET

响应示例：

```
1  {
2    "inferenceOverlay": {
3      "iEnabled": 1, // 0=关闭, 1=开启
4    },
5  }
```

功能：配置视频流AI 结果绘制

URL：/video/{stream_id}/osd-inference

请求方法：POST

数据内容：

```
1  {
2    "inferenceOverlay": {
3      "iEnabled": 1, // 0=关闭, 1=开启
4    },
5  }
```

1.4 遮盖

功能：获取视频流遮盖绘制配置

URL：/video/{stream_id}/osd-mask

请求方法：GET

响应示例：

```
1  {
2    "iEnabled": 1, // 0=关闭, 1=开启
3    "normalizedScreenSize": {
4      "iNormalizedScreenHeight": 608, // web显示高度, 单位为像素
5      "iNormalizedScreenWidth": 1080 // web显示宽度, 单位为像素
6    },
7    "privacyMask": [
8      {
9        "id": 0,      // 遮挡区域ID
10       "iMaskHeight": 77, // 遮挡区域高度, 单位为像素
11       "iMaskWidth": 213, // 遮挡区域宽度, 单位为像素
12       "iPositionX": 53, // 遮挡区域左上角X坐标, 单位为像素
13       "iPositionY": 380 // 遮挡区域左上角Y坐标, 单位为像素
14     },
15     {
16       "id": 1,
17       "iMaskHeight": 0,
18       "iMaskWidth": 0,
19       "iPositionX": 0,
20       "iPositionY": 0
21     }
```

功能：配置视频流遮盖绘制

URL：/video/{stream_id}/osd-mask

请求方法：POST

数据内容：

```
1  {
2    "iEnabled": 1, // 0=关闭, 1=开启
3    "normalizedScreenSize": {
4      "iNormalizedScreenHeight": 608, // web显示高度, 单位为像素
5      "iNormalizedScreenWidth": 1080 // web显示宽度, 单位为像素
6    },
7    "privacyMask": [
8      {
9        "id": 0,      // 遮挡区域ID
10       "iMaskHeight": 77, // 遮挡区域高度, 单位为像素
11       "iMaskWidth": 213, // 遮挡区域宽度, 单位为像素
12       "iPositionX": 53, // 遮挡区域左上角X坐标, 单位为像素
13       "iPositionY": 380, // 遮挡区域左上角Y坐标, 单位为像素
14     }
15   ]
16 }
```

1.5 推流

功能：获取视频流推流配置

URL : /video/{stream_id}/stream

请求方法 : GET

响应示例 :

```
1  {
2      //推流协议rtsp/rtmp/onvif
3      "streamProtocol": "rtsp",
4      //rtsp
5      "rtsp": {
6          "iPort": 8554
7      },
8      //rtmp
9      "rtmp": {
10         "sURL": "rtmp://xxx", //推流地址
11         "iAuthType": 0, //0-秘钥 1-用户密码
12         "sSecretKey": "xxxx", //秘钥
13         "sUserName": "admin",
14         "sPassword": "admin"
15     },
16     //onvif
17     "onvif": {
18         "sUserName": "admin",
19         "sPassword": "admin"
20     }
21 }
```

功能：配置视频流推流

URL：/video/{stream_id}/stream

请求方法：POST

数据内容：

```
1  {
2      //推流协议rtsp/rtmp/onvif
3      "streamProtocol": "rtsp",
4      //rtsp
5      "rtsp": {
6          "iPort": 8554
7      },
8      //rtmp
9      "rtmp": {
10         "sURL": "rtmp://xxx", //推流地址
11         "iAuthType": 0, //0-秘钥 1-用户密码
12         "sSecretKey": "xxxx", //秘钥
13         "sUserName": "admin",
14         "sPassword": "admin"
15     },
16     //onvif
17     "onvif": {
18         "sUserName": "admin",
19         "sPassword": "admin"
20     }
21 }
```

2. 音频设置

功能：获取音频编码相关信息，URL 中的id 0: 主码流的音频，1:子码流的音频

URL：/audio/<id>

请求方法：GET

响应示例：

```
1  {  
2      ""  
3      "iEnable":1, //Int32 使能音频编码  
4      "iBitRate": 32000, //Int32 码率BPS  
5      "sEncodeType": "G711A" // String 编码格式，G711A, G711U, OPUS  
6  }
```

功能：配置音频，URL 中的id 0: 主码流的音频，1:子码流的音频

URL：/audio/<id>

请求方法：GET

响应示例：

```
1  {  
2    "iEnable":1, //Int32 使能音频编码  
3    "iBitRate": 32000, //Int32 码率BPS  
4    "sEncodeType": "G711A" // String 编码格式 , 可选有:G711A, G711U, OPUS  
5  }  
6
```

TODO: 编码格式需要哪几种确定

附录: 支持的音频编码格式

```
1  RK_AUDIO_ID_PCM_ALAW,  /** <g711a> */
2  RK_AUDIO_ID_PCM_MULAW, /** <g711u> */
3  RK_AUDIO_ID_PCM_S16LE, /**< Any variant of PCM_S16LE coding */
4  RK_AUDIO_ID_PCM_S24LE, /**< Any variant of PCM_S24LE coding */
5  RK_AUDIO_ID_PCM_S32LE, /**< Any variant of PCM_S32LE coding */
6  RK_AUDIO_ID_ADPCM_G722,  /**< Any variant of ADPCM_G722 encoded data */
7  RK_AUDIO_ID_ADPCM_G726,  /**< Any variant of ADPCM_G726 encoded data */
8  RK_AUDIO_ID_ADPCM_G726LE,  /**< G.726 ADPCM little-endian encoded data*/
9  RK_AUDIO_ID_ADPCM_IMA_QT,  /**< Any variant of ADPCM_IMA encoded data */
10 RK_AUDIO_ID_AMR_NB,  /**< Any variant of AMR_NB encoded data */
11 RK_AUDIO_ID_AMR_WB,  /**< Any variant of AMR_WB encoded data */
12 RK_AUDIO_ID_GSMFR,  /**< Any variant of GSM fullrate (i.e. GSM610) */
13 RK_AUDIO_ID_GSMEFR,  /**< Any variant of GSM Enhanced Fullrate encoded data*/
14 RK_AUDIO_ID_GSMHR,  /**< Any variant of GSM Halfrate encoded data */
15 RK_AUDIO_ID_PDCFR,  /**< Any variant of PDC Fullrate encoded data */
16 RK_AUDIO_ID_PDCEFR,  /**< Any variant of PDC Enhanced Fullrate encoded data */
17 RK_AUDIO_ID_PDCHR,  /**< Any variant of PDC Halfrate encoded data */
18 RK_AUDIO_ID_TDMAFR,  /**< Any variant of TDMA Fullrate encoded data (TIA/EIA-136-420) */
19 RK_AUDIO_ID_TDMAEFR,  /**< Any variant of TDMA Enhanced Fullrate encoded data (TIA/EIA-136-410) */
20 RK_AUDIO_ID_QCELP8,  /**< Any variant of QCELP 8kbps encoded data */
21 RK_AUDIO_ID_QCELP13,  /**< Any variant of QCELP 13kbps encoded data */
```

3. 图像设置

URL结构 : /image/{cam_id}/{scene_id}/{specific}

cam_id: 摄像头ID , 当前固定为0

scene_id: 场景ID, 0=通用, 1=白天, 2=晚上

specific : adjustment/exposure/white-blance/blc/enhancement

URL	说明
/image/0	当前画面的所有配置 (PUT 请求配置保存)
/image/0/video-adjustment	视频流显示调整, 镜像, 旋转 (PUT 请求配置不保存)
/image/0/{scene_id}/adjustment	图像基础调节: 亮度, 对比度, 饱和度, 锐度, 色调 (PUT 请求配置不保存)
/image/0/{scene_id}/exposure	图像曝光调节: 曝光时间等等 (PUT 请求配置不保存)
/image/0/{scene_id}/white-blance	图像白平衡调节: (PUT 请求配置不保存)
/image/0/{scene_id}/blc	背光 (PUT 请求配置不保存)
/image/0/{scene_id}/night-to-day	日夜参数 (PUT 请求配置不保存)
/image/0/{scene_id}/enhancement	图像增强 (PUT 请求配置不保存)
/image/0/{scene_id}/af	对焦 (PUT 请求配置不保存)

注意事项:

1. 当用户的配置为自动转换时，此时又处于web 画面编辑状态，这种自动转换时刻的边界点先不考虑

3.0 视频画面所有参数

功能：获取当前画面ISP 参数

URL:/image/0

请求方法：GET

请求示例：

```
1  {
2    "id": 0, // 摄像头通道ID ( 0 表示第一个摄像头 )
3    "videoAdjustment": { // 图像方向与防频闪
4      "iImageRotation": 90, // 旋转角度 : 0/90/180/270 , 用于传感器安装方向
5      "sImageFlip": "close", // 镜像翻转 : close=无 ; mirror=水平 ; flip=垂直 ; centrosymmetric=水平+垂直
6      "sPowerLineFrequencyMode": "NTSC(60HZ)" // 防闪频率 : PAL(50HZ)=50Hz地区 ; NTSC(60HZ)=60Hz;
7    },
8    "nightToDay": { // 昼夜切换与补光
9      "iMode": 0, // 转换模式, 0=自动转换, 1=定时转换, 2=保持不变
10     "iNightToDayFilterLevel": 5, // 切换阈值灵敏度,自动转换模式使用,0=高 , 1=中 , 2=低
11     "iNightToDayFilterTime": 5, // 切换滞后时间(s) 防抖,自动转换模式使用,1~60s
12     "iDawnTime": 28800, // 晨曦时间(秒),定时转换模式使用,0~86400s
13     "iDuskTime": 64800, // 黄昏时间(秒),需要大于iDawnTime时间 , 定时转换模式使用,0~86400s
14     "iProfileSelect": 0 // 配置文件选择, 保持不变模式使用, 0=通用, 1=白天, 2=夜晚
15     "iProfileCur": 0 // 当前使用的profile配置
16   },
17   "profile": [ // 0=通用, 1=白天, 2=夜晚
18     {
19       "imageAdjustment": { // 基础后处理调节
20         "iBrightness": 57, // 亮度 0-100 -> 内部 *2.55 到 0-255 ; 过高泛灰
21         "iContrast": 50, // 对比度 0-100 -> *2.55 ; 过高丢失中间层次
```

功能：修改当前画面参数

URL:/image/0

请求方法：POST

请求示例：

```
1  {
2    "id": 0, // 摄像头通道ID ( 0 表示第一个摄像头 )
3    "videoAdjustment": { // 图像方向与防频闪
4      "iImageRotation": 90, // 旋转角度 : 0/90/180/270 , 用于传感器安装方向
5      "sImageFlip": "close", // 镜像翻转 : close=无 ; mirror=水平 ; flip=垂直 ; centrosymmetric=水平+垂直
6      "sPowerLineFrequencyMode": "NTSC(60HZ)" // 防闪频率 : PAL(50HZ)=50Hz地区 ; NTSC(60HZ)=60Hz;
7    },
8    "nightToDay": { // 昼夜切换与补光
9      "iMode": 0, // 转换模式, 0=自动转换, 1=定时转换, 2=保持不变
10     "iNightToDayFilterLevel": 5, // 切换阈值灵敏度,自动转换模式使用,0=高 , 1=中 , 2=低
11     "iNightToDayFilterTime": 5, // 切换滞后时间(s) 防抖,自动转换模式使用,1~60s
12     "iDawnTime": 28800, // 晨曦时间(秒),定时转换模式使用,0~86400s
13     "iDuskTime": 64800, // 黄昏时间(秒),需要大于iDawnTime时间 , 定时转换模式使用,0~86400s
14     "iProfileSelect": 0 // 配置文件选择, 保持不变模式使用, 0=通用, 1=白天, 2=夜晚
15   },
16   "profile": [//0=通用, 1=白天, 2=夜晚
17     {
18       "imageAdjustment": { // 基础后处理调节
19         "iBrightness": 57, // 亮度 0-100 -> 内部 *2.55 到 0-255 ; 过高泛灰
20         "iContrast": 50, // 对比度 0-100 -> *2.55 ; 过高丢失中间层次
21         "iHue": 50, // 色调 0-100 -> *2.55 ; 通常保持 50
```

3.1 切换场景

URL:/image/0/scene

请求方法：PUT

请求示例：

```
1  {  
2    "iProfileCur": 0 // 当前使用的profile配置  
3  }
```

3.2 视频画面调节

URL:/image/0/video-adjustment

请求方法：PUT

请求示例：

```
1  {
2      "iImageRotation": 90, //旋转, 0, 90, 180, 270
3      "sImageFlip": "close", //镜像/翻转方式, close, mirror ( 水平 ) , flip ( 上下 ) , centrosymmetric (水平+垂直都翻)
4      "sPowerLineFrequencyMode": "NTSC(60HZ)" //// "市电频率防闪模式" / "防频闪标准" , PAL(50HZ)\NTSC(60HZ)
5  }
```

3.3 日夜参数

URL:/image/0/{scene_id}/night-to-day

请求方法：PUT

请求示例：

```
1  {
2      "iMode":0, // 转换模式, 0=自动转换, 1=定时转换, 2=保持不变
3      "iNightToDayFilterLevel": 5, // 切换阈值灵敏度,自动转换模式使用,0=高, 1=中, 2=低
4      "iNightToDayFilterTime": 5, // 切换滞后时间(s) 防抖,自动转换模式使用,1~60s
5      "iDawnTime":28800, // 晨曦时间(秒),定时转换模式使用,0~86400s
6      "iDuskTime":64800, // 黄昏时间(秒),需要大于iDawnTime时间, 定时转换模式使用,0~86400s
7      "iProfileSelect": 0 // 配置文件选择, 保持不变模式使用, 0=通用, 1=白天, 2=夜晚
8  }
```

3.4 图像基础调节

URL:/image/0/{scene_id}/adjustment

请求方法：PUT

请求示例：

```
1  {
2      "iBrightness": 57, // 亮度 0-100 -> 内部 *2.55 到 0-255 ; 过高泛灰
3      "iContrast": 50, // 对比度 0-100 -> *2.55 ; 过高丢失中间层次
4      "iHue": 50, // 色调 0-100 -> *2.55 ; 通常保持 50
5      "iSaturation": 50, // 饱和度 0-100 -> *2.55 ; 高会鲜艳失真, 低会发灰
6      "iSharpness": 50 // 锐度 0-100 ( 直接传值 ) ; 高易出现振铃/噪点放大
7  }
```

3.5 曝光参数

URL:/image/0/{scene_id}/exposure

请求方法：PUT

请求示例：

```
1  {  
2      "iExposureGain": 1, // 手动增益值 (整型, 范围依 sensor 一般 1-128+), 自动模式下读取当前值  
3      "sExposureMode": "auto", // 曝光模式: auto=自动; manual=手动;  
4      "sExposureTime": "1/6", // 曝光时间 \"分子/分母\" 或小数秒; 手动需 <= 帧周期 (1/FPS)  
5      "sGainMode": "auto" // 增益模式: auto=自动; manual=手动, 配合 iExposureGain  
6  }
```

3.6 背光参数

URL:/image/0/{scene_id}/blc

请求方法: PUT

请求示例:


```
1  {
2      "iBLCStrength": 28, // 背光补偿强度 0-100 , 越大逆光场景前景被提亮 , 但可能噪声提升
3      "iDarkBoostLevel": 50, // 暗部增强等级 1-100 , 提高暗区亮度 , 过高易噪点放大
4      "iHDRLevel": 1, // HDR宽动态强度等级 ( ISP3.0 tmo未直接用 ; 常用 0-3 或 0-10 )
5      "iHLCLLevel": 1, // 高亮抑制(HLC)强度 1-100 , 压制强光光源 ; 0 在驱动中会被修正为 1
6      "sBLCRegion": "open", // 背光补偿区域开关 : open=开启自动区域 ; close=关闭 ;
7      "sHDR": "close", // HDR 宽动态开关 : open=开启 ; close=关闭 ; 可能扩展 auto
8      "sHLC": "close" // 高亮抑制开关 : open=启用 ; close=关闭
9  }
```

3.7 白平衡参数

URL:/image/0/{scene_id}/white-blance

请求方法 : PUT

请求示例 :

```
1  {
2      "iWhiteBalanceCT": 2800, // color temperature value [2800, 7500]K
3      "sWhiteBlanceStyle": "manualWhiteBalance" // 模式 : auto=自动 ; manual=手动可调色温 ; Natural=自然光 ; streetlight=路灯 ,
        outdoor=室外
4  }
```

3.8 图像增强

URL:/image/0/{scene_id}/enhancement

请求方法：PUT

请求示例：

```
1  {  
2      "iSpatialDenoiseLevel": 50, // 空域降噪 0-100 ; 若 noise_reduce_mode=close 或 3dnr 会被强制回 50  
3      "iTemporalDenoiseLevel": 50, // 时域降噪 0-100 ; 若 noise_reduce_mode=close 或 2dnr 会被强制回 50 ; 过高拖影  
4      "sNoiseReduceMode": "mixnr" // 降噪模式：close=关闭(仍保持默认 50),mixnr=混合  
5  }
```

三、 录制设置

 [Vigil Record 最小 HTTP API 草案.pdf](#)

(md 直接粘进来有问题，晚点会整理好)

1. 录制控制

功能：获取录制配置

URL : /record/control

请求方法 : GET

响应示例 :

```
1  {
2  "sRecordCondition": "time/trigger", 对应定时录制和触发录制
3  "sRecordType": "video/image", 对应录制视频和图片
4  "iRecordTime": 60  录制时长，当录制类型sRecordType为video时有效
5  "sRecordFormat": "jpg/raw", 当录制类型为image时有效
6  "sTriggerMethod": "io/ai/serial", 当录制条件为触发录制时的触发方式
7  "iAntiShake": 5, 防抖时间
8  "sIoPin": "gpio1/...", io引脚
9  "sTriggerLevel": "height/low/up/down", 触发电平
10 "sTriggerComand": "record", 触发指令
11 "iPfTrigger": 7,
12 "sCurrentModel": "YOLOv5", 当前运行模型的模型名
13 "sModelType": "detect" 当前模型的任务类型
14 "sTriggerMask": "base64adf", 触发区域mask base64编码
15 "lCondition": [{
16     "sName": "条件名",
17     "iFrmae": 5, 进入mask区域帧数开始触发
18     "lLabel": ["person", "bus", "moto"] 触发类别标签,
19     ...
20 },
21
```

功能：录制配置

URL：/record/control

请求方法：POST

请求示例：

```
1  {
2  "sRecordCondition": "time/trigger", 对应定时录制和触发录制
3  "sRecordType": "video/image", 对应录制视频和图片
4  "iRecordTime": 60  录制时长，当录制类型sRecordType为video时有效
5  "sRecordFormat": "jpg/raw", 当录制类型为image时有效
6  "sTriggerMethod": "io/ai/serial", 当录制条件为触发录制时的触发方式
7  "iAntiShake": 5, 防抖时间
8  "sIoPin": "gpio1/...", io引脚
9  "sTriggerLevel": "height/low/up/down", 触发电平
10 "sTriggerComand": "record", 触发指令
11 "iPfTrigger": 7,
12 "sTriggerMask": "base64adf", 触发区域mask base64编码
13 "lCondition":[{
14     "sName": "条件名",
15     "iFrmae": 5, 进入mask区域帧数开始触发
16     "lLabel":["person","bus","moto"] 触发类别标签},
17     ...
18 ],
19 }
20
```

2. 日程设置

功能：获取当前录制的时间设置

URL：/record/schedule

请求方法：GET

响应示例：

```
1  {
2    "Monday": [    周一时间列表
3      {
4        "startTime": "00:45",  起始时间段
5        "endTime": "07:00"    结束时间段
6      },
7      {
8        "startTime": "08:45",  起始时间段
9        "endTime": "09:00"    结束时间段
10     }
11
12   ],
13   "Tuesday": [],
14   "Wednesday": [],
15   "Thursday": [],
16   "Friday": [],
17   "Saturday": [],
18   "Sunday": []
19 }
```

功能：设置每周的录制时间

URL：/record/schedule

请求方法：POST

请求示例：

```
1  {
2    "Monday": [    周一时间列表
3      {
4        "startTime": "00:45",  起始时间段
5        "endTime": "07:00"    结束时间段
6      },
7      {
8        "startTime": "08:45",  起始时间段
9        "endTime": "09:00"    结束时间段
10     }
11   ],
12   "Tuesday": [],
13   "Wednesday": [],
14   "Thursday": [],
15   "Friday": [],
16   "Saturday": [],
17   "Sunday": []
18 }
19
```

四、AI推理

1. 模型管理

功能：获取设备已有模型列表

URL：/model/list

请求方法：GET

响应示例：

```
1  [
2  {
3      "sFileName": "model.rknn",      模型文件名
4      "sModelNmae": "yolov5/yolov8",      模型名
5      "sModelType": "cl/det/seg/kp/obb/null",      模型任务类型, 分类/检测/分割/关键点/旋转检测/未知
6      "iModelSize": 22421,      模型大小
7  },
8  {
9      "sFileName": "model.rknn",      模型文件名
10     "sModelNmae": "yolov5/yolov8",      模型名
11     "sModelType": "cl/det/seg/kp/obb",      模型任务类型, 分类/检测/分割/关键点/旋转检测
12     "iModelSize": 22421,      模型大小
13 },
14 ...
15 ]
```

功能：上传模型

URL：/model/upload

请求方法：POST

Content-Type: multipart/form-data

2. 模型配置

功能：获取模型推理配置

URL：/model/config

请求方法：GET

请求示例：

```
1  {  
2  "sFileNmae": "model.rknn" 需要查询的模型文件名  
3  }
```

响应示例：

```
1  {
2  "sModelType": "cl/det/seg/kp/obb/null"  模型任务类型
3  "dConfig":
4  //分类
5  {
6  "fThr": 0.2 ,          置信度阈值
7  "iTopK": 5,           topk值
8  "lLabel":["dog","狗"]  类别列表
9  }
10 //检测/分割/旋转
11 {
12 "fThr": 0.2 ,          置信度阈值
13 "fIouThr": 0.2,        iou阈值
14 "iMaxObject": 300,      最大目标数
15 "lLabel":["dog","狗"]  类别列表
16 }
17 //关键点
18 {
19 "fThr": 0.2,
20 "lLabel":["eye" , "ear"]  类别列表
21 }
```

功能：设置模型推理配置

URL:/model/config

请求方法：POST

请求示例：

```
1  {
2  "sModelType": "cl/det/seg/kp/obb/null"  模型任务类型
3  "dConfig":
4  //分类
5  {
6  "fThr": 0.2 ,          置信度阈值
7  "iTopK": 5,           topk值
8  "lLabel":["dog","狗"]   类别列表
9  }
10 //检测/分割/旋转
11 {
12 "fThr": 0.2 ,          置信度阈值
13 "fIouThr": 0.2,        iou阈值
14 "iMaxObject": 300,      最大目标数
15 "lLabel":["dog","狗"]   类别列表
16 }
17 //关键点
18 {
19 "fThr": 0.2,
20 "lLabel":["eye" , "ear"]  类别列表
21 }
```

3. 模型输出

功能:推理输出配置获取

URL:/inference/config

请求方式 : GET

响应示例 :

```
1  {
2    "iFPS": 10,      FPS
3    "sTemplate": "{timestamp}: 检测到 {class} 置信度 {confidence} 位置 ({x1},{y1},{x2},{y2})"  输出模板
4  }
```

功能:配置推理输出

URL : /inference/config

请求方式 : POST

请求示例 :

```
1  {
2    "iFPS": 10,      FPS
3    "sTemplate": "{timestamp}: 检测到 {class} 置信度 {confidence} 位置 ({x1},{y1},{x2},{y2})"      输出模板
4  }
5
```


4. 输出配置

功能：获取触发记录

URL：/ws/inference/results

请求方式：websocket

响应示例：

```
1  {  
2    "sData": "2024-01-15 14:30:25: 检测到 person 置信度 0.85 位置 (120,80,250,300)"  
3  }
```

功能：获取推理输出配置

URL：/inference/output

请求方式：GET

响应示例：

```
1  {
2  "sOutputType": "mqtt/serial/modbus", 输出方式
3  "dConfig":
4  //mqtt
5  {
6  "sAddress": "mqtt.example.com", broker 地址
7  "sPort": "1883",      端口
8  "sTopic": "results/data", topic
9  "sUser": "name",      用户名
10 "sPassword": "adfasf" 密码 明文
11 }
12 //串口
13 {
14 "sPort": "UART1", 串口号
15 "sBaudRate": "115200"
16 }
17 //modbus
18 {
19 "sAddress": "192.168.1.111", 服务器地址
20 "sPort": "502"              端口号
21 }
```

功能：推理输出配置

URL：/inference/output

请求方式：POST

响应示例：

```
1  {
2  "sOutputType": "mqtt/serial/modbus", 输出方式
3  "dConfig":
4  //mqtt
5  {
6  "sAddress": "mqtt.example.com", broker 地址
7  "sPort": "1883",      端口
8  "sTopic": "results/data", topic
9  "sUser": "name",      用户名
10 "sPassword": "adfasf" 密码 明文
11 }
12 //串口
13 {
14 "sPort": "UART1", 串口号
15 "sBaudRate": "115200"
16 }
17 //modbus
18 {
19 "sAddress": "192.168.1.111", 服务器地址
20 "sPort": "502"              端口号
21 }
```

五、 终端和日志

1. 终端

功能：终端交互

URL:/ws/system/terminal

请求方式：websocket

备注：后端使用ttyd，前端使用Xterm.js

2. 日志

功能：系统日志

URL：/ws/system/logs

请求方式：websocket

响应示例：

```
1  {  
2    "sData": "2025-09-30 13:40:00 INFO 系统启动"  
3  }
```

