VIVOTEK WebAPI for All Series
- Video and Streaming

2014 Edition

Revision History

Version	Issue date	Editor	Comment
0.1	2013/05/14	Dave, Tsai	First draft for video & streaming WebAPI
0.2	2013/06/24	Dave, Tsai	Most content are preliminary done except Privacy Mask, Anysteam, scalable multicast, and backchannel multicast, etc.
0.3	2013/07/22	Dave, Tsai	1. Add affected APIs in 4.2.3.2 2. Add "capability_videoin_flexiblebitrate" in 3 API for Device Capability. This affects 5.5 Codec: H264, 5.6 Codec: MPEG4, and 5.7 Codec: MJPEG. The description of bit rate control and related APIs are modified. 3. Add "capability_videoin_streamcodec" and "capability_ videoin_c<n>_streamcodec " in 3 API for Device Capability. "capability_ videoin_c<n>_streamcodec " is a new added API. 4. Add suggestion: using multiple of 32 pixels for ROIed size in 5.3 ROI.
0.4	2013/08/15	Dave, Tsai	1. Add page number. 2. Fixed some wrong words.
0.5	2013/08/16	Dave, Tsai	1. Let Group: capability_videoin_c<n> be a individual table in 3 API for Device Capability. 2. Unify all API tables' head descriptions format as following: Group: videoin_c<n>_s<m>, n denotes the channel index , range is from 0 to "capability_nvideoin"-1, m denotes the stream index , range is from 0 to "capability_nmediastream"-1. 3. Fixed some wrong words again. 4. Mark unconfirmed parts. 5. Add description to "nmediastream", "streamcodec", "videoout_codec", "<codectype>_ maxframerate" in 3 API for Device Capability. 6. Mark new API is supported since which [httpversion]. 7. Define the represent format for a API and a API group in 1.2 Word Definition. 8. Modified the example in 4.2.2 WebAPI: Information for a mode

Version	Issue date	Editor	Comment
0.5B	2013/08/22	Dave, Tsai	Branch for technical writer. 1. Take Louis's modification. 2. Remove or replace it with TODO. 3. Add 2.1 Security Level 4. Add 2.2 General CGI URL Syntax for APIs 5. Add 4.1 Conventional channel API 6. Modify definition of [httpversion] in 1.2 Definitions and all related parts 7. Modify 5.1.2 Default resolution for streams 8. In 1.2 Definitions, change all "camera APIs" to "product WebAPIs".
0.5C	2013/08/29	Dave, Tsai	1. Remove " textonvideo_position" and "textonvideo_size" in 4.9 Text on video. These features need more discussion.
0.6	2013/09/02	Dave, Tsai	1. Unify the API name to "VIVOTEK WebAPI". 2. Change Chapter structure. 3. Add more description in 1 Introduction. 4. Enhance WebAPI version rules. 5. Remove " protocol_rtp_multicast_scalable" and "protocol_rtp_multicast_backchannel" in 3 API for Device Capability.
0.7	2014/08/22	Dave, Tsai	1. Change "bitraterestriction" default value to upperbound. 2. Modify "capability_eptz" description. 3. Add 'varifocal' to "capability_c<n>_lens_type". 4. Fix errors. 5. Fix the maximum port number from 65536 to 65535. 6. Fix the maximum port number of network_rtp_videoport, network_rtp_audioport, network_rtsp_c<n>_s<m>_multicast_videoport, network_rtsp_c<n>_s<m>_multicast_audioport to 65534. 7. Add 4.5 Privacy mask 8. Add new APIs to 4.9 Text-on-video 9. Modify description about anonymousviewing in 6.3 RTSP and 6.4 Server Push 10. Add 6.6 AnyStream 11. Re-order the chapter and section numbers... 12. Reconstruct references. 13. Change "network_rtsp_authmode" default value to basic.

Contents

1. Introduction

VIVOTEK WebAPI is an application programming interface (API) based on Web services that is used control services provided by VIVOTEK cameras and video servers. This API is accessed via CGI (Common Gateway Interface) commands with added security management features. Please refer to Chapter 2, “Method for accessing VIVOTEK WebAPI” for details.

This document contains information about the APIs for common video-related settings, such as text-on-video, resolution, codec type, etc. With these APIs, you can set up video streaming with customized settings that meet your specific needs or preferences. The APIs are based on the VIVOTEK video system as described in Section 1.1, “VIVOTEK Video System”.

Image tuning and security-related APIs are not described in this document.

VIVOTEK WebAPI has been under version control since version 0300. WebAPI version numbers use a string of 4 integer numbers, and is referenced as [httpversion]. [httpversion] can be retrieved using the "capability_api_httpversion" parameter. For a detailed definition, please refer to Section 1.2, “Definitions”.

Some APIs are commented with "Only available when [httpversion] >= XXXX", where "XXXX" denotes a specific WebAPI version, indicating that these APIs are mandatory if the value of [httpversion] is greater than or equal to XXXX for that product. For example, if the value of a product's [httpversion] is 0301, then the product must meet all definitions and requirements of the APIs that have been validated for use with [httpversion] values of 0300 and 0301.

With the exception of some obsolete APIs, APIs that are not commented as just described are mandatory if the value of a product's [httpversion] is greater than or equal to 0300, and are also available to older products whose [httpversion] value is less than 0300.

1.1. VIVOTEK Video System

There are three fundamental components in the video system of VIVOTEK IP cameras and video servers: channels, streams, and streaming services. Each of these components is detailed in the following sections.

1.1.1. Channel

An IP camera and video server may have one or more video sources, and each such source corresponds to a "channel". That is, one channel provides access to one video source.

A video source may be an image sensor, image module, analog CCTV signal, etc. Most IP cameras have one channel, and video servers may have 1, 4, 8, or more channels. The number of channels provided by a device is recorded in "capability_nvideoin".

In WebAPI, the group videoin is designed for video-related settings, with the suffix "c<n>" indicating the channel index. <n> is defined as an integer from 0 to one less than the value of "capability_nvideoin". That is, the group: videoin_c0 contains video settings for channel 0, the group: videoin_c1 contains video settings for channel 1, and so on.

In the past, IP cameras had only one channel. Therefore, older cameras do not use the suffix "c<n>" in WebAPI. Those settings belonging to a specific channel are located in the group: videoin. For backwards compatibility, we have retained some older APIs in the group: videoin for IP cameras. These settings APIs maintain a one-to-one mapping to those in the group: videoin_c0. However, use of these older APIs is not recommended, and they may be removed without advance notice.

Image tuning is also performed during this phase. For our products, we use either an ISP, a sensor, or both to tune image quality. Associated APIs are located in the group: videoin_c<n> and the group: image_c<n>. However, image settings are not covered by this document. For more details on image settings, please refer to the document “VIVOTEKCameraWebAPI_Image.doc”.

1.1.2. Stream

Each channel can provide multiple streams, each with its own resolution, ROI (region of interest), frame rate, codec type, encoded bitrate, etc. ePTZ can be applied to streams as well. The number of streams per channel is recorded in "capability_nmediastream".

In WebAPI, the group: videoin_c<n>_s<m> is designated for a stream’s settings, with <n> denoting the channel index, whose range is from 0 to one less than the value of "capability_nvideoin", and <m> denotes the stream index, whose range is from 0 to one less than the value of "capability_nmediastream". For more details, please refer to Chapter 5, “Stream Domain API".

Another group, roi_c<n>_s<m>, is designated to store a stream’s ROI settings, with <n> denoting the channel index, whose range is from 0 to one less than the value of "capability_nvideoin", and <m> denoting the stream index, whose range is from 0 to one less than the value of "capability_nmediastream". This group is only available when the stream supports ePTZ. For more details, please refer to Chapter 5.3, “ROI”.

1.1.3. Streaming Service

The streaming service is responsible for sending out encoded video data as a stream to clients. There are 4 such services: RTSP, Server Push, single snapshot, and Anystream.

The RTSP service supports streaming of encoded data using any codec. If a device supports RTSP, then "capability_protocol_rtsp" is set to 1. For more details, please refer to Section 6.3, “RTSP”.

Server Push supports streaming of MJPEG-encoded data only. If the device supports Server Push, then "capability_protocol_spush_mjpeg" is set to 1. For more details, please refer to Section 6.4, “Server Push”.

Single snapshot is an HTTP interface to capture a single snapshot from the device. For more details, please refer to Section 6.5, “Single Snapshot”.

Anystream is a useful feature that enables access to a stream with customized settings. It allows video parameters to be set dynamically by passing them via a URL. For more details, please refer to Section 6.6, “AnyStream”.

1.2. Definitions

Term	Definition
Channel index or <n>	The index number of a channel, start from 0. The maximum number is defined as "capability_nvideoin" -1. We often use short name: <n> for it.
Stream index or <m>	The index number of a stream, start from 0. The maximum number is defined as "capability_nmediastream"-1. We often use short name: <m> for it.
"ONEAPI"	One API is enclosed with double quote. EX: "videoin_c0_s0_resolution" with full path. "resolution" in short. "videoin_c<n>_s<m>_resolution" means "resolution" in all channels' streams.
Group: APIGROUP	Prefix: "Group:" or "group:" following a group name or path represent a group of APIs. EX: Group: capability Group: capbility_videoin_c<n>
[httpversion]	The version of VIVOTEK WebAPI with 4 integers, ex: 0300. The [httpversion] can be got from "capability_api_httpversion". Ex: "capability_api_httpversion"=0300a The 4 integer numbers are WebAPI version. The 5th character is model-based version for API bug-fix and it's default to "a". Ex: If some APIs in a model does not follow the API definition of 0300, we will fix them and change [httpversion] to 0300b.
Available	The API is listed in product WebAPIs.
Non-available	The API is not in product WebAPIs.
Valid	The API is listed in product WebAPIs, and is functional.
Non-valid	The API is listed in product WebAPIs, but is malfunction in this status.
Channel or CH	One video input. One sensor is one channel, one VNC input is one channel. For video server with 4 VNC inputs, we call it as 4 channels video server.
Normalized range	A uniform value range like: 1~100, 0~9999. This provides abstraction layer to hide actual values in lower level, because actual values are different from product to product. Normalized value is translated to actual value internally. The relationship may be one-to-one or multiple-to-one, but never be one-to-multiple.
<codectype>	Any codec type listed in "capability_videoin_codec". Possible options are "mpeg4", "h264", "mjpeg", etc.

Term	Definition
<Integer>	Any single integer number in 32-bits. The range is -2147483648~2147483647.
<IntegerA~IntegerB>	Any integer ranges from A to B. A must < B. Ex: <0~3>, <1~100>, etc.
<Positive Integer>	Any single positive integer number in 32-bits. The range is 1~ 4294967295.
<X,Y>	The point position in 2D. X is a single integer number. Y is a single integer number.
<WxH>	The format for resolution. W is the pixel number of width. H is the pixel number of height. Ex: 1920x1080, 2048x1536
<IP>	An IP address, either IPv4 or IPv6.
<String[N]>	A string with N characters.

1.3. Suggested Setup Procedures

This section provides a suggested procedure to help users set up video features and configure the desired streaming format on a product with factory-default settings. Please note that the default values for all parameters and settings were chosen for the most general use case and the following setup procedures are not mandatory.

Step 1. Open streaming
The first and the most important step is checking image to ensure everything is normal.

When factory default settings are used, the first stream provided by all VIVOTEK products is full-view (the maximum resolution, FOV) H.264 video (or MPEG4 for the older 7-series products). You can use the following URL to access streams with RTSP-compatible players such as VLC:

```
rtsp://<IP>/live.sdp
```

The device IP can be found using the tool: VIVOTEK Installation Wizard 2.

Step 2: Set up video mode
See what video modes are supported by a given product, and select the mode that best meets your requirements. The procedures for accessing information on available modes on a camera and for selecting modes are described in Section 4.2 Video Mode.

The term *mode* refers to a particular combination of resolution and frame rate for the video input. This step can be skipped if you wish to use the default full-view (resolution) mode.

Step 3: Set up image orientation
Described in Section 4.4 Video Orientation.

Step 4: Set up privacy mask
Described in Section 4.5 Privacy mask.

Step 5: Set up power line frequency or modulation type
Described in Section 4.6 Power Line Frequency and section 4.7 Modulation.

Step 6: Set up color and text-on-video
Described in Section 4.8 Color and Section 4.9 Text-on-video.

Step 7: Set each stream's region of interest (ROI)
Described in Section 5.3 Region of Interest.

This step can be skipped if you wish to use the full FOV.

Step 8: Set each stream's resolution
Described in Section 5.1 Resolutions for CMOS Cameras or Section 5.2 Resolution for Video Servers and Cameras with a CCD Module.

Step 9: Configure each stream's codec type and settings
Described in Section 5.4 Codec type, Section 5.5 Codec: H264, Section 5.6 Codec: MPEG4, and Section 5.7 Codec: MJPEG.

Step 10: Set up the streaming service
Described in Section 6 Streaming Service Domain API.

Step 11: Open streaming again
Described in Section 6.1 Opening streaming

2. Method for Accessing VIVOTEK WebAPI

2.1. Security levels

All VIVIOTEK WebAPIs have been assigned a security level as shown below:

SECURITY LEVEL	SUB-DIRECTORY	DESCRIPTION
0	anonymous	Unprotected.
1 [view]	viewer	Can view, listen, and talk to camera.
4 [operator]	operator	Operator access rights can modify most of the camera's parameters except some privileges and network options.
6 [admin]	admin	Administrator access rights can fully control the camera's operations.
7	N/A	Internal APIs. Unable to be changed by any external interfaces.

A viewer account can access all APIs with security level 0 and 1. An operator account can access all APIs with security level 0, 1, or 4. An admin account can access all APIs except internal APIs.

Access management is based on the URL directory structure and is described in following paragraphs.

2.2. General CGI URL Syntax for APIs

The APIs described in this document include those related to device capability and video/streaming. These APIs can be manipulated by passing get/set as parameters in CGI commands. We have included basic syntax and usage examples in this section.

2.2.1. Obtaining Server API Values

Note: Access privileges are based on the URL directory structure.
Method: GET/POST

Syntax:

```
http://<servername>/cgi-bin/anonymous/getparam.cgi?[<parameter>]
[&<parameter>…]

http://<servername>/cgi-bin/viewer/getparam.cgi?[<parameter>]
[&<parameter>…]

http://<servername>/cgi-bin/operator/getparam.cgi?[<parameter>]
[&<parameter>…]

http://<servername>/cgi-bin/admin/getparam.cgi?[<parameter>]
[&<parameter>…]
```

In the above, *<parameter>* should be replaced with *<group>* or with "ONEAPI" in a full path. The value of <group> denotes a path for an API group—e.g. "videoin_c0_s0_h264". "ONEAPI" in a full path denotes a single API such as "videoin_c0_s0_h264_resolution", "videoin_c0_s0_h264_maxframe", etc.

If no parameters are specified, all the APIs on the server will be returned. If only <group> is specified, the APIs of the related group will be returned.

When querying API values, the current API values are returned.
A successful control request returns parameter pairs as follows:

Returns:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Context-Length: <length>\r\n
\r\n
<parameter pair>
```

where <parameter pair> is
<parameter>=<value>\r\n
[<parameter pair>]

<length> is the actual length of the return message.

Example: Requesting an IP address, and a typical response

```
Request:
http://192.168.0.123/cgi-bin/admin/getparam.cgi?network_ipaddress

Response:
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Context-Length: 33\r\n
\r\n
network_ipaddress=192.168.0.123\r\n
```

2.2.2. Setting Server API Values

Note: Access privileges are based on the URL directory structure.
Method: GET/POST

Syntax:

```
http://<servername>/cgi-bin/anonymous/setparam.cgi? <parameter>=<value>
[&<parameter>=<value>…][&return=<return page>]

http://<servername>/cgi-bin/viewer/setparam.cgi? <parameter>=<value>
[&<parameter>=<value>…][&return=<return page>]

http://<servername>/cgi-bin/operator/setparam.cgi? <parameter>=<value>
[&<parameter>=<value>…][&return=<return page>]

http://<servername>/cgi-bin/admin/setparam.cgi? <parameter>=<value>
[&<parameter>=<value>…][&return=<return page>]
```

PARAMETER	DESCRIPTION
<parameter>	"ONEAPI" in a full path denotes a single API like: "videoin_c0_s0_h264_resolution", "videoin_c0_s0_h264_maxframe", etc.
<value>	The assigned *<value>* to the *<parameter>.*
<return page>	Redirect to the page *<return page>*after the *<parameter>* is assigned. The *<return page>*can be a full URL path or relative path according to the current path. If you omit this parameter, it will redirect to an empty page. (Note: The return page can be a general HTML file (.htm, .html). It cannot be a CGI command or have any extra parameters. This parameter must be placed at the end of the parameter list

Returns:

```
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Context-Length: <length>\r\n
\r\n
<parameter pair>
```

where <parameter pair> is
<parameter>=<value>\r\n
[<parameter pair>]
Only the parameters that you set and that are readable will be returned.

Example: Setting the IP address of a server to 192.168.0.123:

```
Request:
http://myserver/cgi-bin/admin/setparam.cgi?network_ipaddress=192.168.0.123

Response:
HTTP/1.0 200 OK\r\n
Content-Type: text/html\r\n
Context-Length: 33\r\n
\r\n
network_ipaddress=192.168.0.123\r\n
```

3. Device Capabilities API

The following table lists APIs that represent the video and streaming features of a device. These APIs indicate the capabilities of the system based on hardware and firmware configurations.

Group: **capability**
* All APIs are read-only and all security levels are 0/7 (get/set).

Name	Available Values	Description
api_httpversion	See description	The version of VIVOTEK WebAPI with 4 integers plus 1 alphabet, ex: 0300a The 4 integer numbers are WebAPI version, we use short name: [httpversion] for it in this document. The 5th character is model-based version for API bug-fix and it's default to "a". Ex: If some APIs in a model does not follow the API definition of 0300, we will fix them and change this API value to 0300b.
videoin_type	0, 1, 2	0 => Interlaced CCD/ Interlaced CCTV signal 1 => Progressive CCD 2 => CMOS * For video server, this is set to 0.
nvideoin	<Positive Interger>	Number of video inputs = channel number. * More detail, see 1.1.1 Channel
nmediastream	<Positive Interger>	Number of stream per channel. The number is not including Anystream. * More detail, see 1.1.2 Stream
nanystream	0, <Positive Integer>	Number of Anystream per channel. * More detail, see 6.6 AnyStream
videoin_codec	mpeg4, mjpeg, h264	Available codec of a device, split by comma. The sequence is not limited. EX: IP7361 supports MPEG4 and MJPEG, then this is "mpeg4,mjpeg". IP8371E supports MPEG4, MJPEG and H.264, then this is "mpeg4,mjpeg,h264"
videoin_streamcodec	A list of <Positive Integer>	This equals "capability_videoin_c0_streamcodec". * This is kept for compatibility.

Name	Available Values	Description
videoin_c<n>_streamcodec	A list of <Positive Integer>	Represent supported codec types of each stream. This contains a list of positive integers, split by comma. Each one stands for a stream, and the definition is as following: Bit 0: Support MPEG4. Bit 1: Support MJPEG Bit 2: Support H.264 Ex: IP8371E has 1 channel 4 streams, all streams support MPEG4, MJPEG, and H.264, then this is "7,7,7,7". Ex: IP7161 has 1 channel 4 streams, all streams support MPEG4, and MJPEG, then this is "3,3,3,3". * The number of integers is equal to "capability_nmediastream". * Only available when [httpversion] >= 0300
videoin_flexiblebitrate	0, 1	1: Support flexible bit rate control 0: Non-support flexible bit rate control. * More detail, see 5.5 Codec: H264, 5.6 Codec: MPEG4, 5.7 Codec: MJPEG * Only available when [httpversion] >= 0300
eptz	0, <Positive Integer>	For "nvideoin" = 1, the definition is as following: A 32-bits integer, each bit can be set separately as follows: Bit 0 => 1st stream supports ePTZ or not. Bit 1 => 2nd stream supports ePTZ or not, and so on. For nvideoin >= 2, the definition is different: First all 32 bits are divided into groups for channel. Ex: nvideoin = 2, bit 0~15 are the 1st group for 1st channel, bit 16~31 are the 2nd group for 2nd channel. nvideoin = 3, bit 0~9 are the 1st group for 1st channel, bit 10~19 are the 2nd group for 2nd channel, bit 20~31 are the 3rd group for 3rd channel. Then, the 1st bit of the group indicates 1st stream of a channel support ePTZ or not. The 2nd bit of the group indicates 2nd stream of a channel support ePTZ or not, and so on. * For most products, the last stream of a channel will not support ePTZ. It is reserved for full view of the channel. For some dual-stream products, both streams support ePTZ.
nvideoout	0, <Positive Integer>	Number of video out interface.

Name	Available Values	Description
videoout_codec	-, ntsc, pal	Current output information about video out. 1st element for 1st video-out, 2nd element for 2nd video-out, and so on. The number of element depends on "capability_nvideooout". "-": Video-out is not available ntsc: NTSC analog output pal: PAL analog output Ex: "nvideoout"=0, "videoout_codec"=- "nvideoout"=1 with NTSC, "videoout_codec"=ntsc "nvideoout"=1 with PAL, "videoout_codec"=pal "nvideoout"=2 with both NTSC, "videoout_codec"=ntsc,ntsc * For camera, this feature is controlled by physical jump on device. No WebAPI to control it. This value is set only on camera power-on and maintains the status. * Only available when [httpversion] >= 0301
privacymask_axistype	0, 1, 2	Descript the technical method to implement a privacy mask. 0: Unidentified. The firmware is in an early version 1: The privacy mask is a 2D version 2: The privacy mask is a 3D version The 3D privacy mask means that camera will maintain the mask's position while panning and tilting and adjust relative size while zooming in/out. In the 2D version, each mask will always keep in the viewing window with a fixed size, even though the camera had been driven by PTZ motors. * More detail, see 4.5 Privacy mask * Only available when [httpversion] >= 0301
privacymask_shape	rectangle, polygon	The shape of privacy mask window. rectangle: Rectangle masked window. polygon: Use 4 points to locate the masked window. * More detail, see 4.5 Privacy mask * Only available when [httpversion] >= 0301
privacymask_num	0, <Positive Integer>	The number of privacy mask window. * More detail, see 4.5 Privacy mask * Only available when [httpversion] >= 0301
textonvideo_position	top, bottom	Available text postions. top = Embed text and time information on left-top corner. bottom = Embed text and time information on left-down corner. Ex: capability_textonvideo_position='top,bottom' Ex: capability_textonvideo_position='top' * More detail, see 4.9 Text-on-video * Only available when [httpversion] >= 0301

Name	Available Values	Description
textonvideo_size	15, 25, 30, -	Available font size for text on video. "-" means changing font size is not supported. Ex: capability_textonvideo_size='15,25,30' * More detail, see 4.9 Text-on-video * Only available when [httpversion] >= 0301
protocal_rtsp	0, 1	1: Support RTSP service 0: Non-support RTSP service * More detail, see 6.3 RTSP
protocol_rtp_tcp	0, 1	1: Support RTP over TCP 0: Non-support RTP over TCP * More detail, see 6.3 RTSP
protocol_rtp_http	0, 1	1: Support RTP over HTTP 0: Non-support RTP over HTTP * More detail, see 6.3 RTSP
protocol_spush_mjpeg	0, 1	1: Support server push service for MJPEG 0: Non-support server push service for MJPEG * More detail, see 6.4 Server Push
protocol_maxconnection	<Positive Integer>	The maximum number of allowed simultaneous connections. * More detail, see 6.2 Maximum Number of Active Streams

Group: capability_videoin_c<n>,

n denotes the channel index, with a range from 0 to one less than the value of “capability_nvideoin".

* Only available when the value of [httpversion] is greater than or equal to 0300.
* All APIs are read-only and all security levels are 0/7 (get/set).

Name	Available Values	Description
nmode	<Positive Integer>	Indicate how many video modes supported by this channel.
maxsize	<WxH>	The maximum resolution of all modes in this channel, the unit is pixel. * We call this as "**Full Scene**".
lens_type	fisheye, fixed, varifocal, changeable, motor, -	The lens type of this channel. fisheye: Fisheye lens fixed: Build-in fixed-focus lens. varifocal: Build-in varifocal lens. changeable: changeable lens. Like box-type camera, users can install any C-Mount or CS-Mount lens as they wish. motor: Lens with motor to support zoom, focus, etc. -: N/A * Only available when [httpversion] >= 0301
lens_modelname	<String[64]>	**Optional** model name for lens. Original purpose of this is providing a method to recognize fisheye lens for de-warping. If there is no specific intention to identify lens, this should be set to "-". * Only available when [httpversion] >= 0301
mode	<Integer>	Indicate current video mode.
nresolution	<Positive Integer>	How many resolution options (listed in "resolution") in current video mode.
resolution	A list of <WxH>	Resolution options in current video mode. These options are the possible options for "videoin_c<n>_s<m>_resolution", see 5.1 Resolutions for CMOS Cameras. The last one is the maximum resolution in current mode. * The element number is defined as "nresolution" in this group.
maxframerate	A list of <Integer>	Indicate frame rate that the video source outputs in current video mode. One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on.

Name	Available Values	Description
mpeg4_maxframe rate	A list of <Integer> and "-"	Maximum fps that the device can encoded one stream with MPEG4 on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.6 Codec: MPEG4. * Only available when 'mpeg4' is listed in "capability_videoin_codec".
mpeg4_maxbitrat e	<Positive Integer>	Maximum bitrates of MPEG4. The unit is bps. * Only available when 'mpeg4' is listed in "capability_videoin_codec". * More details, see 5.6 Codec: MPEG4.
mjpeg_ maxframerate	A list of <Positive Integer> and "-"	Maximum fps that the device can encoded one stream with MJPEG on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.7 Codec: MJPEG. * Only available when 'mjpeg' is listed in "capability_videoin_codec".
mjpeg_maxbitrate	<Positive Integer>, -	Maximum bitrates of MJPEG. The unit is bps. "-" means MJPEG does not support bit rate control. * Only available when 'mjpeg' is listed in "capability_videoin_codec". * More details, see 5.7 Codec: MJPEG.

Name	Available Values	Description
h264_ maxframerate	A list of <Positive Integer> and "-"	Maximum fps that the device can encoded one stream with H.264 on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.5 Codec: H264. * Only available when 'h264' is listed in "capability_videoin_codec".
h264_maxbitrate	<Positive Integer>	Maximum bitrates of H.264. The unit is bps. * Only available when 'h264' is listed in "capability_videoin_codec". * More details, see 5.5 Codec: H264.

4. Channel Domain API

4.1. Conventional Channel API

For VIVOTEK's 7000 series and some models in the 8000 series, the conventional channel API is used. Those products support only a single native sensor output resolution. Note that the conventional API provides only limited information regarding the video and streams.

Group: capability_videoin

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
resolution	A list of <WxH>	0/7	Resolution options available. These resolution options are available to all video streams unless otherwise specified.
maxframerate	A list of <Integer>	0/7	Indicate maximum frame rate available for the corresponding resolution. Those values are one-to-one mapping to the "resolution" parameter in this group.

Taking the IP8161, IP7161 as examples, those parameters are:

```
capability_videoin_resolution='176x144,320x240,640x480,800x600,1280x960,1600x1200'
capability_videoin_maxframerate='30,30,30,30,15,15'
```

In brief, there are 6 resolutions available for all video streams and each has a corresponding maximum frame rate value. For more details, please refer to Section 4.3, "Cropping".

The conventional channel API provides only limited and imprecise information. Therefore, integrators are encouraged to use the "Video mode" API (documented starting in the next section) for all the latest VIVOTEK cameras so as to retrieve more complete and accurate information.

For information pertaining to special fisheye models, please refer to Section 4.2.4 Fisheye Products without Video Mode.

4.2. Video Mode

Video mode, also called sensor mode, is a newly designed data structure to describe a device’s video capabilities.

Currently available megapixel sensors can switch between different settings to output video data with different resolutions and frame rates, such as 3M@30fps, or 1080P@60fps. The resolution affects the size of the visible area, and the frame rate affects the frame rate when streaming. Therefore, we have designed a data structure to describe these parameters, and made it available to all products. Besides video mode, some information about a channel, such as maximum encoded bitrates, is also included in this data structure.

If a model supports video mode, the value of [httpversion] must be greater than or equal to 0300.

4.2.1. WebAPI: Channel-specific Information

Information specific to an individual channel includes the following:

Group: capability_videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of“capability_nvideoin".
* Only available when the value of [httpversion] is greater than or equal to 0300.

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
nmode	<Positive Integer>	0/7	Indicate how many video modes supported by this channel.
maxsize	<WxH>	0/7	The maximum resolution of all modes in this channel, the unit is pixel. * We call this as "**Full Scene**".
lens_type	fisheye, fixed, varifocal, changeable, motor, -	0/7	The lens type of this channel. fisheye: Fisheye lens fixed: Build-in fixed-focus lens. varifocal: Build-in varifocal lens. changeable: changeable lens. Like box-type camera, users can install any C-Mount or CS-Mount lens as they wish. motor: Lens with motor to support zoom, focus, etc. -: N/A * Only available when [httpversion] >= 0301
lens_modelname	<String[64]>	0/7	**Optional** model name for lens. Original purpose of this is providing a method to recognize fisheye lens for de-warping. If there is no specific intention to identify lens, this should be set to "-". * Only available when [httpversion] >= 0301
mode	<Integer>	0/7	Indicate current video mode.

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
nresolution	<Positive Integer>	0/7	How many resolution options (listed in "resolution") in current video mode.
resolution	A list of <WxH>	0/7	Resolution options in current video mode. These options are the possible options for "videoin_c<n>_s<m>_resolution", see 5.1 Resolution. The last one is the maximum resolution in current mode. * The element number is defined as "nresolution" in this group.
maxframerate	A list of <Integer>	0/7	Indicate the frame rate video source outputs in current video mode. One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on.
mpeg4_maxframerate	A list of <Integer> and "-"	0/7	Maximum fps that the device can encoded one stream with MPEG4 on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.6 Codec: MPEG4. * Only available when 'mpeg4' is listed in "capability_videoin_codec".
mpeg4_maxbitrate	<Positive Integer>	0/7	Maximum bitrates of MPEG4. The unit is bps. * Only available when 'mpeg4' is listed in "capability_videoin_codec". * More details, see 5.6 Codec: MPEG4.

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
mjpeg_ maxframerat e	A list of <Positive Integer> and "- "	0/7	Maximum fps that the device can encoded one stream with MJPEG on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.7 Codec: MJPEG. * Only available when 'mjpeg' is listed in "capability_videoin_codec".
mjpeg_maxb itrate	<Positive Integer>, -	0/7	Maximum bitrates of MJPEG. The unit is bps. "-" means MJPEG does not support bit rate control. * Only available when 'mjpeg' is listed in "capability_videoin_codec". * More details, see 5.7 Codec: MJPEG.
h264_ maxframerat e	A list of <Positive Integer> and "- "	0/7	Maximum fps that the device can encoded one stream with H.264 on resolutions in current video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter may be changed when "videoin_c<n>_cmosfreq"=50 or "videoin_c<n>_modulation"=pal. Ex: 30 fps is changed to 25 fps, 60 fps is changed to 50 fps, and so on. More details, see 5.5 Codec: H264. * Only available when 'h264' is listed in "capability_videoin_codec".
h264_maxbit rate	<Positive Integer>	0/7	Maximum bitrates of H.264. The unit is bps. * Only available when 'h264' is listed in "capability_videoin_codec". * More details, see 5.5 Codec: H264.

Example 1:

IP8371E 3M mode:

```
capability_videoin_codec=''mpeg4,mjpeg,h264''
capability_videoin_c0_nmode='2'
capability_videoin_c0_maxsize='2048x1536'
capability_videoin_c0_lens_type='motor'
capability_videoin_c0_mode='0'
capability_videoin_c0_nresolution='7'
capability_videoin_c0_resolution='176x144,320x240,640x480,800x600,1280x96
0,1600x1200,2048x1536'
capability_videoin_c0_maxframerate='30,30,30,30,30,30,30'
capability_videoin_c0_mpeg4_maxframerate='30,30,30,30,30,30,30'
capability_videoin_c0_mpeg4_maxbitrate= 40000000
capability_videoin_c0_mjpeg_maxframerate='30,30,30,30,30,30,30'
capability_videoin_c0_mjpeg_maxbitrate=40000000
capability_videoin_c0_h264_maxframerate='30,30,30,30,30,30,30'
capability_videoin_c0_h264_maxbitrate=40000000
```

This information displayed includes the following:

1. There are 2 video modes on channel 0 of this device.
2. The maximum resolution of channel 0 is 2048x1536.
3. The current video mode is mode 0.
4. In the current mode, the source (image sensor) outputs 2048x1536 (full view) video data at 30 fps.
5. The current video mode has 7 resolution options, any of which can be applied to a stream.
6. On this device with the current mode, all codecs at all resolutions can achieve 30 fps.
7. Maximum bitrates for MPEG4, MJPEG, and H.264 are all 40000000 bps.

Example 2:

1080P mode on the IP8371E:

```
capability_videoin_codec=''mpeg4,mjpeg,h264''
capability_videoin_c0_nmode='2'
capability_videoin_c0_maxsize='2048x1536'
capability_videoin_c0_lens_type='motor'
capability_videoin_c0_mode='1'
capability_videoin_c0_nresolution='7'
capability_videoin_c0_resolution='176x144,384x216,640x360,1280x720,1360x768,1600x904,1920x10
80'
capability_videoin_c0_maxframerate=' 60,60,60,60,60,60,60'
capability_videoin_c0_mpeg4_maxframerate=' 60,60,60,60,60,60,52'
capability_videoin_c0_mpeg4_maxbitrate= 40000000
capability_videoin_c0_mjpeg_maxframerate=' 60,60,60,60,60,60,60'
capability_videoin_c0_mjpeg_maxbitrate=40000000
capability_videoin_c0_h264_maxframerate=' 60,60,60,60,60,60,60'
capability_videoin_c0_h264_maxbitrate=40000000
```

The information displayed includes the following:

1. There are 2 video modes on channel 0 of this device.
2. The maximum resolution of channel 0 is 2048x1536.
3. The current video mode is mode 1.
4. In the current mode, the source (image sensor) outputs 1920x1080 video data at 60 fps.
5. The current video mode has 7 resolution options, any of which can be applied to a stream.
6. On this device with the current mode, all codecs at all resolutions can achieve 60 fps, with the exception of MPEG4. Encoding 1920x1080 video with MPEG4 only allows a frame rate of 52 fps.
7. Maximum bitrates for MPEG4, MJPEG, and H.264 are all 40000000 bps.

4.2.2. WebAPI: Mode-specific Information

This structure represents information about a mode.

Group: capability_videoin_c<n>_mode<m>,
n denotes the channel index, with a range from 0 to one less than the value of"capability_nvideoin";
m denotes the mode index, with a range from 0 to one less than the value of "capability_videoin_c<n>_nmode".

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
effectivepixel	<WxH>	0/7	The visible area in this video mode. The unit is pixel in source. * This value must <= "capability_videoin_c<n>_maxsize" * If "effectivepixel"<"capability_videoin_c<n>_maxsize", then the visible area is located at the center of full scene.
outputsize	<WxH>	0/7	The output size of source, equal to the captured size by device, in this video mode. The unit is pixel. This value is used as a basic coordinate system for many features, like ePTZ, privacy mask, motion, etc. * Source (most for image sensor) may perform scale or binning, etc on image data, and output data with smaller size. This parameter is designed to represent this.
binning	0, 1, 3	0/7	Indicate binning is used or not in this video mode. 0: No binning 1: 2x2 binning 3: 3x3 binning * Binning is a technology to increase light sensitivity by combining multiple pixels to one. The drawback is reduced resolution. We design this parameter to disclose this information.
nresolution	<Positive Integer>	0/7	How many resolution options in this video mode.
resolution	A list of <WxH>	0/7	Resolution options in this video mode. The last one is the maximum resolution in this video mode. * The element number is defined as "nresolution" in this group.

NAME	VALUE	SECURITY (get/set)	DESCRIPTION
maxframerate	A list of <Positive Integer>	0/7	Indicates frame rate that the video source outputs in this video mode. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter records the frame rate when "videoin_c<n>_cmosfreq"=60 or "videoin_c<n>_modulation"=ntsc
maxfps_mpeg4	A list of <Positive Integer> and "-"	0/7	Maximum fps which the device can encoded with MPEG4 on resolutions in this video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter records the frame rate when "videoin_c<n>_cmosfreq"=60 or "videoin_c<n>_modulation"=ntsc * Only available when 'mpeg4' is listed in "capability_videoin_codec".
maxfps_mjpeg	A list of <Positive Integer> and "-"	0/7	Maximum fps which the device can encoded with MJPEG on resolutions in this video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter records the frame rate when "videoin_c<n>_cmosfreq"=60 or "videoin_c<n>_modulation"=ntsc * Only available when 'mjpeg' is listed in "capability_videoin_codec".
maxfps_h264	A list of <Positive Integer> and "-"	0/7	Maximum fps which the device can encoded with H.264 on resolutions in this video mode. "-" means not support. * One to one mapping to the resolution in "resolution". * The element number is defined as "nresolution" in this group. * This parameter records the frame rate when "videoin_c<n>_cmosfreq"=60 or "videoin_c<n>_modulation"=ntsc * Only available when 'h264' is listed in "capability_videoin_codec".
description	<String[128]>	0/7	Description about this mode.

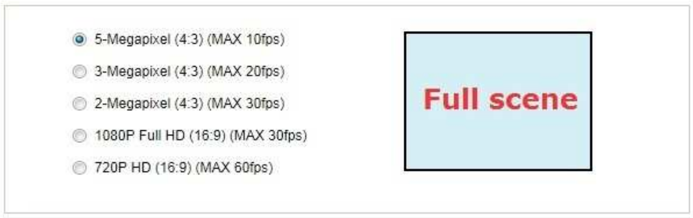
Example (FD8372):

```
capability_nvideoin=1
capability_videoin_codec=''mpeg4,mjpeg,h264''
capability_videoin_c0_nmode=5
capability_videoin_c0_maxsize='2560x1920'
capability_videoin_c0_mode0_effectivepixel='2560x1920'
capability_videoin_c0_mode0_outputsize='2560x1920'
capability_videoin_c0_mode0_binning=0
capability_videoin_c0_mode0_nresolution=7
capability_videoin_c0_mode0_resolution='176x144,320x240,640x480,800x600,1600x1200,2048x153
6,2560x1920'
capability_videoin_c0_mode0_maxframe='13,13,13,13,13,13,13'
capability_videoin_c0_mode0_maxfps_mpeg4='13,13,13,13,13,-,-'
capability_videoin_c0_mode0_maxfps_mjpeg='13,13,13,13,13,13,13'
capability_videoin_c0_mode0_maxfps_h264='13,13,13,13,13,13,10'
capability_videoin_c0_mode0_description=' 5-Megapixel (4:3) (MAX 10fps)'
capability_videoin_c0_mode1_effectivepixel='2048x1536'
capability_videoin_c0_mode1_outputsize='2048x1536'
capability_videoin_c0_mode1_binning=0
capability_videoin_c0_mode1_nresolution=6
capability_videoin_c0_mode1_resolution='176x144,320x240,640x480,800x600,1600x1200,2048x153
6'
capability_videoin_c0_mode1_maxframe='20,20,20,20,20,20'
capability_videoin_c0_mode1_maxfps_mpeg4='20,20,20,20,20,-'
capability_videoin_c0_mode1_maxfps_mjpeg='20,20,20,20,20,20'
capability_videoin_c0_mode1_maxfps_h264='20,20,20,20,20,20'
capability_videoin_c0_mode1_description='3-Megapixel (4:3) (MAX 20fps)'
capability_videoin_c0_mode2_effectivepixel='1600x1200'
capability_videoin_c0_mode2_outputsize='1600x1200'
capability_videoin_c0_mode2_binning=0
capability_videoin_c0_mode2_nresolution=5
capability_videoin_c0_mode2_resolution='176x144,320x240,640x480,800x600,1600x1200'
capability_videoin_c0_mode2_maxframe='30,30,30,30,30'
capability_videoin_c0_mode2_maxfps_mpeg4='30,30,30,30,30'
capability_videoin_c0_mode2_maxfps_mjpeg='30,30,30,30,30'
capability_videoin_c0_mode2_maxfps_h264='30,30,30,30,30'
capability_videoin_c0_mode2_description='2-Megapixel (4:3) (MAX 30fps)'
capability_videoin_c0_mode3_effectivepixel='1920x1080'
capability_videoin_c0_mode3_outputsize='1920x1080'
capability_videoin_c0_mode3_binning=0
capability_videoin_c0_mode3_nresolution=7
capability_videoin_c0_mode3_resolution='176x144,384x216,640x360,1280x720,1360x768,1600x904,
1920x1080''
capability_videoin_c0_mode3_maxframe='30,30,30,30,30,30,30'
capability_videoin_c0_mode3_maxfps_mpeg4='30,30,30,30,30,30,26'
capability_videoin_c0_mode3_maxfps_mjpeg='30,30,30,30,30,30,30'
capability_videoin_c0_mode3_maxfps_h264='30,30,30,30,30,30,30'
capability_videoin_c0_mode3_description='1080P Full HD (16:9) (MAX 30fps)'
capability_videoin_c0_mode4_effectivepixel='2560x1440'
capability_videoin_c0_mode4_outputsize='1280x720'
capability_videoin_c0_mode4_binning=1
capability_videoin_c0_mode4_nresolution=4
```

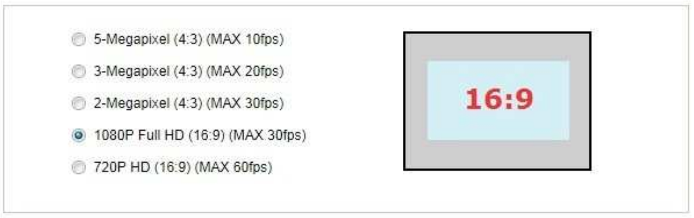
```
capability_videoin_c0_mode4_resolution='176x144,384x216,640x360,1280x720'
capability_videoin_c0_mode4_maxframe='60,60,60,60'
capability_videoin_c0_mode4_maxfps_mpeg4='60,60,60,60'
capability_videoin_c0_mode4_maxfps_mjpeg='60,60,60,60'
capability_videoin_c0_mode4_maxfps_h264='60,60,60,60'
capability_videoin_c0_mode4_description='720P HD (16:9) (MAX 60fps)'
```

The information displayed includes the following:

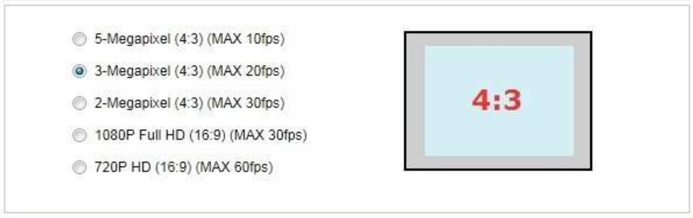
1. FD8372 has only 1 channel. The channel has 5 video modes, and its maximum resolution (full view) is 2560x1920.
2. Mode 0 provides full-view 2560x1920 resolution. The source outputs video data in 2560x1920 resolution at 13fps. This device can encode and stream 2560x1920 MJPEG-encoded video at 13fps, but the frame rate is reduced to 10 fps if the video is encoded with H.264. MPEG4 cannot encode video at 2560x1920 and 2048x1536 resolutions on this device due to hardware limitations.
3. In mode 1, the source outputs video data in 2048x1536 resolution at 20fps. The visible area is the central 2048x1536 portion of the full view. This device can encode and stream 2048x1536 video using any codec except MPEG4 at 20 fps. MPEG4 encoding on this device does not support 2048x1536.
4. In mode 2, the source outputs video data in 1600x1200 resolution at 30fps. The visible area is the central 1600x1200 portion of the full view. This device can encode and stream 1600x1200 video using any codec at 30 fps.
5. In mode 3, the source outputs video data in 1920x1080 resolution at 30fps. The visible area is the central 1920x1080 portion of the full view. This device can encode and stream 1920x1080 video using any codec except MPEG4 at 30 fps. MPEG4 can only achieve 26 fps.
6. In mode 4, the source outputs video data in 1280x720 resolution at 60fps. However, the video data is taken from the central area of the full view, which is at 2560x1440 resolution, and processed via binning. That is, the horizontal view of mode 4 is wider than modes 1 to 3 although its resolution is lower.
7. The figure below illustrates various video modes, with the blue portion indicating the area that is visible.



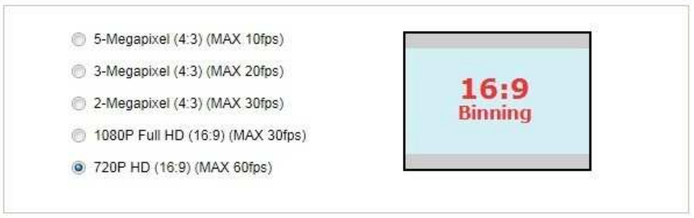
[✓ 5-Megapixel (4:3) (MAX 10fps)]
3-Megapixel (4:3) (MAX 20fps)
2-Megapixel (4:3) (MAX 30fps)
1080P Full HD (16:9) (MAX 30fps)
720P HD (16:9) (MAX 60fps)



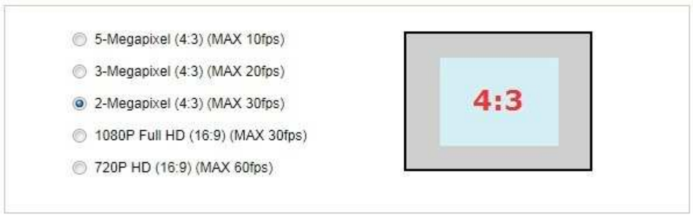
5-Megapixel (4:3) (MAX 10fps)
3-Megapixel (4:3) (MAX 20fps)
2-Megapixel (4:3) (MAX 30fps)
[✓ 1080P Full HD (16:9) (MAX 30fps)]
720P HD (16:9) (MAX 60fps)



5-Megapixel (4:3) (MAX 10fps)
[✓ 3-Megapixel (4:3) (MAX 20fps)]
2-Megapixel (4:3) (MAX 30fps)
1080P Full HD (16:9) (MAX 30fps)
720P HD (16:9) (MAX 60fps)



5-Megapixel (4:3) (MAX 10fps)
3-Megapixel (4:3) (MAX 20fps)
2-Megapixel (4:3) (MAX 30fps)
1080P Full HD (16:9) (MAX 30fps)
[✓ 720P HD (16:9) (MAX 60fps)]



5-Megapixel (4:3) (MAX 10fps)
3-Megapixel (4:3) (MAX 20fps)
[✓ 2-Megapixel (4:3) (MAX 30fps)]
1080P Full HD (16:9) (MAX 30fps)
720P HD (16:9) (MAX 60fps)

4.2.3. WebAPI: Changing Video Modes

4.2.3.1. WebAPI

The video mode is channel-based, and the control API is shown below:

Group: videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
mode	0 ~ "capability_videoin_c<n>_nmode"-1	0 (Full scene)	4/4	Set video mode.

4.2.3.2. Affected Parameters

When the value of "videoin_c<n>_mode" is modified, some other parameters must be modified as well because the associated coordinate system (resolution) is changed. The affected parameters are listed below:

n = channel index, m = stream index, p = mode index .

Name	Effect
capability_videoin_c<n>_mode	This is set to the same value with "videoin_c<n>_mode" when the mode is changed done.
capability_videoin_c<n>_nresolution capability_videoin_c<n>_resolution	They are updated to fit current mode.
capability_videoin_c<n>_maxframerate capability_videoin_c<n>_mpeg4_maxframerate capability_videoin_c<n>_mjpeg_maxframerate capability_videoin_c<n>_h264_maxframerate	They are updated to fit current mode with correction by power line frequency (Please refer to Section 4.6 Power Line Frequency).
Group: roi_c<n>,	All stream ROIs in the channel are set to the largest area ("outputsize") of the new mode. Besides, the referring coordinate system is changed to the output size of current mode.
Group: motion_c<n>	All motion parameters of the channel are cleaned to 0 or <Null>.
Group: privacymask_c<n>	All privacy parameters of the channel are cleaned to 0 or <Null>.
Group: focuswindow_c<n>	TODO
Group: exposurewin_c<n>	TODO
Group: eptz_c<n>	All ePTZ parameters of the channel are cleaned to 0 or <Null>.

Name	Effect
videoin_c<n>_s<m>_resolution	Streams' resolutions are changed to default values to fit the new mode. The policy for default values, please refer 5.1.2Default Resolution for Streams
videoin_c<n>_s<m>_h264_maxframe	Frame rate for H.264 is changed to the default value listed in "capability_videoin_c<n>_mode<p>_maxfps_h264" with correction by power line frequency (Please refer to Section 4.6 Power Line Frequency). The default value is based on stream's resolution ("videoin_c<n>_s<m>_resolution"). Ex: capability_videoin_c0_mode1_resolution='320x240,640x480 ' capability_videoin_c0_mode1_maxfps_h264='60,30' If a stream is 640x480, this parameter is set to 30 fps. If a stream is 320x240, this parameter is set to 60 fps.
videoin_c<n>_s<m>_h264_bitrate	Bitrates for H.264 is changed to a default value. The default value is chosen by resolution and frame rate. More details, please refer 5.5.2 Default H.264 bit rates.
videoin_c<n>_s<m>_mpeg4_maxframe	Frame rate for MPEG4 is changed to the default value listed in "capability_videoin_c<n>_mode<p>_maxfps_mpeg4" with correction by power line frequency (Please refer to Section 4.6 Power Line Frequency). The default value is based on stream's resolution ("videoin_c<n>_s<m>_resolution"). Ex: capability_videoin_c0_mode1_resolution='320x240,640x480 ' capability_videoin_c0_mode1_maxfps_mpeg4='60,30' If a stream is 640x480, this parameter is set to 30 fps. If a stream is 320x240, this parameter is set to 60 fps.
videoin_c<n>_s<m>_mpeg4_bitrate	Bitrates for MPEG4 is changed to a default value. The default value is chosen by resolution and frame rate. More details, please refer 5.6.2 Default MPEG4 bit rates.

Name	Effect
videoin_c<n>_s<m>_mjpeg_maxframe	Frame rate for MJPEG is changed to the default value listed in "capability_videoin_c<n>_mode<p>_maxfps_mjpeg" with correction by power line frequency (Please refer to Section 4.6 Power Line Frequency). The default value is based on stream's resolution ("videoin_c<n>_s<m>_resolution"). Ex: capability_videoin_c0_mode1_resolution='320x240,640x480' capability_videoin_c0_mode1_maxfps_mjpeg='60,30' If a stream is 640x480, this parameter is set to 30 fps. If a stream is 320x240, this parameter is set to 60 fps.
videoin_c<n>_s<m>_mjpeg_bitrate	Bitrates for MJPEG is changed to a default value. The default value is chosen by resolution and frame rate. More details, please refer 5.7.2 Default MJPEG bit rates

4.2.3.3. Usage: Changing Video Modes

Changing modes is a complex action due to the many parameters affected and many internal actions involved. Therefore, we strongly recommend changing modes in a single CGI request, and not combining it with other actions.

Moreover, to ensure changing modes is an atomic operation, but CGI responses are not blocked while the operation is underway, we have designed an asynchronous process, as described below:

1. Change the video mode with a single CGI request:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_mode=<Integer>
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_mode=1

Returns:

```
videoin_c0_mode='1'
```

2. Check the value of "capability_videoin_c<n>_mode".
   The CGI request is:

```
http://<IP>/cgi-bin/viewer/getparam.cgi?capability_videoin_c<n>_mode
```

Ex:

http:// <IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mode

Returns:

```
capability_videoin_c0_mode='1'
```

Note:

"capability_videoin_c<n>_mode" is modified after all actions involved in changing the video mode have been completed. If the returned value is the same as the configured one, then the mode change is complete.

4.2.4. Fisheye Products without Video Mode

Fisheye products with a value of [httpversion] less than 0300 do not implement video mode. Only limited information is provided. Following are some examples to illustrate.

A 5MP fisheye camera provides information such as the following:

```
capability_videoin_resolution='192x192,256x256,384x384,512x512,768x768,1056
x1056,1280x1280,1536x1536,1920x1920'
capability_videoin_resolution1x1='192x192,256x256,384x384,512x512,768x768,1
056x1056,1280x1280,1536x1536,1920x1920'
capability_videoin_resolution16x9='176x144,384x216,640x360,1280x720,1360x7
68,1600x904,1920x1080'
capability_videoin_nresolution='9'
capability_videoin_maxframerate='15,15,15,15,15,15,15,15,15'
capability_videoin_fov='1920x1080,1952x1944'
capability_videoin_mpeg4_maxframerate='15,15,15,15,15,15,15,15,15'
capability_videoin_mjpeg_maxframerate='15,15,15,15,15,15,15,15,15'
capability_videoin_h264_maxframerate='15,15,15,15,15,15,15,15,15'
capability_videoin_svc_maxframerate='15,15,15,15,15,15,15,15,15'
```

For fisheye models without video mode, special parameters have been introduced.

Fisheye models provide a special 1:1 width/height ratio but can also be set to the original 16:9 ratio. “capability_videoin_resolution1x1” denotes resolutions available with 1:1 ratio output and “capability_videoin_resolution16x9” denotes resolutions available with 16:9 output. The fisheye models default to 1:1 ratio resolutions and current resolution options are shown in “capability_videoin_resolution”.

4.3. Cropping

Cropping has been deprecated in the current implementation.

Cropping is a feature to capture only the portion of a device’s full view, removing areas of no interest in order to save bandwidth.

However, ePTZ can also be used to select a region of interest (ROI), making cropping obsolete.

4.3.1. WebAPI

Although cropping has been deprecated, to maintain compatibility, the APIs listed below are available.

Group: videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
options	quality, framerate, crop	quality	4/4	quality: video quality first mode framerate: video frame rate first mode crop: cropping mode
crop_position	<X,Y>	<product dependent>	1/4	Point of left-top corner of cropped area. The unit is pixel. * Width must be multiple of 16 or 32 pixels (by product) and height must be multiple of 8 pixels. * Only valid when "options"='crop'.
crop_size	<WxH>	<product dependent>	1/4	Width and height of cropped area. The unit is pixel. * Width must be multiple of 16 or 32 pixels (by product) and height must be multiple of 8 pixels. * Only valid when "options"='crop'.

The background on cropping is complex. It can be divided into 3 parts.

Part 1:
Cropping first appeared on the IP7161 and was then extended to the IP7361, MD7560, IP8161, IP8361, FD8161, FD8361 with the same functionality. All of these models are equipped with a 2-megapixel sensor. The sensor had 3 working modes: full view at 1600x1200 and 15fps, full view with 2x2 binning and output of 800x600 and 30fps, and any cropped size without binning with variable frame rates. The 3 working modes were mapped one-to-one to the **quality**, **framerate**, and **crop** values for “videoin_c<n>_options".

Everything worked reasonably well at that time except the information in the group, capability. "capability_videoin_resolution" only lists available resolutions in quality mode.
"capability_videoin_maxframerate" lists whichever frame rate is higher, that of the quality mode or that of the framerate mode.

Part 2:

We included cropping in our 5 MP products such as the IP8172, IP8372, and FD8372 as well. With 5 MP sensors, the increased number of pixels allows additional working modes. Therefore, the quality and framerate options for "videoin_c<n>_options" have been eliminated, and only the crop option has been retained. There is a total of 5 working modes for a 5 MP sensor: 2560x1920@13fps, 2048x1536@20fps, 1600x1200@30fps, 1920x1080@30fps, and 1280x720@60fps. However, the usage of cropping APIs does not conform to the original design.

First, only the 5 sizes listed above are working correctly. Other sizes can be accepted, but they may result in incorrect display of video (in more recent firmware versions, the options for "videoin_c<n>_crop_size" have been restricted to valid sizes only).

Second, the 1280x720 option does not result in a 1280x720 area of the screen being cropped from the full-view 2560x1920. Instead an 2560x1440 area is cropped and then is binned down to 1280x720 for output. As a result the 1280x720 actually provides a larger field-of-view (FOV) than the 1600x1200 and 1920x1080 options, though this is not apparent from the interface. In addition, there is no way to crop a 1280x720 area without binning.

A new parameter, “capability_fov”, was created to store working crop sizes. The parameter was, however, easily misinterpreted and made integration difficult, violating the common understanding that FOV is measured in degrees, not pixels.

For these crop sizes, 2 more parameters were added:

"capability_videoin_resolution4x3" stores available resolution options for streams when the cropped size is 2560x1920, 2048x1536, or 1600x1200.

"capability_videoin_resolution16x9" stores available resolution options for streams when the cropped size is 1920x1080 or 1280x720.

Part 3:
Our 5 MP fisheye camera also uses the method described in part 2. The difference is that there are only 2 working crop sizes, 1952x1944 and 1920x1080, listed in "capability_fov".

As described in part 2, there are parameters to show available resolutions:

"capability_videoin_resolution1x1" stores available resolution options for streams when the cropped size is 1952x1944.

"capability_videoin_resolution16x9" stores available resolution options for streams when the cropped size is 1920x1080.

These APIs have the problems described in part 2 and make integration very difficult.

Summary:
We are aware of cropping-related issues, and we have received complaints from customers about difficulties in integrating our 5 MP product. As cropping’s purpose—selecting a limited area of the full view—can also be accomplished via ePTZ’s region of interest (ROI) feature, we have deprecated cropping and replaced it with ePTZ and a newly designed video mode described in Section 4.2, “Video mode”. Video

mode, also called sensor mode, can fully describe the working modes of sensors in an easily understood, logical manner.

In newer models, "capability_fov", "capability_videoin_resolution4x3", "capability_videoin_resolution16x9", "capability_videoin_resolution1x1" and cropping-related APIs are no longer supported.

4.3.2. Affected Parameters

With the exception of fisheye cameras, a product that supports cropping will also support ePTZ. However, the reverse is not true, as cameras that support ePTZ will not necessarily support cropping. When cropping-related APIs are modified, the following APIs are affected:

n = channel index, m = stream index.

Name	Effect
Group: roi_c<n>,	All ROIs of the channel are set to the cropped size. Besides, the referring coordinate system is changed to the cropped size.
Group: motion_c<n>	All motion parameters of the channel are cleaned to 0 or <Null>.
Group: privacymask_c<n>	All privacy parameters of the channel are cleaned to 0 or <Null>.
Group: focuswindow_c<n>	TODO
Group: exposurewin_c<n>	TODO
Group: eptz_c<n>	All ePTZ parameters of the channel are cleaned to 0 or <Null>.
videoin_c<n>_s<m>_resolution	Streams' resolutions are basically changed to cropped size.
videoin_c<n>_s<m>_h264_maxframe	Frame rate for H.264 is changed according to cropped size.
videoin_c<n>_s<m>_h264_bitrate	Bitrates for H.264 is changed according to cropped size.
videoin_c<n>_s<m>_mpeg4_maxframe	Frame rate for MPEG4 is changed according to cropped size.
videoin_c<n>_s<m>_mpeg4_bitrate	Bitrates for MPEG4 is changed according to cropped size.
videoin_c<n>_s<m>_mjpeg_maxframe	Frame rate for MJPEG is changed according to cropped size.
videoin_c<n>_s<m>_mjpeg_bitrate	Bitrates for MJPEG is changed according to cropped size.

4.3.3. Usage: Configuring Cropping

The usage is also obsolete for the reasons previously discussed.

If you have already integrated cropping with some models, you can continue to use it, as we will maintain cropping-related APIs on those models until they are phased out.

If you have not integrated cropping yet, for newer models (those whose [httpversion] value is greater than or equal to 0300), we recommend using the video mode described in Section 4.2 Video Mode.

4.4. Video Orientation

Orientation is a feature to flip or mirror the video to suit a camera's installed position.

4.4.1. WebAPI

All VIVOTEK products support video flipping and mirroring. The orientation APIs are as follows:

Group: videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
flip	0, 1	0	4/4	0: Non-flip the image. 1: Flip the image.
mirror	0, 1	0	4/4	0: Non-mirror the image. 1: Mirror the image.

4.4.2. Affected Parameters

Changing orientation APIs affects the following APIs because of the change in the coordinate space of the video:

Name	Effect
Group: eptz_c<n>	All ePTZ parameters of the channel are cleaned to 0 or <Null>.

4.4.3. Usage: Changing Orientation

Set flipping:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_flip=<0,1>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_flip=1

Returns:

```
videoin_c0_flip='1'
```

Set mirroring:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_mirror=<0,1>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_mirror=1

Returns:

```
videoin_c0_mirror='1'
```

4.5. Privacy mask

Privacy mask is the function to mask the selected area (masked window) in image.
There are two types: 2D and 3D. The type can be known by "capability_ privacymask_axistype". The value 1 means 2D mask, and the value 2 means 3D mask.

3D privacy mask is often applied on speed-dome and PTZ camera. The masked window is kept on the absolute position while the camera is moving. That is, if a masked window covers a door, than the window keeps hiding the door no matter the camera moves up, down, or anywhere. The control method of 3D privacy mask is very different with 2D mask. To control 3D privacy mask, please refer the document: VivotekCameraWebAPI_PTZCTRL.doc

This section only contains APIs for 2D privacy mask.

4.5.1. WebAPI

Privacy mask has 2 kinds of window shapes: rectangle and polygon.
Traditionally, privacy mask window is rectangle. Polygon window is implemented for fisheye camera. Polygon window has more flexibility than rectangle window and it can be applied on all cameras. However, we still prefer rectangle window on non-fisheye camera for better performance.

"capability_privacymask_shape" shows the supported window shape. For a device, only one shape is supported.
* Note: Group: capability_privacymask is only available when [httpversion] >= 0301. For most products, 2D rectangle privacy mask is applied.

APIs for rectangle and polygon window are different, too. They are listed as following:

Rectangle:

Group: privacymask_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
enable	0, 1	0	4/4	The main switch of all privacy mask windows. 0: Disable 1: Enable

Group: privacymask_c<n>_win_i<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".
m denotes the window index, with a range from 0 to one less than the value of "capability_privacymask_num".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
enable	0, 1	0	4/4	0: Disable this privacy mask window 1: Enable this privacy mask window
name	String[14]	<blank>	4/4	Name of this privacy mask window
left	0~319	0	4/4	Left coordinate of window position.
top	0~239	0	4/4	Top coordinate of window position.
width	0~319	0	4/4	Width of privacy mask window.
height	0~239	0	4/4	height of privacy mask window.

For historical reasons, 2D privacy mask window is set in QVGA (320x240) domain...
The QVGA domain is mapped to the current image. Internal module will transform the settings to the actual position and size in image. "left" and "top" mean the start point of a window. "width" and "height" mean the size of a window. For example, (left, top, width, height)= (0,0,159,119) means left-top quarter of the image no matter the image size is 1920x1080, 2560x1600 or others.

Polygon:

Group: privacymask_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
enable	0, 1	0	4/4	The main switch of all privacy mask windows. 0: Disable 1: Enable

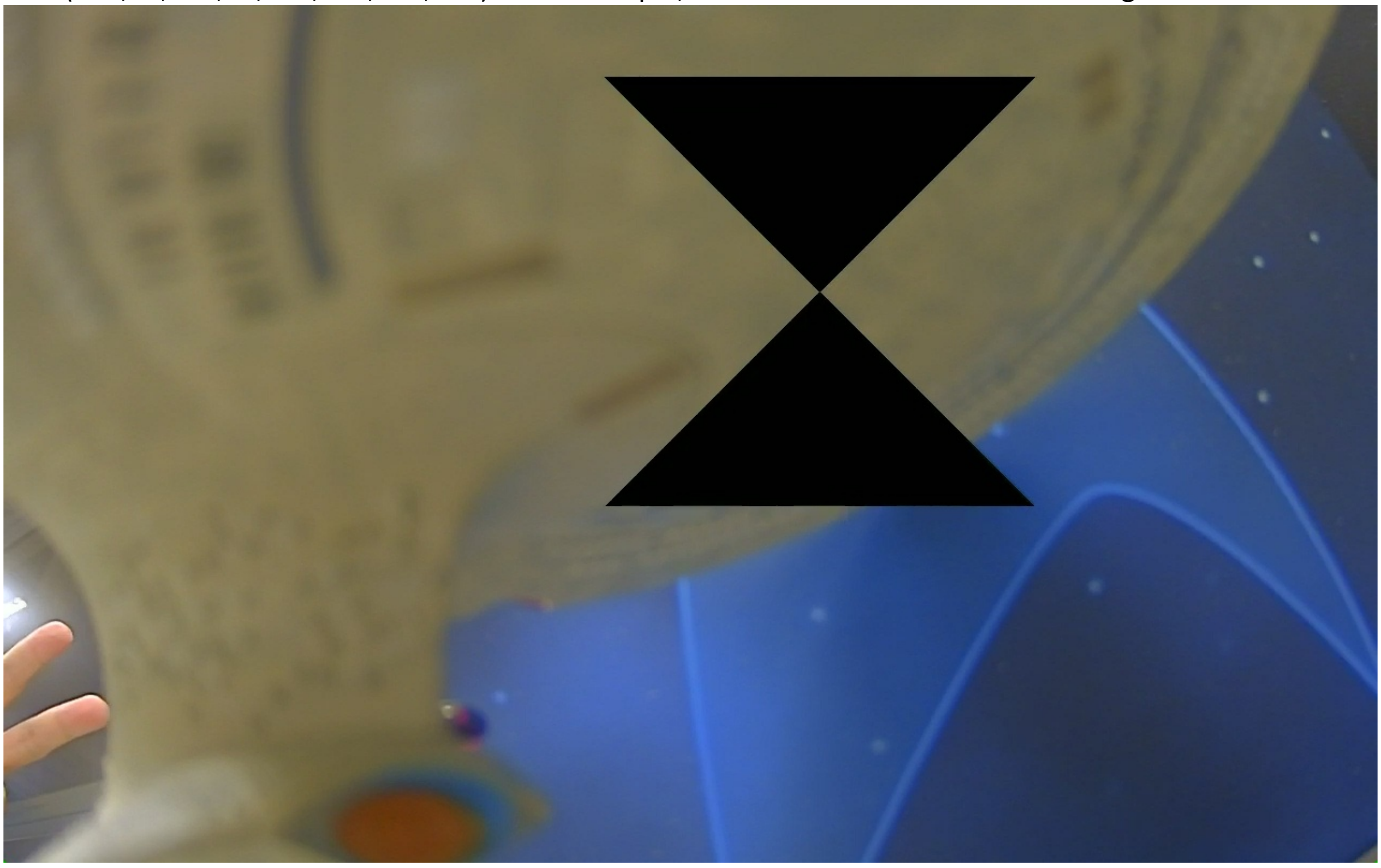
Group: privacymask_c<n>_win_i<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".
m denotes the window index, with a range from 0 to one less than the value of "capability_privacymask_num".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
enable	0, 1	0	4/4	0: Disable this privacy mask window 1: Enable this privacy mask window
name	String[14]	<blank>	4/4	Name of this privacy mask window
polygon	<0~319,0~239, 0~319,0~239, 0~319,0~239, 0~319,0~239> * 4 points, total 8 integers.	<blank>	4/4	Coordinate of polygon window points. The format is X_0,Y_0,X_1,Y_1,X_2,Y_2,X_3,Y_3

	Integers are split by comma.			

Like rectangle, polygon privacy mask window is set in QVGA (320x240) domain, too...
The QVGA domain is mapped to the current image. Internal module will transform the settings to the actual position and size in image.

The order of points is essential. There are 4 virtual lines: Point0 to Point1, Point1 to Point2, Point2 to Point3, and Point3 to Point0. Internal module finds the inner area closed by lines as window.
Take (140,20,240,20,140,140,240,140) as an example, the masked window is as following:



4.5.2. Affected Parameters

None for WebAPI.

However, privacy masks consume hardware resources, and this affects the product's performance. For example, the encoded frame rate may be reduced, video latency may be increased, etc. The impact depends on the number of masks present and the size of the masked area.

4.5.3. Usage: Setting a Privacy Mask

Enable all privacy mask windows:

```
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c<n>_enable=1
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c0_enable=1

Returns:

```
privacymask_c0_enable='1'
```

Disable all privacy mask windows:

```
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c<n>_enable=0
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c0_enable=0

Returns:

```
privacymask_c0_enable='0'
```

Setup a rectangle masked window and enable it:

```
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c<n>_win_i<m>_name=<String[14]>&privacymask_c<n>_win_i<m>_left=<0~319>&privacymask_c<n>_win_i<m>_top=<0~239>&privacymask_c<n>_win_i<m>_width=<0~319>&privacymask_c<n>_win_i<m>_height=<0~239>&privacymask_c<n>_win_i<m>_enable=1
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c0_win_i0_name=firstmask&privacymask_c0_win_i0_left=0&privacymask_c0_win_i0_top=0&privacymask_c0_win_i0_width=159&privacymask_c0_win_i0_height=119&privacymask_c0_win_i0_enable=1

Returns:

```
privacymask_c0_win_i0_name='firstmask'
privacymask_c0_win_i0_left='0'
privacymask_c0_win_i0_top='0'
privacymask_c0_win_i0_width='159'
privacymask_c0_win_i0_height='119'
privacymask_c0_win_i0_enable='1'
```

Setup a polygon masked window and enable it:

```
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c<n>_win_i<m>_name=<String[14]>&privacymask_c<n>_win_i<m>_polygon=<0~319,0~239, 0~319,0~239, 0~319,0~239, 0~319,0~239>
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c0_win_i0_name=polygonmask&privacymask_c0_win_i0_polygon=140,20,240,20,140,140,240,140&privacymask_c0_win_i0_enable=1

Returns:

```
privacymask_c0_win_i0_enable='1'
privacymask_c0_win_i0_name='polygonmask'
privacymask_c0_win_i0_polygon='140,20,240,20,140,140,240,140'
```

Disable a privacy mask windows:

```
http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c<n>_win_i<m>_enable=0
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?privacymask_c0_win_i0_enable =0

Returns:

```
privacymask_c0_win_i0_enable='0'
```

4.6. Power Line Frequency

CMOS frequency is used to resolve the problem of flicker resulting from fluorescent lamps (for more information, see http://en.wikipedia.org/wiki/Fluorescent_lamp#Flicker_problems). Setting the CMOS frequency to match the frequency of the power lines in a user's location can help eliminate flicker.

4.6.1. WebAPI

The ability to set the power line frequency is only available on cameras with a value of 2 for "capability_video_type". The API is described below:

Group: videoin_c<n>,

n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

* Only available if the value of "capability_video_type" is 2.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
cmosfreq	50, 60	60	4/4	Set CMOS frequency. The unit is Hz. * Only available when "capability_videoin_type"=2

4.6.2. Affected Parameters

Power line frequency affects exposure time and subsequently frame rates for a given video source. For example, a sensor might output 60 fps at 60 Hz, but output 50 fps at 50 Hz.

If the value of "videoin_c<n>_cmosfreq" is changed to 50, then the affected parameters are as shown below:

Name	Effect
capability_videoin_c<n>_maxframerate	If maximum fps is 30 fps, then 26~30 fps will be changed to 25 fps. If maximum fps is 60 fps, then 51~60 fps will be changed to 50 fps. If maximum fps is 120 fps, then 101~120 fps will be changed to 100 fps.
capability_videoin_c<n>_mpeg4_maxframerate	
capability_videoin_c<n>_mjpeg_maxframerate	
capability_videoin_c<n>_h264_maxframerate	
videoin_c<n>_s<m>_h264_maxframe	
videoin_c<n>_s<m>_mpeg4_maxframe	
videoin_c<n>_s<m>_mjpeg_maxframe	

Image-related parameters are also affected. For details, please refer to the document, “VIVOTEKWebAPI_Image.doc”.

4.6.3. Usage: Changing Power Line Frequency

Change power line frequency to 60:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_cmosfreq=60
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_cmosfreq=60

Returns:

```
videoin_c0_cmosfreq='60'
```

Change power line frequency to 50:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_cmosfreq=50
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_cmosfreq=50

Returns:

```
videoin_c0_cmosfreq='50'
```

4.7. Modulation

CCTV cameras capture video in either of two formats: NTSC or PAL. When a video server converts analog video from a CCTV camera into digital data, the server's modulation setting must match the format of the installed CCTV camera's video.

4.7.1. WebAPI

Only the VS8102 provides this API.

Normally, the VS8102 can auto-detect the video format of the installed CCTV camera. However, if the detected format is wrong, the user can set it manually.

Newer video servers only support auto-detection, so the API is not provided.

Group: videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin"**.**

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
modulation	ntsc, pal, auto	auto	4/4	Set modulation type of input. * This API is available on VS8102.

4.7.2. Affected Parameters

The modulation type affects the frame rate. NTSC has maximum of 30 fps and PAL has maximum of 25 fps.

If the value of "videoin_c<n>_modulation" is set to pal, then the affected parameters are as shown below:

Name	Effect
videoin_c<n>_s<m>_h264_maxframe	For PAL, maximum frame rate is 25 fps.
videoin_c<n>_s<m>_mpeg4_maxframe	
videoin_c<n>_s<m>_mjpeg_maxframe	

Image-related parameters are also affected. For details, please refer to the document "VIVOTEKWebAPI_Image.doc".

4.7.3. Usage: Changing Modulation

Change modulation to auto-detect:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_modulation=auto
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_modulation=auto

Returns:

```
videoin_c0_modulation='auto'
```

Change modulation to NTSC:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_modulation=ntsc
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_modulation=ntsc

Returns:

```
videoin_c0_modulation='ntsc'
```

Change modulation to PAL:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_modulation=pal
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_modulation=pal

Returns:

```
videoin_c0_modulation='pal'
```

4.8. Color

4.8.1. WebAPI

All VIVOTEK IP cameras and video servers can set video to display in color or in monochrome. The API is described below:

Group: videoin_c<n>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
Color	0, 1	1	4/4	0 => monochrome 1 => color

4.8.2. Affected Parameters

None.

4.8.3. Usage: Setting Video Display to Color or Monochrome

Set video display to color:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_color=1
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_color=1

Returns:

```
videoin_c0_color='1'
```

Set video display to monochrome:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_color=0
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_color=0

Returns:

```
videoin_c0_color='0'
```

4.9. Text-on-video

Text-on-video is a useful feature to overlay the time and other information such as location and names on an image. The overlay action is performed after any ISP operations and privacy masks are applied.

4.9.1. WebAPI

The basic APIs are described below:
Group: videoin_c<n>,
where n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
imprinttimestamp	0, 1	0	4/4	0 => Disable 1 => Embed time and text on image.
text	String[16] for 7000-series. String[60] for 8000-series.	<blank>	1/4	Text string to be embedded on image. * Only valid when "imprinttimestamp"=1

When text-on-video is enabled, text and time information is displayed in the upper left corner of the video. The format is text + time. The time represents the time when capture of the frame was completed by the camera or video server. Note that if the width of the streamed video is less than 320 pixels, text information will not be displayed, but only time information.

For some new products, we enhance the function of text on video by customer's request. After [httpversion] >= 0301, a group: capability_textonvideo is designed for marking these enhanced features. Please refer: 3 Device Capabilities API

APIs for enhanced text on video are as following:

Group: videoin_c<n>_textonvideo,
where n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
position	top, bottom * Available options are listed in "capability_textonvideo_position"	top	4/4	top = Embed text and time information on left-top corner. bottom = Embed text and time information on left-down corner. * Only valid when

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
				"videoin_c<n>_imprinttimestamp"=1
size	15, 25, 30 * Available options are listed in "capability_textonvideo_size"	15	4/4	Available font size (px) for text on video. * Only valid when "videoin_c<n>_imprinttimestamp"=1

4.9.2. Affected Parameters

None.

4.9.3. Usage: Configuring Text-on-video

Enable text-on-video:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_imprinttimestamp=1
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_imprinttimestamp=1

Returns:

```
videoin_c0_imprinttimestamp='1'
```

Disable text-on-video:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_imprinttimestamp=0
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_imprinttimestamp=0

Returns:

```
videoin_c0_imprinttimestamp='0'
```

Set text:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_text=<String[16] or
String[60]>
```

Ex: http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_text=IamTextonVideo

Returns:

```
videoin_c0_text='IamTextonVideo'
```

Set text position:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_textonvideo_position=<top,bottom>
```

Ex: http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_textonvideo_position=bottom

Returns:

```
videoin_c0_textonvideo_position='bottom'
```

Set text font size:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_textonvideo_size=<15,25,30>
```

Ex: http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_textonvideo_size=30

Returns:

```
videoin_c0_textonvideo_size='30'
```

5. Stream Domain API

5.1. Resolutions for CMOS Cameras

Each stream has a particular resolution, and regardless of any ROI that has been set, channel data is scaled to this resolution.

This section applies only to those products for which the value of "capability_videoin_type" is 2. For products for which the value of "capability_videoin_type" is 0 or 1, please refer to Section 5.2 Resolution for Video Servers and Cameras with a CCD Module.

5.1.1. WebAPI

The API for stream resolution is described below:

Group: videoin_c<n>_s<m>,
    n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
    m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available if the value of "capability_videoin_type" is 2.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
resolution	Listed in "capability_videoin_c<n>_resolution"	Please refer: 5.1.2 Default Resolution for Streams	4/4	Video resolution in pixels for a stream.

A stream's maximum resolution is limited to the maximum resolution of the channel. The maximum resolution of a channel depends on the video mode currently in use.

At the WebAPI level, any size that exceeds the maximum resolution of all video modes in a channel is rejected. However, a stream's resolution can be set to a value higher than the maximum resolution of the channel (when using a video mode with lower resolution) by an internal module that takes the maximum resolution of the channel as the stream's resolution. In this case, no notification is provided and no change is made to the stream's resolution at the WebAPI level.

*** Note:**
On high-megapixel cameras, usable resolutions may be limited by the particular encoding module. For example, most VIVOTEK cameras have a 1920-pixel width limitation when the MPEG4 codec is used (though some models support a 2048-pixel width with MPEG4).

Therefore, we have exposed the encoded frame rates for each resolution with each codec in "capability_videoin_c<n>_<codectype>_maxframe". Please refer to Section 4.2.1 WebAPI: Channel-specific Information. When a resolution's frame rate is recorded as "-", this indicates that the particular codec does not support this resolution.

For convenience, the unsupported resolutions can be still set. The internal module will change the unsupported width or height to the maximum value supported by the codec without notification and without changing the stream's resolution at the WebAPI level. However, we do not recommend using this approach.

5.1.2. Default Resolution for Streams

This section describes the policy for determining a default resolution for a stream. The default resolutions are used in factory default settings. In addition, when changing a channel's video mode, the policy is also used. Lower numbers indicate higher priority. Note that default values are subject to change.

1. The first stream (videoin_c<n>_s0) always uses the maximum resolution of the new mode.

2. The last stream uses "176x144" for mobile devices if the stream number is 2.
   The next-to-last stream uses "176x144" for mobile devices if the stream number is greater than or equal to 3.

3. The last stream uses the maximum resolution of the new mode if the stream number is greater than or equal to 3.

4. Stream 2 (videoin_c<n>_s1) uses the second-highest resolution listed in "capability_videoin_c<n>_mode<p>_resolution", and so on for the 3rd, 4th, and further streams.

Ex: IP8372 has 4 streams per channel, and uses all 4 policies, whereas IP8331 has 2 streams per channel, and uses only policies 1 & 2.

5.1.3. Affected Parameters

None.

5.1.4. Usage: Changing Stream Resolution

Two methods are described below.

In the examples, we use stream 0 of channel 0 as the target.

Simple method:

1. Get available resolutions from the group capability_videoinc_<n>:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c<n>_resolution
```

Ex:

http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_resolution

Returns:

```
capability_videoin_c0_resolution='320x240,640x480,2048x1536'
```

2. Set the stream's resolution

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_resolution=<WxH>
```

Ex:

http:// <IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_resolution="2048x1536"

Returns:

```
videoin_c0_s0_resolution ='2048x1536'
```

More sophisticated method:

1. Get the stream's codec type:

```
http://<IP>/cgi-bin/admin/getparam.cgi?videoin_c<n>_s<m>_codectype
```

Ex:

http://<IP>/cgi-bin/admin/getparam.cgi?videoin_c*0*_s0_codectype

Returns:

```
videoin_c0_s0_codectype='mpeg4'
```

2. Get available resolutions from the group capability_videoinc_<n>:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c<n>_resolution
```

Ex:

http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_resolution

Returns:

```
capability_videoin_c0_resolution='320x240,640x480,2048x1536'
```

3. Get available frame rate from the group capability_videoinc_<n>:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c<n>_<codectype>_maxframerate
```

Ex:

http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mpeg4_maxframerate

Returns:

```
capability_videoin_c0_mpeg4_maxframerate='30,30,-'
```

4. Set the stream's resolution to a supported value:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_resolution=<WxH>
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_resolution="640x480"

Returns:

```
videoin_c0_s0_resolution='640x480'
```

In this case, 2048x1536 is not supported by MPEG4. Therefore, the 2nd option, 640x480, is taken.

5.2. Resolution for Video Servers and Cameras with a CCD Module

5.2.1. WebAPI

The resolution options for video servers and cameras with a CCD module ("capability_videoin_type"= 0 or 1) are very different from those for other devices. Conventional formats such as D1, 4CIF, QCIF, and CIF are used. The benefit of using these formats is that they conceal the difference in resolution between NTSC and PAL. The following table maps these formats to the resolutions they provide:

Format	Resolution
D1	NTSC: 720x480 PAL: 720x576
4CIF	NTSC: 704x480 PAL: 704x576
CIF*	NTSC: 352x240 PAL: 352x288
QCIF	NTSC: 176x120 PAL: 176x144

* Note: For CIF, we have actually used the definition of SIF. For reference, see http://en.wikipedia.org/wiki/Common_Intermediate_Format.

The API for stream resolution is described below.
Group: videoin_c<n>_s<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available if the value of "capability_videoin_type" is 0 or 1.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
resolution	D1, 4CIF, CIF, QCIF	4CIF	4/4	Video resolution for a stream.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
ratiocorrect	0, 1	0	1/4	Change resolution to fit ratio 4:3. For PAL: D1/4CIF(720/704x576) -> (768x576) CIF(352x288)->(384x288) For NTSC: D1/4CIF(720/704x480) -> (640x480) CIF(352x240)->(320x240) * This is not working on QCIF. * Only available on VS8801, VS8401.

NTSC or PAL:
Video servers and CCD cameras use different approaches for recognizing whether a channel is NTSC or PAL.

Video servers:
For video servers, the installed CCTV camera determines whether video is in the NTSC or PAL format. For multi-channel video servers, each channel can have its own modulation type.

The API to get the current modulation type is shown below:

Product name	Current modulation type
VS8102	NTSC: "status_videomode_c0"=ntsc PAL: "status_videomode_c0"=pal
VS8401, VS8801	NTSC: "status_videomode_c<n>"=ntsc PAL: "status_videomode_c<n>"=pal Where n denotes the channel index , range is from 0 to "capability_nvideoin"-1,

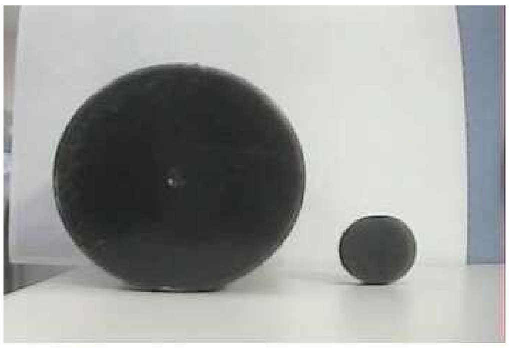
CCD cameras:
For CCD cameras, the installed CCD module determines the modulation type. The CCD module is installed at the time of manufacture, and modulation type is fixed at that time.

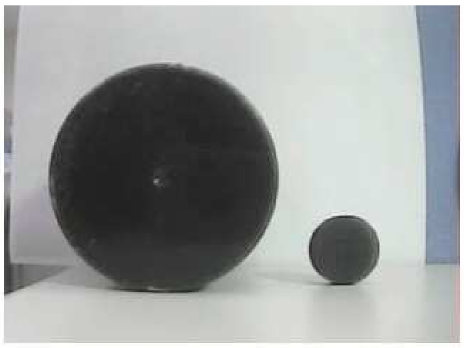
The API to get the modulation type of a CCD camera is as shown below:

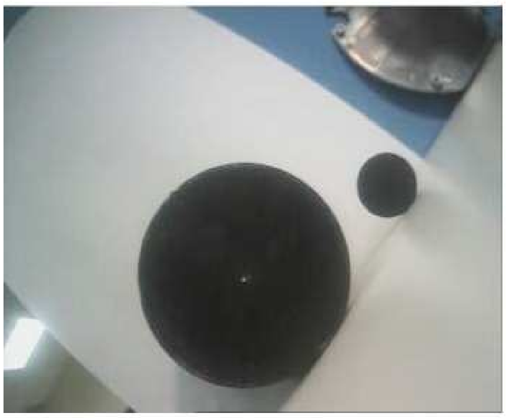
Product name	Current modulation type
PZ81x1, PZ81x1W	NTSC: "status_videomode_c0"=ntsc PAL: "status_videomode_c0"=pal
SD8111, SD8311E, SD8312E, SD8313E	Fixed NTSC: "status_videomode_c0"=ntsc
SD8121, SD8321E, SD8322E, SD8323E	Fixed PAL: "status_videomode_c0"=pal

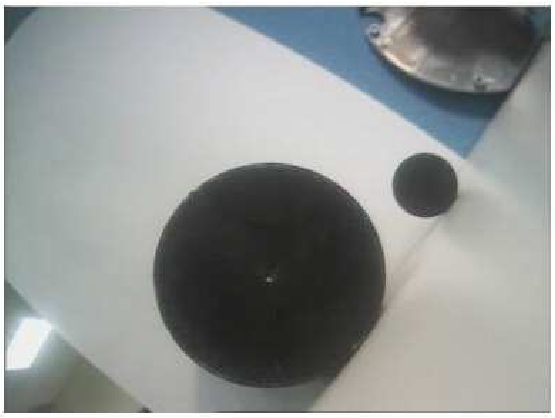
Ratio correction:
Video servers receive analog TV signals from a CCTV camera in one of two formats, NTSC or PAL. Both were designed for old analog televisions, which had a pixel aspect ratio that was not 1:1.

However, modern digital televisions and monitors do have a 1:1 pixel aspect ratio. As a result, displaying NTSC or PAL video on modern digital equipment results in a stretched image. The way to solve this problem is to scale the image to fit a 4:3 aspect ratio. "videoin_c<n>_s<m>_ratiocorrect" is designed to control this ratio correction. Screenshots of video before and after ratio correction are shown below.


NTSC CIF


Ratio correction


PAL CIF


Ratio correction

5.2.2. Affected Parameters

None.

5.2.3. Usage: Changing Stream Resolution

Set stream resolution:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_resolution=<D1,
4CIF, CIF, QCIF>
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_resolution=D1

Returns:

```
videoin_c0_s0_resolution='D1'
```

Enable ratio correction:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_ratiocorrect=1
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_ratiocorrect=1

Returns:

```
videoin_c0_s0_ratiocorrect='1'
```

Disable ratio correction:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_ratiocorrect=0
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_ratiocorrect=0

Returns:

```
videoin_c0_s0_ratiocorrect='0'
```

5.3. Region of Interest

Region of interest (ROI) is a component of ePTZ functionality, and as a result is only available for streams supporting ePTZ. Information about whether or not a given stream supports ePTZ is recorded in the "capability_eptz" parameter. Please refer to Section 3 Device Capabilities API.

With ROI, you can specify a portion of the full view to stream, cropping out unneeded parts, thereby conserving network bandwidth.

There is a use case for mixing ROI and multiple streams: one small stream provides a full view, while one ROI stream that is not scaled down in size covers the door providing access, and another covers the actual target being monitored. Compared to a single full-size stream at the same bit rate, this method applies a high bit rate only on the areas of interest, providing a more detailed view.

5.3.1. WebAPI

Group: roi_c<n>_s<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available when the stream supports ePTZ. See Section 3 Device Capabilities API.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
home	<X,Y>	0,0	6/6	Coordinate of left-top corner of selected area. The coordinate system is based on current maximum resolution of this channel, and the origin is located at left-top corner. The unit is pixel. * X must be multiples of 16 pixels. * Y must be multiples of 8 pixels. * For current maximum size, please refer 4.2 Video Mode.
size	176x144 ~ Current maximum size of this channel. Ex: 176x144~2048x1536, 176x144~2560x1920 * For current maximum resolution, please refer 4.2 Video Mode.	Current maximum size of this channel.	6/6	Width and height of selected area. The unit is pixel. * Width must be multiples of 16 pixels or 32 pixels on some products with MPEG4. We suggest use the common value: 32 pixels. * Height must be multiples of 8 pixels.

Range of acceptable values at the WebAPI level:
For "size", the minimum value is 176x144, while the maximum value is the full-view (see Section 4.2.1) resolution of a channel.

For "home", the minimum value is "0,0". The maximum value of x is the width of the full view minus 176. The maximum value of y is the height of the full view minus 144.

Ex:
IP8371E is a 3 MP camera with a 2048x1536 full-view resolution.
The range of values for "size" is from 176x144 to 2048x1536.
The range of values for "home" is from “0,0" to "1872,1392"

Protection mechanism:
Protection against invalid input values is performed using the following steps:

1. When input values for pixels are not multiples of 8 or 16, they will be changed to the closest value that satisfies this condition in internal modules. Note that no notification is provided and no parameter values are modified in WebAPI.

2. If the value of "size" is larger than the current maximum size of a channel, this maximum size will be used. Note that no notification is provided and no parameter values are modified in WebAPI.

3. The entire ROI area must be located inside the current full-size view of the channel (see section 4.2.1) in order for the ROI setting to be valid.

Because it is difficult to implement a comprehensive protection mechanism at the WebAPI level, an internal video module is used. When an invalid ROI setting is detected, the nearest area with a valid value for "size" as defined in step 2 is used. This adjustment is performed automatically by the internal video module. Note that no notification is provided and no parameter values are modified in WebAPI.

Ex:
IP8371E is a 3 MP camera with a 2048x1536 full-view resolution.
A value of 512x384 for “size” is valid if the value of “home” is 1536,1152.
A value of 512x384 for “size” is not valid if the value of “home” is 1792,1152, and the internal module will change the latter to “1536,1152”.
A value of 512x384 for “size” is not valid if the value of “home” is 1536,1344, and the internal module will change the latter to 1536,1152.

If the IP8371E is using its 2MP mode with 1920x1080 as the maximum resolution, then the information below applies:
A value of 1744,936 for “home” is valid if the value of “size” is 176x144.
A value of 1872,1392 for “home” is not valid if the value of “size” is 176x144, and the internal module will change it to 1744,936.

5.3.2. Affected Parameters

None.
However, we recommend adjusting "videoin_c<n>_s<m>_resolution" to maintain a consistent aspect ratio with the ROI area, avoiding distortion of the video. For example, if the ROI area is 1280x960, the recommended stream resolutions are 1280x960, 640x480, 320x240, and so on, since they all share the same aspect ratio as the ROI area. A circle in these display resolutions will thus be rendered accurately, and not as an ellipse.

It is worth mentioning that when a stream supports ePTZ, it supports more resolution options than those listed in "capability_videoin_c<n>_resolution". The range of values is from 176x144 to the maximum resolution in "capability_videoin_c<n>_resolution", while the width must be a multiple of 16 (or 32) pixels and the height must be a multiple of 8 pixels.

5.3.3. Usage: Changing a Stream's ROI

1. Check if the target stream supports ePTZ:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_nvideoin&capability_eptz
```

Returns:

```
capability_nvideoin='1'
capability_eptz='7'
```

The returned values indicate that this product has one channel and that the 1st to 3rd streams support ePTZ.

2. There are two methods for retrieving the current maximum resolution of a channel as a coordinate system:

2a. Get available resolutions from the group capability_videoinc_<n> and retrieve the last one, which is the maximum resolution:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c<n>_resolution
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_resolution

Returns:

```
capability_videoin_c0_resolution='480x270,960x540,1920x1080'
```

In this case, 1920x1080 is the current maximum resolution of a channel.

2b-1. Get the current mode index of a channel:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c<n>_mode
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mode

Returns:

```
capability_videoin_c0_mode='1'
```

.   2b-2. Use the mode index to get the output size (which is equal to the maximum resolution) of the mode:

```
http://<IP>/cgi-
bin/admin/getparam.cgi?capability_videoin_c<n>_mode<modeindex>_outputsize
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mode1_outputsize

Returns:

```
capability_videoin_c0_mode1_outputsize='1920x1080'
```

3. Set the ROI:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?roi_c<n>_s<m>_home=<X,Y>&roi_c<n>_s<m>_size=<Wx
H>
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?roi_c0_s2_home="480,272"&roi_c0_s2_size="960x544"

Returns:

```
roi_c0_s2_home='480,272'
roi_c0_s2_size='960x544'
```

In this example, we have set the ROI of stream 3 on channel 1 to the central 960x544 area.

5.4. Codec type

Each stream uses a particular codec to encode video data. Available codec types are listed in “capability_videoin_codec".

5.4.1. WebAPI

Group: videoin_c<n>_s<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
codectype	Listed at "capability_videoin_codec" Possible values are: mpeg4, mjpeg, h264	h264 (mpeg4 for old products 7000-series)	1/4	Codec type for this stream.

5.4.2. Affected Parameters

None

5.4.3. Usage: Changing a Stream’s Codec Type

1. Get available codec types:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_codec
```

Returns:

```
capability_videoin_codec='mpeg4,mjpeg,h264'
```

2. Set a stream’s codec type:

```
http://<IP>/cgi-
bin/admin/getparam.cgi?videoin_c<n>_s<m>_codectype=<codectype>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_codectype=h264

Returns:

```
videoin_c0_s0_codectype='h264'
```

5.5. Codec: H264

When a stream's codec type is set to H.264 ("videoin_c<n>_s<m>"= h264), the APIs in this section are valid.

5.5.1. WebAPI

Group: videoin_c<n>_s<m>_h264,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxframe	1~999	The maximum value in "capability_videoin_c<n>_h264_maxframerate"	4/4	The maximum encoded frame rate.
intraperiod	250, 500, 1000, 2000, 3000, 4000	1000	4/4	The time interval between two I-frames (Intra coded picture). The unit is millisecond (ms).
ratecontrolmode	cbr, vbr	cbr	4/4	cbr: Constant bit rate mode. vbr: Fixed quality mode, all frames are encoded in the same quality.
quant	1~5, 99, 100	3	4/4	* Only valid when "ratecontrolmode"= vbr. Set the pre-defined quality level: 1: Median 2: Standard 3: Good 4: Detailed 5: Excellent 100: Use the quality level in "qpercent" 99: Use the quality level in "qvalue"
qpercent	1~100	50	4/4	Select customized quality in a normalized full range. 1: Worst quality 100: Best quality * Only valid when "ratecontrolmode"= vbr and "quant"= 100.
qvalue	<By product> * Not recommended to use this.	<By product>	4/4	The Q value which is used by encoded library directly. The range is different from product to product. We keep this for compatibility and internal use. * It's strongly non-recommended to use this. * Only valid when "ratecontrolmode"= vbr

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
				and "quant"= 99.
bitrate	1000~"capability_videoin_c<n>_h264_maxbitrate"	Please refer 5.5.2 Default H.264 bit rates	4/4	The target bit rate in constant bit rate mode. * Only valid when "ratecontrolmode"= cbr
profile	0, 1, 2	1	4/4	Indicate H264 profiles 0: baseline 1: main profile 2: high profile

Group: videoin_c<n>_s<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
forcei	1	N/A This is write-only.	7/6	Force stream to generate an I-frame in any time.

The following APIs are only available when "capability_videoin_flexiblebitrate" = 1.
Group: videoin_c<n>_s<m>_h264,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available if the value of "capability_videoin_flexiblebitrate" is 1.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxvbrbitrate	1000~"capability_videoin_c<n>_h264_maxbitrate"	"capability_videoin_c<n>_h264_maxbitrate "	4/4	The maximum allowed bit rate in fixed quality mode. When the bit rate exceeds this value, frames will be dropped to restrict the bit rate. * Only valid when "ratecontrolmode"= vbr

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
bitraterestriction	average, upperbound	upperbound	4/4	average: The average of encoded bit rate meets the target bit rate ("bitrate"), but the peak bit rate may beyond it. upperbound: The encoded bit rate will not exceed the target bit rate ("bitrate"). * Only valid when "ratecontrolmode"= cbr
prioritypolicy	framerate, imagequality	framerate	4/4	The policy to apply when the target bit rate is not sufficient to satisfy current encoded conditions. “framerate” indicates frame rate first. “imagequality” indicates image quality first. * Only valid when "ratecontrolmode"= cbr

Maximum frame rate:
"maxframe" is used to limit the frame rate of the encoded video to a level appropriate for your needs. Setting "maxframe" to a value higher than the product supports is the equivalent of applying no frame rate limit on the stream.

For older products, we have restricted the range of "maxframe" to values from 1 to the maximum fps that the product supports. However, setting "maxframe" to a higher value causes no harm. To enable easy integration, we have expanded the range to include values from 1 to 999 when the value of [httpversion] is greater than or equal to 0301.

Note that the encoded frame rate may not reach the specified “maxframe" rate if many streams are active simultaneously, as the streams share hardware resources equally. Some services such as privacy masks, motion, and tampering detection may also affect the frame rate of the encoded video.

The maximum frame rates of an H.264 stream in different resolutions are recorded in "capability_videoin_c<n>_h264_maxbitrate".

Intra-frame interval:
Unlike other vendors, VIVOTEK does not use the GOP structure to determine the length between two I-frames. Instead, we use the actual time interval, as we believe this approach is more intuitive and more suitable for real-world usage.

For more information, see http://en.wikipedia.org/wiki/Group_of_pictures

Fixed-quality mode:
In fixed-quality mode, frames are encoded at a fixed quality level no matter what the bit rate is. Five quality levels have been defined—Median, Standard, Good, Detailed, Excellent—ensuring consistent quality levels across products.

"qpercent" can be used to set a custom quality level. While “qvalue" provides the same capability, the range of "qvalue" varies from product to product, creating integration difficulties. As a result, “qvalue" is no longer supported, and VIVOTEK strongly recommends using "qpercent".

If the value of "capability_videoin_flexiblebitrate" is 1, then the bit rate in fixed-quality mode is limited by the value of "maxvbrbitrate". The maximum value of "maxvbrbitrate" is the maximum bit rate the product can support. The drop-frame mechanism is applied to throttle the bit rate if necessary. This feature can also be used to control the bit rate when using the fixed-quality mode.

Constant bit rate mode:
In constant bit rate mode, frames are encoded with variable quality between inter frames to achieve the target bit rate.

If the value of "capability_videoin_flexiblebitrate" is 0, the bit rate of the encoded video may sometimes exceed the target bit rate, but the average bit rate will conform.

If the value of "capability_videoin_flexiblebitrate" is 1, there are two approaches available—average and upperbound—and two priority policies available—framerate and imagequality.
When average is used, the encoded bit rate may sometimes exceed the target bit rate, but the average bit rate will conform.
When upperbound is used, the encoded bit rate is not allowed to exceed the target bit rate at any time.

Priority policy affects how the bit rate is controlled.
When framerate is used, image quality is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is the frame rate adjusted.
When imagequality is used, the frame rate is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is image quality adjusted.

Note that if the content of the video is extremely simple visually, the bit rate may be correspondingly low and not reach the target bit rate.

H.264 Profile:
For more information, seehttps://en.wikipedia.org/wiki/H.264/MPEG-4_AVC#Profiles

5.5.2. Default H.264 bit rates

TODO

5.5.3. Affected Parameters

None

5.5.4. Usage: Configuring H.264

Get H.264 maximum bit rates:

```
http://<IP>/cgi-
bin/viewer/getparam.cgi?capability_videoin_c<n>_h264_maxbitrate
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_h264_maxbitrate

Returns:

```
capability_videoin_c0_h264_ maxbitrate='40000000'
```

Set H.264 maximum encoded frame rates:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_maxframe=<0~999>
```

Ex: http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_maxframe=10

Returns:

```
videoin_c0_s0_h264_maxframe='10'
```

Set H.264 I-frame interval:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_intraperiod=<250, 500, 1000,
2000, 3000, 4000>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_intraperiod=4000

Returns:

```
videoin_c0_s0_h264_ intraperiod='4000'
```

Set H.264 bit rate control mode to fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_ratecontrolmode=vbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_ratecontrolmode=vbr

Returns:

```
videoin_c0_s0_h264_ ratecontrolmode='vbr'
```

Set H.264 bit rate control mode to constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_ratecontrolmode=cbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_ratecontrolmode=cbr

Returns:

```
videoin_c0_s0_h264_ ratecontrolmode='cbr'
```

Select uniform quality level for fixed-quality mode:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_quant=<1~5>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_quant=5

Returns:

```
videoin_c0_s0_h264_ quant='5'
```

Select custom quality level for fixed-quality mode:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_quant=100&
videoin_c<n>_s<m>_h264_qpercent=<1~100>
```

Ex:
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c0_s0_h264_quant=100&videoin_c0_s0_h264_qercent=50

Returns:

```
videoin_c0_s0_h264_quant ='100'
videoin_c0_s0_h264_qpercent='50'
```

Set maximum bit rate for fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_maxvbrbitrate=<1000~"capa
bility_videoin_c<n>_h264_maxbitrate">
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_maxvbrbitrate=20000000

Returns:

```
videoin_c0_s0_h264_maxvbrbitrate='20000000'
```

Set target bit rate for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_bitrate=<1000~"capability_vi
deoin_c<n>_h264_maxbitrate">
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_bitrate=8000000

Returns:

```
videoin_c0_s0_h264_bitrate='8000000'
```

Set bit rate strategy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_bitraterestriction=<average,
upperbound>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_bitraterestriction=upperbound

Returns:

```
videoin_c0_s0_h264_bitraterestriction='upperbound'
```

Set priority policy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_prioritypolicy=<framerate,
imagequality>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_prioritypolicy=imagequality

Returns:

```
videoin_c0_s0_h264_prioritypolicy='imagequality'
```

Set H.264 profile:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_h264_profile=<0, 1,
2>
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_h264_profile=1

Returns:

```
videoin_c0_s0_h264_profile ='1'
```

Force generation of a single I-frame:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_forcei=1
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_forcei=1

Returns:

```
videoin_c0_s0_forcei ='1'
```

5.6. Codec: MPEG4

When a stream's codec type is set to MPEG4 ("videoin_c<n>_s<m>"= mpeg4), the APIs in this section are valid.

5.6.1. WebAPI

Group: videoin_c<n>_s<m>_mpeg4,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxframe	1~999	The maximum value in "capability_videoin_c<n>_mpeg4_maxframerate"	4/4	The maximum encoded frame rate.
intraperiod	250, 500, 1000, 2000, 3000, 4000	1000	4/4	The time interval between two I-frames (Intra coded picture). The unit is millisecond (ms).
ratecontrolmode	cbr, vbr	cbr	4/4	cbr: Constant bit rate mode. vbr: Fixed quality mode, all frames are encoded in the same quality.
quant	1~5, 99, 100	3	4/4	* Only valid when "ratecontrolmode"= vbr. Set the pre-defined quality level: 1: Median 2: Standard 3: Good 4: Detailed 5: Excellent 100: Use the quality level in "qpercent" 99: Use the quality level in "qvalue"
qpercent	1~100	50	4/4	Select customized quality in a normalized full range. 1: Worst quality 100: Best quality * Only valid when "ratecontrolmode"= vbr and "quant"= 100.
qvalue	<By product> * Not recommended to use this.	<By product>	4/4	The Q value which is used by encoded library directly. The range is different from product to product. We keep this for compatibility and internal use. * It's strongly non-recommended to use this.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
				* Only valid when "ratecontrolmode"= vbr and "quant"= 99.
bitrate	1000~"capability_videoin_c<n>_mpeg4_maxbitrate"	Please refer 5.6.2 Default MPEG4 bit rates	4/4	The target bit rate in constant bit rate mode. * Only valid when "ratecontrolmode"= cbr

Group: videoin_c<n>_s<m>,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
forcei	1	N/A This is write-only.	7/6	Force stream to generate an I-frame in any time.

The following APIs are only available when the value of "capability_videoin_flexiblebitrate" is 1.
Group: videoin_c<n>_s<m>_mpeg4,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available if the value of "capability_videoin_flexiblebitrate" is 1.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxvbrbitrate	1000~"capability_videoin_c<n>_mpeg4_maxbitrate"	"capability_videoin_c<n>_mpeg4_maxbitrate "	4/4	The maximum allowed bit rate in fixed quality mode. When the bit rate exceeds this value, frames will be dropped to restrict the bit rate. * Only valid when "ratecontrolmode"= vbr
bitraterestriction	average, upperbound	upperbound	4/4	average: The average of encoded bit rate meets the target bit rate ("bitrate"), but the peak bit rate may beyond it. upperbound: The encoded bit rate will not exceed the target bit rate ("bitrate").

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
				* Only valid when "ratecontrolmode"= cbr
prioritypolicy	framerate, imagequality	framerate	4/4	The policy to apply when the target bit rate is not sufficient to satisfy current encoded conditions. “framerate” indicates frame rate first. “imagequality” indicates image quality first. * Only valid when "ratecontrolmode"= cbr

Maximum frame rate:
"maxframe" is used to limit the encoded fps to a level appropriate for your needs. Setting "maxframe" to a value higher than the product supports is the equivalent of applying no frame rate limit to the stream.

For older products, we have restricted the range of "maxframe" to values from 1 to the maximum frame rate that the product supports. However, setting "maxframe" to a higher value causes no harm. To enable easy integration, we have expanded the range to include values from 1 to 999 when the value of [httpversion] is greater than or equal to 0301.

Note that the encoded frame rate may not reach the specified “maxframe" rate if many streams are active simultaneously, as the streams share hardware resources equally. Some services such as privacy masks, motion in the frame, or tampering may also affect the encoded frame rate.

The maximum frame rates of a MPEG4 stream at different resolutions are recorded in "capability_videoin_c<n>_mpeg4_maxframerate".

Intra-frame interval
Unlike other vendors, VIVOTEK does not use the GOP structure to determine the length between two I-frames. Instead, we use the actual time interval, as we believe this approach is more intuitive and more suitable for real-world usage.

For more information, seehttp://en.wikipedia.org/wiki/Group_of_pictures

Fixed-quality mode:
In fixed-quality mode, frames are encoded at a fixed quality level no matter what the bit rate is. Five quality levels have been defined—Median, Standard, Good, Detailed, Excellent—ensuring consistent quality levels across products.

"qpercent" can be used to set a custom quality level. While “qvalue" provides the same capability, the range of "qvalue" varies from product to product, creating integration difficulties. As a result, “qvalue" is no longer supported, and VIVOTEK strongly recommends using "qpercent".

If the value of "capability_videoin_flexiblebitrate" is 1, then the bit rate in fixed-quality mode is limited by "maxvbrbitrate". The maximum value of "maxvbrbitrate" is the maximum bit rate the product can support.The drop-frame mechanism is applied to throttle the bit rate if necessary. This feature can also be used to control the bit rate when using the fixed-quality mode.

Constant bit rate mode:
In constant bit rate mode, frames are encoded with variable quality between inter frames to achieve the target bit rate.

If the value of "capability_videoin_flexiblebitrate" is 0, the encoded bit rate may sometimes exceed the target bit rate, but the average bit rate will conform.

If the value of "capability_videoin_flexiblebitrate" is 1, there are two approaches available—average and upperbound—and two priority policies available—framerate and imagequality.
When average is used, the encoded bit rate may sometimes exceed the target bit rate, but the average bit rate will conform.
When upperbound is used, the encoded bit rate is not allowed to exceed the target bit rate at any time.

Priority policy affects how the bit rate is controlled.
When framerate is used, image quality is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is the frame rate adjusted.
When imagequality is used, the frame rate is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is image quality adjusted.

Note that if the content of the video is extremely simple visually, the bit rate may be correspondingly low and not reach the target bit rate.

5.6.2. Default MPEG4 bit rates

TODO

5.6.3. Affected Parameters

None

5.6.4. Configuring MPEG4

Get MPEG4 maximum bit rates:

```
http://<IP>/cgi-
bin/viewer/getparam.cgi?capability_videoin_c<n>_mpeg4_maxbitrate
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mpeg4_maxbitrate

Returns:

```
capability_videoin_c0_mpeg4_ maxbitrate='40000000'
```

Set MPEG4 maximum encoded frame rates:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_maxframe=<0~999>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_maxframe=10

Returns:

```
videoin_c0_s0_mpeg4_maxframe='10'
```

Set MPEG4 I-frame interval:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_intraperiod=<250, 500,
1000, 2000, 3000, 4000>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_intraperiod=4000

Returns:

```
videoin_c0_s0_mpeg4_ intraperiod='4000'
```

Set MPEG4 bit rate control mode to fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_ratecontrolmode=vbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_ratecontrolmode=vbr

Returns:

```
videoin_c0_s0_mpeg4_ ratecontrolmode='vbr'
```

Set MPEG4 bit rate control mode to constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_ratecontrolmode=cbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_ratecontrolmode=cbr

Returns:

```
videoin_c0_s0_mpeg4_ ratecontrolmode='cbr'
```

Select uniform quality level for fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_quant=<1~5>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_quant=5

Returns:

```
videoin_c0_s0_mpeg4_ quant='5'
```

Select custom quality level for fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_quant=100&
videoin_c<n>_s<m>_mpeg4_qpercent=<1~100>
```

Ex: http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_quant=100&videoin_c0_s0_mpeg4_qercent=50

Returns:

```
videoin_c0_s0_mpeg4_quant ='100'
videoin_c0_s0_mpeg4_qpercent='50'
```

Set maximum bit rate for fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_maxvbrbitrate=<1000~"cap
ability_videoin_c<n>_mpeg4_maxbitrate">
```

Ex: http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_maxvbrbitrate=20000000

Returns:

```
videoin_c0_s0_mpeg4_maxvbrbitrate='20000000'
```

Set target bit rate for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_bitrate=<1000~"capability_
videoin_c<n>_mpeg4_maxbitrate">
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_bitrate=8000000

Returns:

```
videoin_c0_s0_mpeg4_bitrate='8000000'
```

Set bit rate strategy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_bitraterestriction=<
average, upperbound>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_bitraterestriction=upperbound

Returns:

```
videoin_c0_s0_mpeg4_bitraterestriction='upperbound'
```

Set priority policy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mpeg4_prioritypolicy=<framerate,
imagequality>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mpeg4_prioritypolicy=imagequality

Returns:

```
videoin_c0_s0_mpeg4_prioritypolicy='imagequality'
```

Force generation of a single I-frame:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_forcei=1
```

Ex:

http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_forcei=1

Returns:

```
videoin_c0_s0_forcei ='1'
```

5.7. Codec: MJPEG

When a stream's codec type is set to MJPEG ("videoin_c<n>_s<m>"= mjpeg), the APIs in this section are valid.

5.7.1. WebAPI

Group: videoin_c<n>_s<m>_mjpeg,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxframe	1~999	The maximum value in "capability_videoin_c<n>_mjpeg_maxframerate"	4/4	The maximum encoded frame rate.
quant	1~5, 99, 100	3	4/4	* Only valid when "ratecontrolmode"= vbr. Set the pre-defined quality level: 1: Median 2: Standard 3: Good 4: Detailed 5: Excellent 100: Use the quality level in "qpercent" 99: Use the quality level in "qvalue"
qpercent	1~100	50	4/4	Select customized quality in a normalized full range. 1: Worst quality 100: Best quality * Only valid when "ratecontrolmode"= vbr and "quant"= 100.
qvalue	<By product> * Not recommended to use this.	<By product>	4/4	The Q value which is used by encoded library directly. The range is different from product to product. We keep this for compatibility and internal use. * It's strongly non-recommended to use this. * Only valid when "ratecontrolmode"= vbr and "quant"= 99.

The following APIs are only available when the value of "capability_videoin_flexiblebitrate" is 1.
Group: videoin_c<n>_s<m>_mjpeg,
n denotes the channel index, with a range from 0 to one less than the value of "capability_nvideoin";
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available if the value of "capability_videoin_flexiblebitrate" is 1.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
ratecontrolmode	cbr, vbr	cbr	4/4	cbr: Constant bit rate mode. vbr: Fixed quality mode, all frames are encoded in the same quality.
maxvbrbitrate	1000~"capability_videoin_c<n>_mjpeg_maxbitrate"	"capability_videoin_c<n>_mjpeg_maxbitrate "	4/4	The maximum allowed bit rate in fixed quality mode. When the bit rate exceeds this value, frames will be dropped to restrict the bit rate. * Only valid when "ratecontrolmode"= vbr
bitrate	1000~"capability_videoin_c<n>_mjpeg_maxbitrate"	Please refer 5.7.2 Default MJPEG bit rates	4/4	The target bit rate in constant bit rate mode. * Only valid when "ratecontrolmode"= cbr
bitraterestriction	average, upperbound	upperbound	4/4	average: The average of encoded bit rate meets the target bit rate ("bitrate"), but the peak bit rate may beyond it. upperbound: The encoded bit rate will not exceed the target bit rate ("bitrate"). * Only valid when "ratecontrolmode"= cbr
prioritypolicy	framerate, imagequality	framerate	4/4	The policy to apply when the target bit rate is not sufficient to satisfy current encoded conditions. "framerate" indicates frame rate first. "imagequality" indicates image quality first. * Only valid when "ratecontrolmode"= cbr

Maximum frame rate:

"maxframe" is used to limit the encoded frame rate to a level appropriate for your needs. Setting "maxframe" to a value higher than the product supports is the equivalent of applying no frame rate limit to the stream.

For older products, we have restricted the range of "maxframe" to values from 1 to the maximum frame rate that the product supports. However, setting "maxframe" to a higher value causes no harm. To enable easy integration, we have expanded the range to include values from 1 to 999 when the value of [httpversion] is greater than or equal to 0301.

Note that the encoded frame rate may not reach the specified “maxframe" rate if many streams are active simultaneously, as the streams share hardware resources equally. Some services such as privacy masks, motion in the frame, or tampering may also affect the encoded frame rate.

The maximum frame rates of a MJPEG stream at different resolutions are recorded in "capability_videoin_c<n>_mjpeg_maxframerate".

Fixed-quality mode:
In fixed-quality mode, frames are encoded at a fixed quality level no matter what the bit rate is. Five quality levels have been defined—Median, Standard, Good, Detailed, Excellent—ensuring consistent quality levels across products.

"qpercent" can be used to set a custom quality level. While “qvalue" provides the same capability, the range of "qvalue" varies from product to product, creating integration difficulties. As a result “qvalue" is no longer supported, and VIVOTEK strongly recommends using "qpercent".

If the value of "capability_videoin_flexiblebitrate" is 1, then the bit rate in fixed-quality mode is limited by "maxvbrbitrate". The maximum value of "maxvbrbitrate" is the maximum bit rate the product can support.The drop-frame mechanism is applied to throttle the bit rate if necessary. This feature can also be used to control the bit rate when using the fixed-quality mode.

Constant bit rate mode:
If the value of "capability_videoin_flexiblebitrate" is 0, MJPEG has no control over the bit rate.

If the value of "capability_videoin_flexiblebitrate" is 1, MJPEG has control over the bit rate. Frames are encoded with variable quality between frames to reach the target bit rate.

There are two approaches available—average and upperbound—and two priority policies available—framerate and imagequality.

When average is used, the encoded bit rate may sometimes exceed the target bit rate, but the average bit rate will conform.

When upperbound is used, the encoded bit rate is not allowed to exceed the target bit rate at any time.

Priority policy affects how the bit rate is controlled.

When framerate is used, image quality is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is the frame rate adjusted.

When imagequality is used, the frame rate is first adjusted in an attempt to achieve the target bit rate, but if unsuccessful, only then is image quality adjusted.

Note that if the content of the video is extremely simple visually, the bit rate may be correspondingly low and not reach the target bit rate.

5.7.2. Default MJPEG bit rates

TODO

5.7.3. Affected Parameters

None

5.7.4. Usage: Setting MJPEG

Get MJPEG maximum bit rates:

```
http://<IP>/cgi-bin/viewer/getparam.cgi?capability_videoin_c<n>_mjpeg_maxbitrate
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_videoin_c0_mjpeg_maxbitrate

Returns:

```
capability_videoin_c0_mjpeg_ maxbitrate='40000000'
```

Set MJPEG maximum encoded frame rate:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_maxframe=<0~999>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_maxframe=10
Return:
Returns:

```
videoin_c0_s0_mjpeg_maxframe='10'
```

Set MJPEG bit rate control mode to fixed-quality mode:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_ratecontrolmode=vbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_ratecontrolmode=vbr
Return:
Returns:

```
videoin_c0_s0_mjpeg_ ratecontrolmode='vbr'
```

Set MJPEG bit rate control mode to constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_ratecontrolmode=cbr
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_ratecontrolmode=cbr

Returns:

```
videoin_c0_s0_mjpeg_ ratecontrolmode='cbr'
```

Select uniform quality level for fixed quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_quant=<1~5>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_quant=5

Returns:

```
videoin_c0_s0_mjpeg_ quant='5'
```

Select custom quality level for fixed-quality mode:

```
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_quant=100&
videoin_c<n>_s<m>_mjpeg_qpercent=<1~100>
```

Ex:
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_quant=100&videoin_c0_s0_mjpeg_qercent=50

Returns:

```
videoin_c0_s0_mjpeg_quant ='100'
videoin_c0_s0_mjpeg_qpercent='50'
```

Set maximum bit rate for fixed-quality mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_maxvbrbitrate=<1000~"cap
ability_videoin_c<n>_mpegj_maxbitrate">
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_maxvbrbitrate=20000000

Returns:

```
videoin_c0_s0_mjpeg_maxvbrbitrate='20000000'
```

Set target bit rate for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_bitrate=<1000~"capability_v
ideoin_c<n>_mjpeg_maxbitrate">
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_bitrate=8000000

Returns:

```
videoin_c0_s0_mjpeg_bitrate='8000000'
```

Set bit rate strategy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_bitraterestriction=< average,
upperbound>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_bitraterestriction=upperbound

Returns:

```
videoin_c0_s0_mpeg4_bitraterestriction='upperbound'
```

Set priority policy for constant bit rate mode:

```
http://<IP>/cgi-
bin/admin/setparam.cgi?videoin_c<n>_s<m>_mjpeg_prioritypolicy=<framerate,
imagequality>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?videoin_c0_s0_mjpeg_prioritypolicy=imagequality

Returns:

```
videoin_c0_s0_mjpeg_prioritypolicy='imagequality'
```

6. Streaming Service Domain API

Streaming services are responsible for sending out encoded video data from a server to a client. There are four such services: Server Push, RTSP, single snapshot, and Anystream.

Server Push supports streaming of MJPEG-encoded video only. If the device supports Server Push, then the value of "capability_protocol_spush_mjpeg" is set to 1.

RTSP supports streaming of encoded video using any codec. If the device supports RTSP, then the value of "capability_protocol_rtsp" is set to 1.

Single snapshot provides an HTTP interface for capturing a single snapshot from the device. All VIVOTEK IP cameras and video servers support this service.

Anystream is a useful feature that enables access to a stream with customized settings. It allows video parameters to be set dynamically by passing them via a URL. For more details, please refer to Section 6.6, “AnyStream”. If the device supports Anystream, then value of "capability_nanystream" is not 0, and the value indicates the number of Anystream streams provided.

6.1. Opening streaming

6.1.1. Usage: Opening a RTSP Stream

For 1-channel products ("capability_nvideoin"= 1):

1. **Get the RTSP stream's access name:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_s<m>_accessname
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_s0_accessname

Returns:

```
network_rtsp_s0_accessname='live.sdp'
```

2. **Get the RTSP network port:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_port
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_port

Returns:

```
network_rtsp_port='554'
```

3. **Open a RTSP stream with a RTSP-compatible player such as VLC:**

```
rtsp://<IP>:<network_rtsp_port>/<network_rtsp_s<m>_accessname>
```

Ex:

rtsp://*169.254.0.99:554*/live.sdp

Note: The port number can be omitted if its value is the default, 554.

rtsp://*169.254.0.99*/live.sdp

4. **Get an RTSP SDP file via HTTP:**

```
http://<IP>/<network_rtsp_s<m>_accessname>
```

Ex:

http://*169.254.0.99/live.sdp*

Returns:

```
v=0
o=RTSP 1370975556 311 IN IP4 0.0.0.0
s=RTSP server
c=IN IP4 0.0.0.0
t=0 0
a=charset:Shift_JIS
a=range:npt=0-
a=control:*
a=etag:1234567890
m=video 0 RTP/AVP 98
b=AS:0
a=rtpmap:98 H264/90000
a=control:trackID=1
a=x-onvif-track:trackID=1
a=fmtp:98 packetization-mode=1; profile-level-id=428033; sprop-
parameter-sets=J0KAM4uVAEABgyA=,KN4DmIA=
```

For multi-channel products ("capability_nvideoin">= 2):

1. **Get a RTSP stream's access name:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_c<n>_s<m>_accessname
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_c1_s1_accessname

Returns:

```
network_rtsp_c1_s1_accessname='live2s2.sdp'
```

2. **Get the RTSP network port:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_port
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_rtsp_port

Returns:

```
network_rtsp_port='554'
```

3. **Open a RTSP stream with a RTSP-compatible player such as VLC:**

```
rtsp://<IP>:<network_rtsp_port>/<network_rtsp_c<n>_s<m>_accessname>
```

Ex:

rtsp://*169.254.0.99:554*/live2s2.sdp

Note: The port number may be omitted if its value is the default, 554.

rtsp://*169.254.0.99*/live2s2.sdp

4. Get an RTSP SDP file via HTTP:

```
http://<IP>/<network_rtsp_c<n>_s<m>_accessname>
```

Ex:

http://*169.254.0.99/live.sdp*

Returns:

```
v=0
o=RTSP 1370975556 311 IN IP4 0.0.0.0
s=RTSP server
c=IN IP4 0.0.0.0
t=0 0
a=charset:Shift_JIS
a=range:npt=0-
a=control:*
a=etag:1234567890
m=video 0 RTP/AVP 98
b=AS:0
a=rtpmap:98 H264/90000
a=control:trackID=1
a=x-onvif-track:trackID=1
a=fmtp:98 packetization-mode=1; profile-level-id=428033; sprop-
parameter-sets=J0KAM4uVAEABgyA=,KN4DmIA=
```

6.1.2. Usage: Opening a Server Push Stream

For 1-channel products ("capability_nvideoin"= 1):

1. Get a server push stream's access name:

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_s<m>_accessname
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_s0_accessname

Returns:

```
network_http_s0_accessname='video.mjpg'
```

2. Get the HTTP network port:

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_port
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_port

Returns:

```
network_http_port='80'
```

3. **Open a server push stream with a HTTP-compatible player such as VLC or Google Chrome:**

```
http://<IP>:<network_http_port>/<network_http_s<m>_accessname>
```

Ex:

http://*169.254.0.99:80*/video.mjpg

Note: The port number can be omitted if its value is the default, 80.

http://*169.254.0.99*/video.mjpg

For multi-channel products ("capability_nvideoin">= 2):

1. **Get server push stream's access name:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_c<n>_s<m>_accessname
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_c2_s1_accessname

Returns:

```
network_http_c2_s1_accessname='video3s2.mjpg'
```

2. **Get the HTTP network port:**

```
http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_port
```

Ex:

http://<IP>/cgi-bin/viewer/getparam.cgi?network_http_port

Returns:

```
network_http_port='80'
```

3. **Open a server push stream with a HTTP-compatible player such as VLC or Google Chrome:**

```
http://<IP>:<network_http_port>/<network_http_c<n>_s<m>_accessname>
```

Ex:

http://*169.254.0.99:80*/video3s2.mjpg

Note: The port number may be omitted if its value is the default, 80.

http://*169.254.0.99*/video3s2.mjpg

6.2. Maximum Number of Active Streams

The number of active streams supported by any given product is limited. Streaming connections may use RTSP or server push services. “Active” means the connection has been established.

Attempts to open a stream may fail if the number of active streams has already reached the maximum supported.

6.2.1. WebAPI

The maximum number of active streams supported by a given product can be found via the following API:
Group: capability_protocol

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
maxconnection	<Positive Integer>	10 (on most products)	1/7	The maximum number of active streaming connections.

Runtime connection information can be obtained via the following APIs:
Group: status_onlinenum

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
rtsp	0~<Positive Integer>	0	1/7	Current number of RTSP connections.
httppush	0~<Positive Integer>	0	1/7	Current number of server push connections.

Note: The sum of the values for "rtsp" and "httppush" will always be less than or equal to the value of "capability_protocol_maxconnection".

6.2.2. Affected Parameters

None. These APIs are read-only.

6.2.3. Usage: Obtaining Connection Information

Get the maximum number of active streams:

```
http://<IP>/cgi-bin/admin/getparam.cgi?capability_protocol_maxconnection
```

Ex:
http://<IP>/cgi-bin/admin/getparam.cgi?capability_protocol_maxconnection

Returns:

```
capability_protocol_maxconnection='10'
```

Get run-time connection information:

```
http://<IP>/cgi-bin/admin/getparam.cgi?status_onlinenum
```

Ex:

http://<IP>/cgi-bin/admin/getparam.cgi?status_onlinenum

Returns:

```
status_onlinenum_rtsp='5'
status_onlinenum_httppush='3'
```

The return values shown indicate that 5 RTSP connections and 3 server push connections are active.

6.3. RTSP

RTSP can send out streams encoded in the MPEG4, H.264, or MJPEG formats.

The default configuration of our RTSP server uses RTP-over-UDP to stream data. It also supports RTP-over-TCP when the value of “capability_protocol_rtp_tcp” is set to 1, and RTP-over-HTTP when the value of “capability_protocol_rtp_http" is set to 1.

6.3.1. WebAPI

The APIs for RTSP and RTP are listed in the tables below:
Group: network_rtsp

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
port	554, 1025~65535	554	1/6	RTSP port. * Only available when "capability_protocol_rtsp"=1
authmode	disable, basic, digest	basic	1/6	RTSP authentication mode. disable: allow anonymous streaming viewing. * Only available when "capability_protocol_rtsp"=1
anonymousvie wing	0, 1	0	1/6	Enable anonymous streaming viewing. When enable this, anyone can access RTSP streaming without any password check. * Only available when "capability_protocol_rtsp"=1

Group: network_rtp

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
videoport	1025~65534	5556	6/6	Video channel port for RTP.
videoport_rtcp	"videoport"+1	5557	7/7	Video channel port for RTCP. The value is auto-set to "videoport"+1. * This is a hidden parameter.
audioport	1025~65534	5558	6/6	Audio channel port for RTP.
audioport_rtcp	"audioport"+1	5559	7/7	Audio channel port for RTCP. The value is auto-set to "audioport"+1. * This is a hidden parameter.

For 1-channel products ("capability_nvideoin"= 1):
Group: network_rtsp

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
s<m>_accessname	String[32]	live.sdp for s0, live2.sdp for s1, live3.sdp for s2, and so on.	1/6	RTSP access name for a stream. * The maximum value of m is ("capability_nmediastream" + "capability_nanystream" -1). * Only available when "capability_protocol_rtsp"=1

* Note: On 1-channel products on which the value of [httpversion] is greater than or equal to 0301, the group network_rtsp_c0 is also available. We strongly recommend use of this group.

Group: network_rtsp_s<m>_multicast,
m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".
* Only available when "capability_protocol_rtsp_multicast"=1

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
alwaysmulticast	0, 1	0	4/4	Always enable multicast.
ipaddress	<IP>	239.128.1.99 for s0, 239.128.1.100 for s1, and so on.	4/4	Multicast IP address.
videoport	1025~65534	5560 for s0, 5564 for s1, 5568 for s2, and so on.	4/4	Multicast video port.
videoport_rtcp	"videoport"+1	5561 for s0, 5565 for s1, 5569 for s2, and so on.	7/7	Multicast video port for RTCP. The value is auto-set to "videoport"+1. * This is a hidden parameter.
audioport	1025~65534	5562 for s0, 5566 for s1, 5570 for s2, and so on.	4/4	Multicast audio port. * Only available when "capability_naudioin">=1
audioport_rtcp	"audioport"+1	5563 for s0, 5567 for s1, 5571 for s2, and so on.	7/7	Multicast audio port for RTCP. The value is auto-set to "audioport"+1. * This is a hidden parameter.
ttl	1~255	15	4/4	Mutlicast time to live value.

* Note: On 1-channel products on which the value of [httpversion] is greater than or equal to 0301, the group network_rtsp_c0_ s<m>_multicast is also available. We strongly recommend use of this group.

For multi-channel products ("capability_nvideoin">= 2):

Group: network_rtsp

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
c<n>_s<m>_accessname	String[32]	live.sdp for c0_s0, lives2.sdp for c0_s1, lives3.sdp for c0_s2, live2.sdp for c1_s0, live2s2.sdp for c1_s1, live2s3.sdp for c1_s2, live3.sdp for c2_s0, live3s2.sdp for c2_s1, live3s3.sdp for c2_s2, and so on.	1/6	RTSP access name for a stream. * The maximum value of m is ("capability_nmediastream" + "capability_nanystream" -1). * Only available when "capability_protocol_rtsp"=1

Group: network_rtsp_c<n>_s<m>_multicast,

m denotes the stream index, with a range from 0 to one less than the value of "capability_nmediastream".

* Only available when "capability_protocol_rtsp_multicast"=1

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
alwaysmulticast	0, 1	0	4/4	Always enable multicast.
ipaddress	<IP>	For 4-CH product with 2 stream per channel: 239.128.1.99 for c0_s0. 239.128.1.100 for c0_s1, 239.128.1.101 for c1_s0, 239.128.1.102 for c1_s1, and so on. For 2-CH product with 3 stream per channel: 239.128.1.99 for c0_s0. 239.128.1.100 for c0_s1, 239.128.1.101 for c0_s2, 239.128.1.102 for c1_s0, 239.128.1.103 for c1_s1, 239.128.1.104 for c1_s2, and so on.	4/4	Multicast IP address.

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
videoport	1025~65534	For 4-CH product with 2 stream per channel: 5560 for c0_s0, 5564 for c0_s1, 5568 for c1_s0, 5572 for c1_s1, and so on. For 2-CH product with 3 stream per channel: 5560 for c0_s0. 5564 for c0_s1, 5568 for c0_s2, 5572 for c1_s0, 5576 for c1_s1, 5580 for c1_s2, and so on.	4/4	Multicast video port.
videoport_rtcp	"videoport"+1	For 4-CH product with 2 stream per channel: 5561 for c0_s0, 5565 for c0_s1, 5569 for c1_s0, 5573 for c1_s1, and so on. For 2-CH product with 3 stream per channel: 5561 for c0_s0. 5565 for c0_s1, 5569 for c0_s2, 5573 for c1_s0, 5577 for c1_s1, 5581 for c1_s2, and so on.	7/7	Multicast video port for RTCP. The value is auto-set to "videoport"+1. * This is a hidden parameter.
audioport	1025~65534	For 4-CH product with 2 stream per channel: 5562 for c0_s0, 5566 for c0_s1, 5570 for c1_s0, 5574 for c1_s1, and so on. For 2-CH product with 3 stream per channel: 5562 for c0_s0. 5566 for c0_s1, 5570 for c0_s2, 5574 for c1_s0, 5578 for c1_s1, 5582 for c1_s2, and so on.	4/4	Multicast audio port. * Only available when "capability_naudioin">=1

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
audioport_rtcp	"audioport"+1	For 4-CH product with 2 stream per channel: 5563 for c0_s0, 5567 for c0_s1, 5571 for c1_s0, 5575 for c1_s1, and so on. For 2-CH product with 3 stream per channel: 5563 for c0_s0. 5567 for c0_s1, 5571 for c0_s2, 5575 for c1_s0, 5579 for c1_s1, 5583 for c1_s2, and so on.	7/7	Multicast audio port for RTCP. The value is auto-set to "audioport"+1. * This is a hidden parameter.
ttl	1~255	15	4/4	Multicast time to live (TTL) value.

6.3.2. Affected Parameters

Changing some streaming service APIs affects other features. These APIs are listed in the following table:

Name	Effect
network_rtsp_s<m>_multicast_alwaysmulticast	When set to 1, the stream starts encoding and streaming service starts to broadcast it.
network_rtsp_c<n>_s<m>_multicast_alwaysmulticast	When set to 1, the stream starts encoding and streaming service starts to broadcast it.
network_rtp_videoport	Each RTP port has a related RTCP port. The rule for RTCP port is fixed to RTP port +1. RTCP port is not listed in WebAPI, but it does uses the port. Please notice RTCP ports when you change any port of network service. Binding two services to one port causes either one or both services fail.
network_rtp_audioport	
network_rtsp_s<m>_multicast_videoport	
network_rtsp_s<m>_multicast_audioport	
network_rtsp_c<n>_s<m>_multicast_videoport	
network_rtsp_c<n>_s<m>_multicast_audioport	

6.3.3. Usage: Configuring the RTSP Service

*** Note: Attempts to switch to a port currently used by another service may fail and should be avoided.**

Set the RTSP port:

```
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_port=<554,1025~65535>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_port=554

Returns:

```
network_rtsp_port='554'
```

Set anonymous viewing of RTSP streams:
Set the RTSP authentication mode:

Please refer to the document, “VIVOTEKWebAPI_Security”.

Set the access name of an RTSP stream:

For 1-channel products ("capability_nvideoin"= 1):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_s<m>_accessname=<String[32]>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_s0_accessname="RTSP.sdp"

Returns:

```
network_rtsp_s0_accessname='RTSP.sdp'
```

For multi-channel products ("capability_nvideoin">= 2):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_c<n>_s<m>_accessname=<String[32]>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_c0_s0_accessname="RTSP.sdp"

Returns:

```
network_rtsp_c0_s0_accessname='RTSP.sdp'
```

Enable/Disable RTSP multicasting for a stream:

For 1-channel products ("capability_nvideoin"= 1):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_s<m>_multicast_alwaysmulticast=<0,1>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_s0_multicst_alwaysmulticast=1

Returns:

```
network_rtsp_s0_multicast_alwaysmulticast='1'
```

For multi-channel products ("capability_nvideoin">= 2):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_c0_s<m>_multicast_alwaysmulticast=<0,1
>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_c0_s0_multicst_alwaysmulticast=1

Returns:

```
network_rtsp_c0_s0_multicast_alwaysmulticast='1'
```

Set an RTSP multicast IP address, video port, audio port, and TTL of a stream:

For 1-channel products ("capability_nvideoin"= 1):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_s<m>_multicast_ipaddress=<IP
address>&network_rtsp_s<m>_multicast_videoport=<1025~65534>&network_r
tsp_s<m>_multicast_audioport=<1024~65534>&network_rtsp_s<m>_multicast_
ttl=<1~255>
```

Ex:
http://<IP>/cgi-
bin/admin/setparam.cgi?network_rtsp_s0_multicst_ipaddress="39.39.39.39"&network_rtsp_s0_multicst_
videoport=3939&network_rtsp_s0_multicst_audioport=39393&network_rtsp_s0_multicst_ttl=39

Returns:

```
network_rtsp_s0_multicst_ipaddress='39.39.39.39'
network_rtsp_s0_multicst_videoport='3939'
network_rtsp_s0_multicst_audioport='39393'
network_rtsp_s0_multicst_ttl='39'
```

For multi-channel products ("capability_nvideoin">= 2):

```
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_c<n>_s<m>_multicast_ipaddress=<IP address>&network_rtsp_c<n>_s<m>_multicast_videoport=<1025~65534>&network_rtsp_c<n>_s<m>_multicast_audioport=<1024~65534>&network_rtsp_c<n>_s<m>_multicast_ttl=<1~255>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_rtsp_c0_s0_multicst_ipaddress="39.39.39.39"&network_rtsp_c0_s0_multicst_videoport=3939&network_rtsp_c0_s0_multicst_audioport=39393&network_rtsp_c0_s0_multicst_ttl=39

Returns:

```
network_rtsp_c0_s0_multicst_ipaddress='39.39.39.39'
network_rtsp_c0_s0_multicst_videoport='3939'
network_rtsp_c0_s0_multicst_audioport='39393'
network_rtsp_c0_s0_multicst_ttl='39'
```

6.4. Server Push

Server push can only stream MJPEG video using the HTTP protocol.

6.4.1. WebAPI

Server push APIs are listed below:
Group: network_http

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
port	80, 1025~65535	80	6/6	HTTP port.
alternateport	1025~65535	8080	6/6	Alternate HTTP port.
authmode	basic, digest	basic	1/6	HTTP authentication mode.
anonymousviewing	0, 1	0	1/6	Enable anonymous streaming viewing. When enable this, anyone can access server push streaming without any password check.

Note: These APIs are shared with other HTTP services such as web servers.

For 1-channel products ("capability_nvideoin"= 1):
Group: network_http

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
s<m>_accessname	String[32]	video.mjpg for s0, video2.mjpg for s1, video3.mjpg for s2, and so on.	1/6	Server push access name for a stream. * The maximum value of m is ("capability_nmediastream" + "capability_nanystream" -1). * Only available when "capability_protocol_spush_mjpeg"=1

* Note: On 1-channel products on which the value of [httpversion] is greater than or equal to 0301, the group network_http_c0 is also available. We strongly recommend use of this group.

For multi-channel products ("capability_nvideoin">= 2):
Group: network_http

NAME	VALUE	DEFAULT	SECURITY (get/set)	DESCRIPTION
c<n>_s<m>_accessname	String[32]	video.mjpg for c0_s0, videos2.mjpg for c0_s1, videos3.mjpg for c0_s2, video2.mjpg for c1_s0, video2s2.mjpg for c1_s1, video2s3.mjpg for c1_s2, video3.mjpg for c2_s0, video3s2.mjpg for c2_s1, video3s3.mjpg for c2_s2, and so on.	1/6	Server push access name for a stream. * The maximum value of m is ("capability_nmediastream" + "capability_nanystream" -1). * Only available when "capability_protocol_spush_mjpeg"=1

6.4.2. Affected Parameters

None.

6.4.3. Usage: Configuring the Server Push Service

Set the Server Push (HTTP) port:

```
http://<IP>/cgi-bin/admin/setparam.cgi?network_http_port=<80,1025~65535>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_http_port=80

Returns:

```
network_http_port='80'
```

Set anonymous viewing of Server Push streams:

Set the Server Push (HTTP) authentication mode:
Please refer to the document, VIVOTEKWebAPI_Security.doc

Set the access name for a Server Push stream:

For 1-channel products ("capability_nvideoin"= 1):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_http_s<m>_accessname=<String[32]>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_http_s0_accessname="SPUSH.mjpg"

Returns:

```
network_http_s0_accessname='SPUSH.mjpg'
```

For multi-channel products ("capability_nvideoin">= 2):

```
http://<IP>/cgi-
bin/admin/setparam.cgi?network_http_c<n>_s<m>_accessname=<String[32]>
```

Ex:
http://<IP>/cgi-bin/admin/setparam.cgi?network_http_c0_s0_accessname="SPUSH.mjpg"

Returns:

```
network_http_c0_s0_accessname='SPUSH.mjpg'
```

6.5. Single Snapshot

Single snapshot is a HTTP service to capture a single snapshot from a device in the JPEG format.

The snapshot is generated after a video module receives a request. The minimum interval between snapshots is 1 second and a new snapshot will not be generated if transmission of the previous one to a client has not been completed.

6.5.1. WebAPI

The HTTP interface:

```
http://<IP>/cgi-bin/viewer/video.jpg?
[channel=<value>][&resolution=<value>][&quality=<value>][&streamid=<value>]
```

Available parameters are listed in the table below:

PARAMETER	VALUE	DEFAULT	DESCRIPTION
channel	0~"capability_nvideoin"-1	0	The index of target channel.
streamid	0~"capability_nmediastream"-1	The index of a stream with full scene. The rule can be simplified as following: Products without ePTZ: 0 Products with ePTZ: "capability_nmediastream"-1	The index of target stream.
resolution	160x120~The maximum resolution of a channel.	The value of "videoin_c<n>_s<m>_resolution"	The requested resolution for snapshot. * W must be multiple of 16 pixels. * H must be multiple of 8 pixels.
quality	1~5	3	The quality of snapshot. 1: Median 2: Standard 3: Good 4: Detailed 5: Excellent

6.5.2. Affected Parameters

None.

6.5.3. Usage: Getting a Single Snapshot

The HTTP interface:

```
http://<IP>/cgi-bin/viewer/video.jpg?
[channel=<value>][&resolution=<value>][&quality=<value>][&streamid=<value>]
```

Ex:
http://<IP>/cgi-bin/viewer/video.jpg
or
http://<IP>/cgi-bin/viewer/video.jpg?streamid=1&resolution=640x480&quality=5

Returns:

```
HTTP/1.0 200 OK\r\n
Content-Type: image/jpeg\r\n
[Content-Length: <image size>\r\n]

<binary JPEG image data>
```

6.6. AnyStream

Anystream is a service that provides a way to access a video stream with custom settings.

The number of available Anystream streams per channel is recorded in "capability_nanystream".

6.6.1. WebAPI

Anystream setups the settings with the first connection request and keeps the settings until all connections are disconnected. 2nd and other connection requests' settings are ignored.

Two accessing URL paths for an Anystream: one for H.264&MPEG4, one for MJPEG. Only one path can be used in the same time.

The accessing method is as following:

H.264 and MPEG4:
For H.264, MPEG4 streaming, Anystream can be accessed by opening following path with RTPS compatible player like VLC:

```
rtsp://<IP>/ liveany.sdp?codectype=<h264,mpeg4>[&<custom settings>]
```

At least codectype must be added. Other custom settings are optional and the last values are used if they are not in the path.

Custom settings are listed in the table below:

PARAMETER	VALUE	DEFAULT	DESCRIPTION
codectype	h264,mpeg4 * Available options are list in "capability_videoin_codec".	N/A	Set the codec of the Anystream.
resolution	Available options are list in "capability_videoin_c0_resolution".	The maximum resolution of available options.	Video resolution in pixel.
h264_maxframe	1~999	15	The maximum frame rate. * Only valid if "codectype" is set to h264.
h264_ratecontrolmode	cbr, vbr	vbr	cbr: Constant bitrate vbr: Fix quality
h264_bitrate	4000~4000000	512000	Set bit rate in bps. * Only valid if "h264_ratecontrolmode" is set to cbr.
h264_quant	1~5,99	3	Quality of video when choosing vbr in "h264_ratecontrolmode".

PARAMETER	VALUE	DEFAULT	DESCRIPTION
			0,99 is the customized manual input setting. 1 = worst quality, 5 = best quality. * Only valid if "h264_ratecontrolmode" is set to vbr.
h264_qvalue	0~51	30	H.264 quantizer value. * Only valid if "h264_ratecontrolmode" is set to vbr, and "h264_quant" is set to 99.
h264_intraperiod	250, 500, 1000, 2000, 3000, 4000	1000	Intra frame period in milliseconds.
mpeg4_maxframe	1~999	15	The maximum frame rate. * Only valid if "codectype" is set to mpeg4.
mpeg4_ratecontrol mode	cbr, vbr	vbr	cbr: Constant bitrate vbr: Fix quality
mpeg4_bitrate	4000~4000000	512000	Set bit rate in bps. * Only valid if "mpeg4_ratecontrolmode" is set to cbr.
mpeg4_quant	1~5,99	3	Quality of video when choosing vbr in "mpeg4_ratecontrolmode". 0,99 is the customized manual input setting. 1 = worst quality, 5 = best quality. * Only valid if "mpeg4_ratecontrolmode" is set to vbr.
mpeg4_qvalue	2~31	7	MPEG4 quantizer value. * Only valid if "mpeg4_ratecontrolmode" is set to vbr, and "mpeg4_quant" is set to 99.
mpeg4_intraperiod	250, 500, 1000, 2000, 3000, 4000	1000	Intra frame period in milliseconds.

MJPEG:
For MJPEG streaming, Anystream can be accessed by opening following path with HTTP-compatible player such as VLC or Google Chrome:

```
http://<IP>/videoany.mjpg?codectype=mjpeg [&<custom settings>]
```

At least codectype must be added. Other custom settings are optional and the last values are used if they are not in the path.

Custom settings are listed in the table below:

PARAMETER	VALUE	DEFAULT	DESCRIPTION
codectype	mjpeg	N/A	Set the codec of the Anystream.
resolution	Available options are list in "capability_videoin_c0_resolution".	The maximum resolution of available options.	Video resolution in pixel.
mjpeg_maxframe	1~999	15	The maximum frame rate.
mjpeg_quant	1~5,99	3	Quality of JPEG video. 0,99 is the customized manual input setting. 1 = worst quality, 5 = best quality.
mjpeg_qvalue	10~200, or 2~97 * Product dependent	50	Manual video quality level * Only valid if "mjpeg_quant" is set to 99.

6.6.2. Affected Parameters

None.

6.6.3. Opening an Anystream Stream

Ex: Open a H.264 Anystream with 1280x720, CBR 1Mbps, 10 fps:

```
rtsp://<IP>/
liveany.sdp?codectype=h264&resolution=1280x720&h264_maxframe=10&h264_r
atecontrolmode=cbr&h264_bitrate=10&h264_intraperiod=4000
```

Ex: Open a MJPEG Anystream with 800x600, best quality, 5 fps:

```
http://<IP>/videoany.mjpg?codectype=mjpeg&resolution=800x600&mjpeg_maxfr
ame=5&mjpeg_quant=5
```