Video Streamer User Guide

UgCS 3.1.871

Contents

1 Video Streaming Service User Guide

Video streaming service is used to stream the video from a vehicle or other source directly into UgCS. For the video streaming service to work correctly the user needs to configure it in the UgCS Client and also in the configuration file. To do so, please carefully read the instructions below. After changing the configuration you must restart the Video Service via the Service Manager.

1.1 Configuration in client

You can configure the video service in client “Menu” > “Configuration” > “Video”. By default it points to a local instance. In the case of a multi node deployment network address and port of the Video service can be specified.

1.2 Configuration in the video-service properties file

You can configure the video-service settings in a video configuration file called vstreamer.conf. It can be found in the following paths:

- **On Microsoft Windows:**

```
C:\Program Files\UgCS\bin\vstreamer.conf
```

- **On Mac OS:**

```
~/Library/Application Support/UGCS/configuration/vstreamer.conf or
/Users/{username}/Library/Application Support/UGCS/configuration/vstreamer.conf
```

- **On GNU/Linux:**

```
/etc/opt/ugcs/vstreamer.conf
```

The configuration file vstreamer.conf has the following sections which will be described below: Main settings, Network streams settings, Video device settings, Log level, File path File, Maximum single file size

1.2.1 Main settings

Parameter name	Default value	Description
vstreamer.server_port	8081	Http port for the main video service server. This server starts streaming servers for each device. Streaming servers run on ports right next to the main server (e.g. if the main server runs on port 8081, streaming servers will run on ports 8082, 8083, etc.). Change this value if port conflicts appear. This port must be as specified in the settings of the video in the client

1.2.2 Network streams settings

You can also set a number of input network streams which will be re-streamed to the client. By default the streams for ArDrone and GoPro are configured. Feel free to add, change or remove streams.

Parameter name	Default value	Description
vstreamer.inputstream.#	-	Example: Ardrone;tcp://192.168.-1.1:5555;60;640;360 Format description: <Name>;<url>;<Timeout>;<-Width>;<Height> <Name>: name of stream as it will be seen in client. <Url>: URL of input network stream <Timeout>: Parameter is optional. Means time in seconds while http stream will keep trying to connect to stream after disconnecting and stream name will appear in list (same as timeout for video devices). <Width> and <Height>: image size. Parameters are optional, but may be needed for some streams.

Since version 2.13 we support Yuneec E50 payloads as well. For better performance Yuneec payload config is allocated to distinct parameters group: vstreamer.yuneecstream.# | - | Example: vstreamer.yuneecstream.0=Yuneec E90;rtsp://192.168.42.1:554/live

Format description:

<Name>;<url>

<Name>: name of stream as it will be seen in client.

<Url>: URL of input network stream

|

Since version 2.13 we support "copy stream" function. When you need to just resend stream from UAV to some video server without transcoding you can use this parameter group: vstreamer.copystream.# | - | Example: vstreamer.copystream.0=rtsp://184.72.239.149/vod/mp4:BigBuckBunny_175k.mov;rtsp://127.0.0.1:554/out;0

Format description:

<SourceURL>;<DestinationURL>;<StreamProfile>

<SourceURL>: URL of source stream

<DestinationURL>: URL of video server endpoint

<StreamProfile>: Profile of stream for better fit. 0 - standard profile, 1 - no-buffer profile for lower the latency, 2 - Yuneec profile.

|

Local ∗.avi file streaming from VLC is also available. You should use a custom MJPEG codec with parameters: encapsulation – M-JPEG, video codec- MJPEG, 800Kb/s, frame rate – same as source. Specify new video source in vstreamer.conf.

1.2.3 Video device settings

By default every video device found is available for streaming. You can change this behaviour by changing following settings:

Parameter name	Default value	Description
vstreamer.videodevices.autodetect	1	Set audodetecting of video devices. If set to “1” every detected device will be available for streaming except those which are in device.exclude list (see below). If set to “0” no devices will be detected and available for streaming except those which are in device.allow list (see below).
vstreamer.videodevices.exclude.#	-	List of device names which must not be available for streaming. Useful for excluding built-in video devices. By default this list is empty.
vstreamer.videodevices.allow.#	-	List of device names which must be available for streaming. If device auto detecting is turned off, you can manually set a list of devices available for streaming. The server will try to open these devices even if they were not auto detected. By default this list is empty.
vstreamer.videodevices.timeout	10	Timeout in seconds for video devices. Timeout occurs after the signal from the device is lost or no image data can be grabbed. After this period the device will be deleted from list of devices available for streaming, but it may appear again if device gives off a signal.

1.2.4 Log level

Optional.

- **Name:** log.level = [error|warning|info|debug]
- **Description:** Logging level.
- **Default:** info
- **Example:** log.level = debug

1.2.5 File path

Optional.

- **Name:** log.file_path = [path to a file]
- **Description:** Absolute or relative (to the current directory) path to a logging file. Logging is disabled if logging file is not defined. File should be writable. Backslash should be escaped with a backslash.
- **Example:** log.file = /var/opt/ugcs/log/vstreamer/vstreamer.log
- **Example:** log.file = C:.log

1.2.6 Maximum single file size

Optional.

- **Name:** log.single_max_size = [size]
- **Description:** Maximum size of a single log file. When maximum size is exceeded, existing file is renamed by adding a time stamp and logging is continued into the empty file. [size] should be defined as a number postfixed by a case insensitive multiplier:
  - Gb, G, Gbyte, Gbytes: for Giga-bytes
  - Mb, M, Mbyte, Mbytes: for Mega-bytes
  - Kb, K, Kbyte, Kbytes: for Kilo-bytes
  - no postfix: for bytes
- **Default:** 100 Mb
- **Example:** log.single_max_size = 500 Mb

1.2.7 For developers

Using the parameter ucs.disable it is possible to connect this VSM directly to the client without using the server. **This** parameter is for developers only and should not be changed.

1.3 Examples for video stream configuration

All configurations for described video devices are set by default. Be sure that all necessary drivers are installed.

1.3.1 Examples of supported devices.

Video source	Configuration properties (vstreamer.conf)
ArDrone camera	vstreamer.inputstream.0=Ardrone;tcp://192.168.1.1-:5555;60;640;360
GoPro camera	vstreamer.inputstream.1=GoPro;http://10.5.-5.9:8080/live/amba.m3u8;30
Web/USB camera	Autodetection: vstreamer.videodevices.autodetect=1 Set manually: vstreamer.videodevices.allow.0=USB Video Device vstreamer.videodevices.allow.1=Webcam

1.3.2 Examples of streaming to endpoint.

Stream type	Description
VLS stream	1. To run sample http stream use VLC. 2. Add to config file a string like this: vstreamer.inputstream.2=VLC;http://127.0.-0.1:8080;30

UDP Stream (H264)	1. To run sample UDP Stream use ffmpeg: “ffmpeg -f dshow -i video="USB Video Device" -vcodec libx264 -tune zerolatency -b 900k -f mpegts udp://127.0.0.1:1234” 2. Add to config file string like this: vstreamer.inputstream.2=UDP_Stream_Name;udp- ://127.0.0.1:1234;60
RTP Stream (H264)	1. To run sample RTP Stream use ffmpeg: “ffmpeg -f dshow -i video="USB Video Device" -vcodec libx264 -tune zerolatency -b 900k -f rtp rtp://127.0.0.1:1234” 2. To read from this stream, you need create a .sdp file which describes the stream. So create a test.sdp file: v=0 o=- 0 0 IN IP4 127.0.0.1 s=No Name c=IN IP4 127.0.0.1 t=0 0 a=tool:libavformat 56.1.100 m=video 1234 RTP/AVP 96 b=AS:900 a=rtpmap:96 H264/90000 a=fmtp:96 packetization-mode=1 3. Add to config file string with path to sdp like that: vstreamer.inputstream.2=RTP_Stream_Name;c- ://test.sdp

RTSP Stream (H264)	1. We checked this stream withan online sample rtsp://184.72.239.149/vod/mp4:BigBuckBunny_115k.-mov 2.Add to config file string like this: “vstreamer.inputstream.2=RTSP_Stream_Name; rtsp://184.72.239.149/vod/mp4:BigBuckBunny_115k.-mov”

1.3.3 Examples of broadcasting to web services.

After having correctly working video source you can stream to web services by using share button in video control of UgCS client. Supported services are Twitch, Youtube and Ustream.

Stream type	Description
Ustream	1. Register at http://www.ustream.tv/ . 2. Create channel. 3. Visit channel Broadcast settings and open Encoder setting. There you can find RTMP URL. Now you have URL for streaming looked like: RTMP URL/Stream key. 4. Put that URL in Ustream input box of UgCS video control and press OK. 5. Check your channel on Ustream.
Youtube	To be implemented.
Twitch	To be implemented.

1.4 Supported formats and devices

List of supported:

- video codecs - https://www.ffmpeg.org/general.html#Video-Codecs
- protocols - https://www.ffmpeg.org/general.html#Network-Protocols
- I/O devices - https://www.ffmpeg.org/general.html#Input_002fOutput-Devices