ADAPTIVE RECOGNITION

TRAFFIC MONITORING CAMERAS

Programmer's Guide

Digital LPR Cameras

.sf-sub-indicator
.cart-menu .cart-icon-w
-outer.transparent header#to
.sf-menu > li.current_page
.sf-menu > li.current-menu-
> ul > li > a:hover > .sf-sub
ul #search-btn a:hover span,
.sf-menu > li.current-menu-
hover .icon-salient-cart,.ascend
!important;color:#ffffff!impo
header#top nav>ul>li.but
-widget-area-toggle a i
-outer.transparent

ARH Digital LPR cameras

Programmer's Guide

Document version: v.3.2.

Table of Contents

1. GENERAL API SPECIFICATION

Every operation on the camera is done by setting and getting parameters. Camera functions are implemented by modules, modules are grouped in module groups. Camera parameters are organized in named sections within modules. Parameters are set and get by HTTP queries in the manner detailed in this chapter

1.1. SECTIONS

Querying available parameter sections:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?getsections&wfilter=1
ANSWER:

```
mimetype=text/plain\r\n
sections=default,sectionname1_sectionvalue1,sectionname2_sectionvalue2,..
.\r\n
```

1.2. PARAMETERS

Query of the parameter values:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?get[parameter1_name]&get[parameter2_name]&...&[sectionname]=[sectionvalue]&wfilter=1
ANSWER:

```
…
parameter1_name=parameter1_value\r\n
parameter2_name=parameter2_value\r\n
…
```

Note

- If there is no section, the default section is applied
- One or even more parameters can be queried (if these belongs to one section)

Querying values of all parameters of the default section:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?getall&wfilter=1

ANSWER:

```
mimetype=text/plain\r\n
sections=default\r\n
parametername1=parametervalue1\r\n
parametername2=parametervalue2\r\n
...
```

Querying values of all parameters of the given section:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?getall&[sectionname]=[sectionvalue]&wfilter=1

ANSWER:

```
mimetype=text/plain\r\n
parametername1=parametervalue1\r\n
parametername2=parametervalue2\r\n
...
```

Note

- Only one section can be queried at a time.

Setting parameter values:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?[parametername1]=[parametervalue1]&[parametername1]=[parametervalue1]&...&[sectionname]= [sectionvalue]&wfilter=1

ANSWER:
```
mimetype=text/plain\r\n
ecode=0\r\n
```

ANSWER:
```
mimetype=text/plain\r\n
ecode=-22\r\n
eparams=parametername1,parametername1\r\n
```

Note

- Add &save to the end of the query to have the parameters saved and thus made permanent, otherwise the changes will be lost upon restarting the camera
- If section is not defined, the default section will be modified
- More parameters (in the same section) can be get/set
- In the answer the following fields will be present:
    - If no error occurred:
      ecode = 0
    - If error occurred:
      ecode= Standard Error Code
      eparams=parametername1,parametername2,...

1.3. PARAMETER ATTRIBUTES

Query of parameter attributes:

GET:
http://[camera_ip]/[modulegroup]/[modulename]?query[parametername1]&query[parametername2]&...&[sectionname]=[sectionvalue]&wfilter=1

Note

- If there is no section defined, the default section is applied
- More parameters (of the same section) can be queried

http://camera_ip/modulegroup/modulename?queryall[&sectionname=sectionvalue]&wfilter=1

Note

- If there is no section defined, the default section is applied

ANSWER:

```
•     B - bound: B[parametername]=mininum,maximum,default,step
•     L - list: L[parametername]=value1,value2,value3,.. (where value1 is
the default value)
•     T - type
```

EXAMPLE

GET:
http://camera_ip/capture/scapture?queryshutter&wfilter=1

EXAMPLE – ANSWER:

```
...
Bshutter=25,33000,1000,0.001\r\n
Tshutter=rw,f,us\r\n
...
```

The bound answer:

bound[parameter_name]=mininum,maximum,default,step

- **minimum** value of the specified property (integer or floating-point number)
- **maximum** value of the specified property (integer or floating-point number)
- **default** (or recommended) value
- the amount of the steps to be set / advisable to set (helpful for programming)

Note

- Some/all of the fields minimum, maximum, default and step may be missing. This means the corresponding value is undefined.

Example for the bound:

- boundshutter=20,10000,1000,1

The value of the shutter can be set between 20 and 10000 in steps of 1, the recommended value is 1000.

- boundshutter=20,10000, ,1

The value of the shutter can be set between 20 and 10000 in steps of 1, there is no recommended value.

- boundshutter=20,10000,1000

The value of the shutter can be set between 20 and 10000, the recommended value is 1000. There are no steps, so in case of integer it is advisable to step one by one.

- boundshutter=20,10000

The value of the shutter can be set between 20 and 10000. There are no recommended value and steps.

The list answer:

L[parameter_name]=value1,value2,value3,.. (where value1 is the default value)

- Possible values in the list (the recommended value should be the first). The values can be integers or floating-point number and strings.

L[parameter_name]=$

- The value of the property is not examined (e.g.: username). Return of property, where neither the bound nor the list can be used.

Example for the list:

- Lifps=22.5,11.25

The set FPS value can be 22.5 and 11.25.
The recommended value is 22.5 (the first).

- Lifps=30,10,20,25,45,50,60

The settable FPS values: may be 10,20,25,30,45,50 and 60.
The recommended value is 30 (the first).

- Lcapture=prev,next,best,local

The prev, next, best or local values can be specified for the capture parameter.

- Lpassword=$

The password parameter can be whatever string (no listing).

The type answer:

T[parameter_name]=access,type,unit,reboot

- access: the mode of the access
  - r: read only
  - w: write only
  - rw: read-write
- type: type of the parameter
  - i8, i16, i32, i64: signed integer 8, 16, 32 and 64 bit type (default value: 32)
  - u8, u16, u32, u64: unsigned integer 8, 16, 32 and 64 bit type (default value: 32, only u is defined)
  - f32, f64: floating-point number (default value: 32, only f is defined)
  - s[number of the characters]: string type with specified character number (if it is not specified, the default will be 32). In case of 0 there are minimum and maximum, which is written in the bound.
  - With all these types a square bracket may be present, which means that an array should be used (e.g.: the i16[4] means 4 x 16-bit integer numbers).
- unit: unit of measurement (optional) - arbitrary string
- reboot: should the system be rebooted after rewriting the parameter

- not specified: no need to restart
- q: quick module restart (see Restart)
- f: complete reboot (see Restart)

Example for the type:

- Tshutter=rw,f,us

The shutter is readable and writable, floating-point type and its unit is microseconds. No restart is needed after rewriting the parameter, because it will be entered immediately.

- Timgshutter=r,f,us

The imgshutter is a read only, floating-point type property and its unit is microseconds.

- Tusername=rw,s32,,f

The username is a readable and writable string type containing 32 characters. It does not have measurement, but the system has to be restarted in case of switching.

1.4. ANSWER DOCUMENT

A HTTP query must be assembled according to the previous chapters. By default, the answer is a multipart document, consisting of a binary and a text part (e.g. an image in the binary and its parameters, like shutter gain, etc. in the textual part). A special parameter 'wfilter' may be used to control this behaviour:

- wfilter=1 only the text part,
- wfilter=2 only the binary part,
- wfilter=3 both parts will be returned by the camera.

The HTTP header gives information about parsing the message. The 'mimetype' field determines the type of the data in the answer.

- mimetype=application/octet-stream: general data
- mimetype=text/plain: standard text type (e.g. parameters)
- mimetype=image/jpeg: JPEG image

Parameters are always sent in text/plain format, row by row as [name]=[value] pairs. Rows are terminated by '\r\n'. For example:

```
      contrast=1\r\n
      shutter=1000\r\n
      gain=4\r\n
```

Parts in the multipart message are separated by a boundary string, for example:

---------------------------7327659421310868402068919661

After the separator the length and the type of the part will be written:

---------------------------7327659421310868402068919661
Content-Length: 93123
Content-Type: image/jpeg

The boundary will be written at the beginning of the answer, between the parts and at the end of the answer.

2. MODULES

2.1. USERS

Module: setup/users
Manual: Ch. 5.3 Users

Only administrators have access to this module.

Users are described by the following parameters:

- user id (uid)
- user name (uname)
- full name (ufull)
- access level (ulevel)
- password (passw1,passw2)

Get/Set access level of users without credentials

Parameter: guestlevel
Possible values: 0: Guest, 1: Viewer, 2: Normal, 3: Power, 4: Administrator

Example:
http://camera_ip/setup/users?wfilter=1&getall

Getting the list of users

Example:
http://camera_ip/setup/users?wfilter=1&getall

In the response for 'getall' we get the 'guestlevel' and lists of the user parameters as comma-separated values. Parameter values with the same position/index in the lists belong to the same user. Passwords cannot be obtained.

Response example:

```
guestlevel=0
useridlist=1001,1002
nicknamelist=user00,user01
fullnamelist=,
userlevellist=2,4
```

Add new user

To add a new user 'uid' must be set to 'new'. 'passw1' and 'passw2' must match.
http://camera_ip/setup/users?wfilter=1&getall&uid=new&uname=user00&ufull=&ulevel=2&passw1=x&passw2=x

Modify the parameters of a user

To modify the parameters of a user the corresponding 'uid' must be given with the modified parameters. 'passw1' and 'passw2' parameters can be omitted if we don't want to change the password.
http://camera_ip/setup/users?wfilter=1&getall&uid=1001&uname=user007&ufull=&ulevel=2&passw1=y&passw2=y

Delete user

To delete a user the user id ('uid') and the 'delete' parameter must be given.
http://camera_ip/setup/users?wfilter=1&getall&uid=1001&delete

2.2. IMAGE RESOLUTION

Module: capture/scapture
Manual: Ch. 6.1 Imaging Pipeline

Get maximum and set resolutions

GET:
http://camera_ip/capture/scapture?getall&wfilter=1[&source=X

ANSWER:

```
...
imgcapxsize=1280\r\n (maximum)
imgcapysize=960\r\n ...
imgxsize=640\r\n (current)
imgysize=480\r\n
...
```

Note

- The source # 1 is the default (primary image source) if there is no source parameter.
- In this property list, each value is listed twice: with and without the 'img' prefix. The parameter value with the 'img' prefix represents the value of the parameter at the time the image was captured, while the one without it represents the actual value.

2.3. SIMPLE RETRIEVAL OF AN IMAGE

Module: capture/scapture
Manual: Ch. 6.1 Imaging Pipeline

Capture an image

GET:
http://camera_ip/capture/scapture?wfilter=0[&source=X
http://camera_ip/capture/scapture?capture=best&framemonotimems=98741855&wfilter=0[&source=X
http://camera_ip/capture/scapture?capture=best&frameindex=8390&wfilter=0[&source=X
http://%3C%3Ccam_ip%3E%3E/capture/scapture?sensor=2
http://<<Cam_IP>>/capture/capture?sensor=2

Note

- The source # 1 is the default if there are no source parameters (primary image source).
- See also #4.Image/Image Parameter Queries

2.4. IMAGE/IMAGE PARAMETER QUERIES

Module: capture/scapture
Manual: Ch. 6.1 Imaging Pipeline

Get all image parameters

GET:
http://camera_ip/capture/scapture?getall&wfilter=1[&source=X
http://camera_ip/capture/scapture?queryall&wfilter=1[&source=X
http://camera_ip/capture/scapture?getall&wfilter=1&capture=next[&source=X

Note

- The source # 1 is the default (primary image source) if there are no source parameters.
- In this property list, each value is listed twice: with and without the 'img' prefix. The parameter value with the 'img' prefix represents the value of the parameter at the time the image was captured, while the one without it represents the actual value.

The scapture module returns three parameters for every captured frame: imgframeindex, imgframetimems and imgframemonotimems.

imgframeindex: After camera startup, captured frames are counted. This parameter provides the number of the frame.
imgframetimems: Time elapsed since 1st of January 1970 until the capturing of the frame in milliseconds.
imgframemonotimems: Time elapsed since camera startup until the capturing of the frame in milliseconds.

The above parameters can be used to determine the time of capturing when getting frames/parameters from the scapture module.

Querying the image buffer

capture=last

The camera returns the last captured image/parameter.
GET:
http://camera_ip/capture/scapture?capture=last&wfilter=

capture=next

It returns the frame/frame's parameter that follows the last returned (queried) one. If this frame is not in the buffer anymore, (too much time has elapsed and it has been rewritten by newer images), it returns the oldest image from the buffer.
The **capture=next** can be used by itself or together with the framemonotimems, imgframetimems and frameindex parameters: In these cases, frames, following the specified number/timestamp are returned (if they are still in the buffer):

GET:
http://camera_ip/capture/scapture?capture=next&framemonotimems=987418558&wfilter=
http://camera_ip/capture/scapture?capture=next&frameindex=8390&wfilter=

capture=best

It returns the frame/frame's parameter that is closest (before or after) to the specified timestamp or frame index. If the buffer does not contain such a frame, the oldest one is returned. If the specified timestamp (framemonotimems, imgframetimems) or frameindex value is too high (refers to a yet non-existing frame) then the newest frame from the buffer is returned.

GET:
http://camera_ip/capture/scapture?capture=best&framemonotimems=106964188&wfilter=1
http://camera_ip/capture/scapture?capture=best&frameindex=193390&wfilter=1

Note

- Use 'wfilter' to get either or both the image and its parameters.

SET:
http://camera_ip/capture/scapture?parametername1=parametervalue1&parametername2=parametervalue2&...&wfilter=1[&source=X

SAVE:
http://camera_ip/setup/profiles?save_imgparams&wfilter=1[&c_source=X

2.5. FREQUENTLY USED IMAGE PARAMETERS

Module: capture/scapture
Manual: Ch. 6.1 Imaging Pipeline

Get image parameters

GET:
http://camera_ip/capture/scapture?getall&wfilter=1[&source=X

ANSWER:

```
…
imggamma=1.700\r\n
gamma=1.700\r\n
…
```

SET:
http://camera_ip/capture/scapture?gamma=1.0&brightness=0.0&contrast=0.0&jpegquality=75&blacklevel=0.0 &whitelevel=1.0&whitebalance=0.0,0.0,0.0&saturation=0.0&wfilter=1[&source=X]

SAVE:
http://camera_ip/setup/profiles?save_imgparams&wfilter=1[&c_source=1

2.6. CAPTURE PARAMETERS

Module: capture/scapture
Manual: Ch. 6.1 Imaging Pipeline

Get and set capture parameters (shutter, gain, etc.)
GET:
http://camera_ip/capture/scapture?getall&source=1&wfilter=1

ANSWER:

```
…
imggain=2.000\r\n
imgshutter=4000.000\r\n
imgllm=3\r\n
...
gain=2.000\r\n
shutter=4000.000\r\n
llm=3\r\n
…
```

SET:
http://camera_ip/capture/scapture?gain=2.5&shutter=2000&wfilter=1

SAVE:
http://camera_ip/setup/profiles?save_imgparams&wfilter=1

2.7. AUTO BRIGHTNESS CONTROL

Module: control/brightnesscontrol
Manual: Ch. 6.3 Brightness Control

Get/Set Brightness Control Properties

GET:
http://camera_ip/control/brightnesscontrol?getall&wfilter=1
http://camera_ip/control/brightnesscontrol?queryall&wfilter=1
SET:
http://camera_ip/control/brightnesscontrol?enable=1&save&wfilter=1
http://camera_ip/control/brightnesscontrol?target=-17&save&wfilter=1

2.8. AUTO FOCUS

Module: control/focuscontrol
Manual: Ch. 6.5 Optics

Get autofocus state and properties
GET:
http://camera_ip/control/focuscontrol?getall&wfilter=1

ANSWER:

```
…
mode=0\r\n
roi=0,0,1280,960\r\n
…
```

SET:
http://camera_ip/control/focuscontrol?roi=0,0,100,200&save&wfilter=1
http://camera_ip/control/focuscontrol?mode=1&wfilter=1

Note

- during focusing mode will equal 1, when the focusing is done resets to 0
- roi: operation area of the autofocus; x1, y1, x2, y2, where x1, y1 the upper left corner of the rectangle and x2, y2 the lower right corner of the rectangle

2.9. AUTO COLOR COMPENSATION

Module: control/colcorr
Manual: Ch. 6.2.2 Color

Get color correction state and properties
GET:
http://camera_ip/control/colcorr?getall&wfilter=1

ANSWER:

```
…
mode=0\r\n
…
```

SET: http://camera_ip/control/colcorr?mode=1&save&wfilter=1

2.10. OPTICS CONTROL

Module: hwlayer/extopt, control/focuscontrol
Manual: Ch. 6.5 Optics

Operate motorized optics (zoom, focus, iris, filter exchanger)
GET:
http://camera_ip/hwlayer/extopt?getall&wfilter=1

ANSWER:

```
…
focus_pc=56\r\n
zoom_pc=28\r\n
iris_pc=100\r\n
zfcomovement=1\r\n
filter=0\r\n
…
```

Add zoom/focus presets
GET:
http://camera_ip/control/focuscontrol?getall&wfilter=1

ANSWER:

```
…
zfc_mode=0\r\n
zfc_pairs=564,327;1120,213;\r\n
…
```

SET:
http://camera_ip/hwlayer/extopt?zoom_pc=10&focus_pc=20&iris_pc=30&zfcomovement=0&filter=1&save&wfilter=1
http://camera_ip/control/focuscontrol?zfc_addpair=1120,213&save&wfilter=1
http://camera_ip/control/focuscontrol?zfc_rmpair=1120,213&save&wfilter=1
http://camera_ip/control/focuscontrol?zfc_mode=2&save&wfilter=1

Note

- extopt/zfcomovement (can be 0,1) is equal to focuscontrol/zfc_mode, recommended to use zfc_mode (can be 0,1,2)

2.11. FLASH CONTROL

Module: hwlayer/cperiph, config/ledpc
Manual: Ch. 6.7 Flash Control

Get current settings

GET:
http://camera_ip/config/ledpc?getall&wfilter=1
http://camera_ip/hwlayer/cperiph?getall&wfilter=1

ANSWER:

```
...
mimetype=text/plain\r\n
sections=default,panel_0,panel_1,panel_2\r\n
npanels=3\r\n
...
```

Note

- Index of the internal flash index: panel 0; external illuminators: panel 1, 2, ...
  npanels = 3, in this case 2 external illuminators are connected
- http://camera_ip/config/ledpc?getall&panel=0&wfilter=1

ANSWER:

```
...
mimetype=text/plain\r\n
led_diff_pc=-1\r\n
led_pcdose=0\r\n
...
```

SET:
http://camera_ip/config/ledpc?led_diff_pc=20&led_pcdose=10&panel=0&save&wfilter=1
http://camera_ip/config/ledpc?led_diff_pc=-1&led_pcdose=0&panel=0&save&wfilter=1

http://camera_ip/hwlayer/cperiph?flashpwr=2&ftime_even=200&panel=0&ledena=1&wfilter=1
http://camera_ip/hwlayer/cperiph?ftime_odd=100&panel=0&fpfena=1&wfilter=1
http://camera_ip/hwlayer/cperiph?ledena=0&fpfena=0&panel=0&save&wfilter=1

Note

- In case of cameras equipped with rolling shutter sensors the camera illumination can be only enabled or disabled. For enabling: led_pcdose=100

2.12. EVENT MANAGER

Module: trigger/eventman
Manual: Ch. 7.1.1 Event Manager

Get available trigger sources

GET:
http://camera_ip/trigger/eventman?getsources&wfilter=1

ANSWER:

```
…
mimetype=text/plain\r\n
sources=__motdet[3](Hardware Motion
Detection),trigger/swtrigger[1](Software
Trigger),trigger/uarttrigger[1](UART Trigger),trigger/gpiotrigger[1](GPIO
Trigger),trigger/scheduler[2](Scheduler Trigger)\r\n
…
```

General syntax

```
sources=groupname/modulename[numberofoutputs](stringname),groupname/modul
ename[numberofoutputs](stringname),...\r\n
```

Get registered trigger sources and input configuration

http://camera_ip/trigger/eventman?getconfig&wfilter=1

ANSWER:

```
 …
mimetype=text/plain\r\n
config=trigger/swtrigger/0:A,__motdet/0:B,trigger/uarttrigger/0:C\r\n
…
```

GENERAL SYNTAX:

```
config=
groupname/modulename/outputindex:assignedletter,groupname/modulename/outp
utindex:assignedletter\r\n
```

Get/set formula

http://camera_ip/trigger/eventman?getformula&wfilter=1

ANSWER:

```
…
mimetype=text/plain\r\n
formula=C\r\n
…
```

Get/set state

http://camera_ip/trigger/eventman?getstate&wfilter=1

ANSWER:

```
…
mimetype=text/plain\r\n
state=1\r\n
…
```

SET:

Register

http://camera_ip/trigger/eventman?register=__motdet/0:A&wfilter=1
http://camera_ip/trigger/eventman?register=__motdet/1:B&wfilter=1
http://camera_ip/trigger/eventman?register=trigger/swtrigger/0:C&wfilter=1
http://camera_ip/trigger/eventman?register=trigger/uarttrigger/0:D&wfilter=1
http://camera_ip/trigger/eventman?register=trigger/gpiotrigger/0:E&wfilter=1
http://camera_ip/trigger/eventman?register=trigger/scheduler/0:F&wfilter=1
http://camera_ip/trigger/eventman?register=trigger/scheduler/1:G&wfilter=1

Note

- The digit ID of each trigger source must be unique during the registration (recommended alphabetically, see above). The system will manage it automatically during registration via web interface.

Start module

http://camera_ip/trigger/eventman?formula=B&state=1&save&wfilter=1

Stop module

http://camera_ip/trigger/eventman?state=0&save&wfilter=1

Reset module

http://camera_ip/trigger/eventman?state=2&save&wfilter=1

2.13. MOTION DETECTION TRIGGER

Module: capture/scapture
Manual: Ch. 7.1.3 Motion Detector

Get/set motion detection parameters

GET:
http://camera_ip/capture/scapture?queryall&wfilter=1
http://camera_ip/capture/scapture?getall&wfilter=1&follow=0&frame=2

ANSWER:

```
…
mdsensitivity=75\r\n
mdspeed=75\r\n
mdreslevel=50\r\n
mdtriglevel=50,50\r\n
```

SET:
http://camera_ip/capture/scapture?mdsensitivity=80&mdspeed=80&mdreslevel=55&mdtriglevel=55,55&wfilter=1&follow=0&frame=0

SAVE:
http://camera_ip/setup/profiles?wfilter=1&save_motdets

2.14. SOFTWARE TRIGGER

Module: trigger/swtrigger
Manual: Ch. 7.1.4 Software trigger

Get/set software trigger properties
GET:
http://camera_ip/trigger/swtrigger?getall&output=0&wfilter=1

ANSWER:

```
…
start_offs=0\r\n
end_offs=0\r\n
…
```

SET:
http://camera_ip/trigger/swtrigger?start_offs=100&end_offs=100
http://camera_ip/trigger/swtrigger?sendtrigger=1&wfilter=1
http://camera_IP/trigger/swtrigger?start_offs=-100&end_offs=100&save&output=0&wfilter=1

It is possible to attach max. 256 bytes of data to a software trigger (e.g. a trigger ID, device measurement data, etc.):
http://camera_ip/trigger/swtrigger?sendtrigger=1&trigger_data=TRIGGER_0057&wfilter=1

- This piece of data will be inserted in the uploaded data file (See 18. Upload Manager) with parameter name 'triggerdata'.
- In case of ANPR capable devices, trigger data will be available addressing the database column TRIGGER_DATA (See 21. Database contents)

In both cases, trigger data will be encoded with Base64.

2.15. UART TRIGGER

Module: trigger/uarttrigger
Manual: Ch. 7.1.7 UART Trigger

Get/set UART trigger properties
GET:
http://camera_ip/trigger/uarttrigger?getall&wfilter=1
http://camera_ip/trigger/uarttrigger?getall&output=0&wfilter=1

SET:
http://camera_ip/trigger/uarttrigger?baudrate=9600&bytesize=8&parity=0&stopbits=1&start_token=256&end_token=10&start_offs=0&end_offs=0&mode=0&save&output=0&wfilter=1

Note

start_token=256 means: Start on first byte (decimal value)

- Start/End token can be decimal and hexadecimal (in case of hexadecimal the 0x prefix is needed)
- The website returns the values in decimal

2.16. GPIO TRIGGER

Module: trigger/gpiotrigger
Manual: Ch. 7.1.5 GPIO Trigger

Get/set GPIO trigger properties
GET:
http://camera_ip/trigger/gpiotrigger?getall&wfilter=1
http://camera_ip/trigger/gpiotrigger?getall&output=0&wfilter=1
http://camera_ip/trigger/gpiotrigger?getgpin&wfilter=1
http://camera_ip/trigger/gpiotrigger?getgpout&wfilter=1

SET:
http://camera_ip/trigger/gpiotrigger?gpout=1&wfilter=1
http://camera_ip/trigger/gpiotrigger?gpout=0&wfilter=1
http://camera_ip/trigger/gpiotrigger?gpout=0&samplerate=100&reqsamples=10&act_level=1&start_offs=0&end_offs=0&mode=0&output=0&save&wfilter=1

2.17. SCHEDULER TRIGGER

Module: trigger/scheduler
Manual: Ch. 7.1.6 Scheduler Trigger

Get/set scheduler trigger properties
GET:
http://camera_ip/trigger/scheduler?getall&wfilter=1&output=0
http://camera_ip/trigger/scheduler?getall&wfilter=1&output=1

SET:
http://camera_ip/trigger/scheduler?enabled=1&mode=1&wday=mon;tue;wed;thu;fri;sat;sun;&day=every&hour=every&min=0;5;10;30-35;50;&sec=0;5;50-55;&output=0&save&wfilter=1
http://camera_ip/trigger/scheduler?enabled=0&output=1&save&wfilter=1

2.18. UPLOAD MANAGER

Module: upload/uploadman
Manual: Ch. 7.1.9 Result Upload

Get log of event upload attempts

GET:
http://camera_ip/upload/uploadman?getlog

ANSWER:

```
…
2,15-13/2221093.jpg,0,1,Couldn't resolve host name;0,15-
13/2331476.jpg,0,0,No error;0x00
…
```

GENERAL SYNTAX:

```
     protocol,filenametemplate,include,ecode,estring;...;0x00

     protocol: [ftp:0, smtp:1, http:2, gxdctrf:3, invalid:4]
     content: [image+data: 0, image: 1, data: 2]
     ecode: [0: OK, 1: failed, 2: image OK, data failed]
```

Note

- If no log information exists, 1 byte of data will be returned (0x00)
- At the end there will be one 0x00 byte even if there is log information.

Upload data to FTP server

GET:
http://camera_ip/upload/uploadman?getall&wfilter=1

ANSWER:

```
…
sections=default,protocol_smtp,protocol_ftp,protocol_post\r\n
currentprotocol=0\r\n
…
```

http://camera_ip/upload/uploadman?getall&wfilter=1&protocol=ftp
http://camera_ip/upload/uploadman?content=0&currentprotocol=0&host=testserver&username=&password=&filenametemplate=$d-$h/$m$s$l&protocol=ftp&save&wfilter=1

Upload Data to SMTP Server (e-mail)

GET:
http://camera_ip/upload/uploadman?getall&wfilter=1

ANSWER:

```
...
sections=default,protocol_smtp,protocol_ftp,protocol_post\r\n
currentprotocol=0\r\n
...
```

http://camera_ip/upload/uploadman?getall&wfilter=1&protocol=smtp
http://camera_ip/upload/uploadman?content=0&currentprotocol=1&host=testserver&username=&password=&from=&to=user1@testserver;user2@testserver&filenametemplate=$d-$h/$m$s$l&protocol=smtp&save&wfilter=1

Upload Data to HTTP Server

GET:
http://camera_ip/upload/uploadman?getall&wfilter=1

ANSWER:

```
...
sections=default,protocol_smtp,protocol_ftp,protocol_post\r\n
currentprotocol=0\r\n
...
```

http://camera_ip/upload/uploadman?getall&wfilter=1&protocol=post
http://camera_ip/upload/uploadman?content=0&currentprotocol=2&host=testserver&filenametemplate=$d-$h/$m$s$l&protocol=post&save&wfilter=1

2.19. SET/GET ANPR PROPERTIES

Module: lpr/cff
Manual: Ch. 7.2.1 Browse

> **Note**
>
> This module will deliver its response in the binary part either as an XML document or a JPEG file.

Get a property of the License Plate Recognition module
GET:
http://[camera_ip]/lpr/cff?cmd=getproperty&name=[property path]

ANSWER:

```
…
<ans>\r\n
     <property>\r\n
          <[property name] value ="[property value]"/>\r\n
     <property/>\r\n
</ans>
…
```

Set a property of the License Plate Recognition module
GET:
http://[camera_ip]/lpr/cff?cmd=setproperty&name=[property path]&value=[property value]

ANSWER:

```
…
<ans>\r\n
     <ecode value ="0"/>\r\n
</ans>
…
```

Save properties of the License Plate Recognition module
GET
http://[camera ip]/lpr/cff?cmd=saveproperties

ANSWER:

```
…
<ans>\r\n
<ecode value="0"/>\r\n
</ans>…
```

2.20. BROWSE THE DATABASE

Module: lpr/cff
Manual: Ch. 7.2.1 Browse

> **Note**
>
> This module will deliver its response in the binary part either as an XML document or a JPEG file.

Get last record ID

GET:
http://camera_ip/lpr/cff?cmd=getid&id=last

ANSWER:

```
…
<ans>\r\n
<ecode value="0"/>\r\n
<id value="1396268058242"/>\r\n
</ans>
…
```

Get data file corresponding to ID

GET:
http://camera_ip/lpr/cff?cmd=getdata&id=last
http://camera_ip/lpr/cff?cmd=getdata&id=1396268058242

ANSWER:

```
…
<result>
      <location value="-"/>
      <cameraid value="00001009"/>
      <ID value="1396306892338"/>
      <image_hash value="D834B424BBE513390F635AD59E8EDED796D978F0"/>
      <capture>
            <frametime value="2014.04.02 3:59:16.733"/>
            <frametimems value="1396403956733"/>
            <frameindex value="0"/>
      </capture>
      <anpr>
            <text value="ARH002"/>
            <type value="111"/>
            <frame value="578,443,734,412,730,445,574,476"/>
            <bgcolor value="16777215"/>
            <color value="0"/>
            <confidence value="94"/>
            <timems value="360"/>
            <resultcnt value="1"/>
      </anpr>
      <motdet>
```

```
          <rect value="0,0,1,1"/>
          <confidence value="100"/>
          <objectid value="22257"/>
          <objectix value="4"/>
     </motdet>
     <trigger>
          <speed value="52.00"/>
          <speed_limit value="1.00"/>
          <direction value="1"/>
          <category value="679"/>
          <vclass value="1"/>
          <timems value="0"/>
     </trigger>
     <misc>
          <gps_lat value="47.4930"/>
          <gps_lon value="19.0234"/>
     </misc>
</result>
```

Get image corresponding to ID

GET:
http://camera_ip/lpr/cff?cmd=getimage&id=last
http://camera_ip/lpr/cff?cmd=getimage&id=1396268058242
http://camera_ip/lpr/cff?cmd=getimage&id=1396268058242 (cropped license plate image, will not exist, if plate was not found)

2.21. DATABASE CONTENTS

The following table lists the data stored to each event

Field	Type	Description
ID	BIGINT	Primary key/Event ID
ANPR_BGCOLOR	INTEGER	License plate background color
ANPR_COLOR	INTEGER	License plate text color
ANPR_CONFIDENCE	INTEGER	License plate confidence
ANPR_FRAME	TEXT	License plate coordinates (x0,y0,...,x3,y3)
ANPR_RESULTCNT	INTEGER	Number of identical readings
ANPR_TEXT	TEXT	License plate text (encoded)
ANPR_TIMEMS	INTEGER	Recognition time [ms]
ANPR_TYPE	INTEGER	License plate jurisdiction code
COUNTRY_LONG	TEXT	License plate country – textual, long format
COUNTRY_SHORT	TEXT	License plate country – textual, short format
FRAMEINDEX	UNSIGNED INTEGER	Image frame index
FRAMETIMEMS	BIGINT	Image /event timestamp [ms]
HASH	TEXT	Image SHA1 hash
IMGSIZ_AUX_X	INTEGER	Overview image X size
IMGSIZ_AUX_Y	INTEGER	Overview image Y size
IMGSIZ_LP_X	INTEGER	License plate image X size
IMGSIZ_LP_Y	INTEGER	License plate image Y size
IMGSIZ_NORM_X	INTEGER	Normal image X size
IMGSIZ_NORM_Y	INTEGER	Normal image Y size
IMGSIZ_STP_X	INTEGER	Trigger strip image X size
IMGSIZ_STP_Y	INTEGER	Trigger strip image Y size
MD_OBJECTID	UNSIGNED INTEGER	Event ID
MD_OBJECTIX	UNSIGNED INTEGER	Image index (nth frame of the event)
MD_RECT	TEXT	Motion frame (left,top,right,bottom)
MISC_GPS_LAT	TEXT	GPS coordinate (latitude)
MISC_GPS_LON	TEXT	GPS coordinate (longitude)
STATE_LONG	TEXT	License plate state – textual, long format
STATE_SHORT	TEXT	License plate state – textual, short format
TRIGGER_CATEGORY	INTEGER	Vehicle category
TRIGGER_DATA	BLOB	Trigger data (Base64 encoded)
TRIGGER_DATALEN	INTEGER	Tirgger data length
TRIGGER_DIRECTION	SMALLINT	Vehicle heading (textual, see
TRIGGER_SOURCE	TEXT	Trigger source
TRIGGER_SPEED	UNSIGNED INT	Vehicle speed
TRIGGER_SPEEDLIMIT	INTEGER	Vehicle speed limit
TRIGGER_VCLASS	INTEGER	Vehicle e-length

2.22. QUERY DATABASE

Module: lpr/cff
Manual: Ch. 7.2.1 Browse

Note

This module will deliver its response in the binary part either as an XML document or a JPEG file.

Steps of retrieving results from the database:

1. Query IDs with regular SQL commands formulating searching criteria
2. Get image and data of the events corresponding to the returned IDs

(see Browse the database Programmers section)

Executing an SQL query

GET:
http://camera_ip/lpr/cff?cmd=getresultlist&select=WHERE ID>0

ANSWER:

```
<results>
<result_0 value="1396465612217"/>
<result_1 value="1396465612218"/>
...
<result_999 value="1396465613216"/>
<n_results value="1000"/>
</results>
```

Note

- default value of max_record parameter is 1000. This value can be increased using the default/cfs/db/max_record parameter

GET:
http://camera_ip/lpr/cff?cmd=getresultlist&select=WHERE ID>1396465612672 AND ID<1396465612680
http://camera_ip/lpr/cff?cmd=getresultlist&select=WHERE ANPR_TEXT LIKE '%5%'

Note

- use '%25' instead of '%' in the web browser
  ex.: ... LIKE '%255%25'

GET:
http://camera_ip/lpr/cff?cmd=getresultlist&select=WHERE TRIGGER_SPEED>5000

Note

- 'speed' and 'speed limit' values are stored as fixed point numbers (multiplied by 100), thus SQL queries must be formulated accordingly: 50.00 km/h should be entered as 5000

2.23. TEMPLATES

The content of the result data file (and the file name) is customizable using templates. The following short example, which will describe a result formatted as an XML contains all features:

```
1      <?xml version="1.0" encoding="UTF-8"?>
2     <result>
3          <ID value ="$(ID);"/>
4          <text value="$DB2XML($(ANPR_TEXT));"/>
5          <location value="$(location);"/>
6           <image value="$(normal_img);"/>
7     </result>
```

Lines #1, #2 and #7 are text only (of course in this example we have to adhere to the XML standard). Line #3 is an example of a `database field`, line #4 is an example of a `function`, line 5 and 6 are examples of a `property/special field`. This template may evaluate to something like:

```
<?xml version="1.0" encoding="UTF-8"?>
<result>
<ID value ="123456789"/>
<text value="ABC123"/>
<location value="Test Site 34b"/>
<image value="/9j/4AAQSkZJRgABAQAAAQABA[... a base 64 encoded
image...]"/>
</result>
```

The same content in a different format might be:

```
event_id=$(ID);
plate_text=$DB2XML($(ANPR_TEXT));
camera_location=$(location);
vehicle_image=$(normal_img);
```

which will evalutate to:

```
event_id=123456789
plate_text=ABC123
camera_location=Test Site 34b
vehicle_image=/9j/4AAQSkZJRgABAQAAAQABA[... a base 64 encoded image...]
```

Database fields

The contents of the database (see Chapter ...) can be injected using the following syntax:
$([database field id])

Functions

To format the output the following functions are provided, with the general syntax:
$[function name]([argument1],[argument2],...,[argumentN]);

Plate text formatting:

DB2XML(*text,flags:optional*) - converts the license plate text *text* from DB to XML format

DB2JSON(*text,flags:optional*) - converts the license plate text *text* from DB to JSON format

DB2UTF8(*text,flags:optional*) - converts the license plate text *text* from DB to UTF8

The following flags are available:
ARABIC2LATIN - which converts arabic numbers to their latin equivalent, and non-numeric arabic to '.'.

Time formatting:
FormatTime(*timestamp,format_string:optional*) - formats timestamp given in milliseconds according to the optional format_string (see the POSIX strftime function) For example:
"$FormatTime($(FRAMETIMEMS),%Y%m%dT%H%M%S%z)"
evaluates to "20180419T145713+0200". If the format string is omitted, the same timestamp is evaluated as "2018.04.19 14:57:13.594".

Special fields

The following keywords are defined:
normal_img - the image representing the event, Base64 encoded
lp_img - the cropped license plate image, Base64 encoded
aux_img - the overview image, Base64 encoded
location - the location string as defined in default/cfs/db/location
cameraid - the camera HW id, may be overridden with default/cfs/db/cameraidstrip_img - image
strip_img - image strip Base64 encoded

Note

Special characters
The character '$' must always be escaped with '\'. Within an expression the characters '$', '(', ')', ',' and ';' have to be escaped with a '\' character. For example the following expression:

```
();$TEST(a\$a\,,$(LP));
```

will evaluate as

```
();a$a,DATA
```

provided that the function TEST concatenates its arguments and the field LP contains the text 'DATA'.

2.24. ENGINE MANAGER

Module: lpr/cff
Manual: Ch. 7.2.4 Engine Manager

Note

This module will deliver its response in the binary part either as an XML document or a JPEG file.

Default answer in case of error:

```
<ans>
  <ecode value="ERROR_CODE"></ecode>
  <estr value="ERROR_STRING"></estr>
</ans>
```

Query installed engines

GET:
http://[camera_ip]/lpr/cff?cmd=getenginelist

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
  <anpr_engines>
    <n_engines value="NUMBER_OF_ENGINES"></n_engines>
    <engines>
      <engine0 value="ENGINE_NAME"></engine0>
      ...
      <engineN value="ENGINE_NAME"></engineN>
    </engines>
  </anpr_engines>
</ans>
```

Query engine properties

GET:
http://[camera_ip]/lpr/cff?cmd=getengineproperties&engine=ENGINE_NAME

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
  <properties>
    <PROPERTY_NAME value="PROPERTY_NAME"></PROPERTY_NAME>
    ...
  </properties>
</ans>
```

Install an engine

ZIP files can be sent the same way the browser does with the '<input type="file" />' element. This is a HTTP POST request to the URL http://[camera_ip]/lpr/cff?cmd=engineupload and a zip file in content as form data.

Uninstall an installed engine (uninstalling default engines is not allowed)

http://[camera_ip]/lpr/cff?cmd=removeengine&engine=ENGINE_NAME

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
</ans>
```

Set the default engine for a group (default, normal, quick, post)

http://[camera_ip]/lpr/cff?cmd=setdefaultengine&engine=ENGINE_NAME&group=PROPERTY_GROUP

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
</ans>
```

Get the default engine for a group (default, normal, quick, post)

http://[camera_ip]/lpr/cff?cmd=getdefaultengine&group=PROPERTY_GROUP

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
  <default_engine value="ENGINE_NAME"></default_engine>
</ans>
```

Check if an engine is default for a group (default, normal, quick, post)
http://[camera_ip]/lpr/cff?cmd=isdefaultengine&engine= ENGINE_NAME

ANSWER:

```
<ans>
  <ecode value="0"></ecode>
  <default value="0 or 1"></default>
</ans>
```

2.25. RESULT UPLOAD

Module: lpr/cff
Manual: Ch. 7.2.5 Result Upload

Note

This module will deliver its response in the binary part either as an XML document or a JPEG file.

Upload settings can be managed by setting and querying properties with the setproperty and getproperty commands #19.Set/Get ANPR properties

The properties of the upload module reside in the 'default/cfs/uplm' property path:

```
<uplm>
  <general>
    <method value="-1"/>
    <maxtryconnect value="1"/>
  </general>
  <http>
    <host value="192.168.1.250"/>
    <content value="7"/>
    <timeoutms value="2000"/>
  </http>
  <ftp>
    <host value="192.168.1.250"/>
    <content value="7"/>
    <timeoutms value="2000"/>
    <username value="user"/>
    <password value="pass"/>
  </ftp>
</uplm>
```

`'general/method'`: Upload method.

Possible values:

- `-1: disabled`
- `0: HTTP`
- `1: FTP.`
- `2: SFTP`

`'general/maxtryconnect'`: Max. number of connection attempts. Possible values: 1-10

`'content'`: Bitfield that describes what data should be uploaded. Possible values: 1-7

- `bit 0: image (jpg)`
- `bit 1: plate image (jpg)`
- `bit 2: result (xml)`

example:

```
all data -> content=7 (=0b111),
image+result -> content=5 (=0b101)
```

`'timeoutms'`: Timeout of one attempt in milliseconds.

Examples:
SET `'maxtryconnect'`:
http://IP/lpr/cff?cmd=setproperty&name=/default/cfs/uplm/general/maxtryconnect&value=5

GET 'maxtryconnect':
http://IP/lpr/cff?cmd=getproperty&name=/default/cfs/uplm/general/maxtryconnect

ANSWER:

```
<ans>
  <property>
    <maxtryconnect value="1"/>
  </property>
</ans>
```

Possible error messages in ANPR LOG:

- UploadMan:: Cannot initialize curl.
- UploadMan:: Upload is disabled.
- UploadMan:: No HTTP host defined.
- UploadMan:: No FTP host defined.
- UploadMan:: No FTP username defined.

2.26. CLEAR DATABASE/REMOVE RECORD

Module: lpr/cff
Manual: Ch. 7.2.1 Browse

Note

This module will deliver its response in the binary part either as an XML document or a JPEG file.

Clear database or remove record
GET:
http://camera_ip/lpr/cff?cmd=cleardb
http://camera_ip/lpr/cff?cmd=removebyid&id=1396465617988

ANSWER

```
<ans>
<ecode value="0"/>
</ans>
```

2.27. MODULE RESTART

Module: lpr/cff
Manual:

Note

This module will deliver its response in the binary part either as an XML document or a JPEG file.

Restart license plate reading module

GET
http://[camera ip]/lpr/cff?cmd=restart

ANSWER:

```
<ans>\r\n
<ecode value="0"/>\r\n
</ans>
```

2.28. SYSTEM INFORMATION

Module: stat/identify, stat/system, hwlayer/cperiph
Manual: Ch. 8.1 System Info

Get information about system diagnostic values
GET:
http://camera_ip/stat/identify?getall&wfilter=1
http://camera_ip/stat/system?getall&section=memory&wfilter=1
http://camera_ip/hwlayer/cperiph?getcamtemp&panel=0&wfilter=1
http://camera_ip/hwlayer/sensors?getEv&device=lightsensor&wfilter=1

2.29. CAMERA LOG

Module: logging/logstream
Manual: Ch. 8.3 Camera Log

Get camera log
GET:
http://camera_ip/logging/logstream?getfilter&wfilter=1
http://camera_ip/logging/logstream?getlog

Note

- Lines are separated by '\n' (0x0a characters)
- The log will not be empty after the query, so the queried lines will remain even after the next query, unless it was not overwritten

(It is possible to set the already queried rows to be deleted.
See: plainconfig mode parameter)

SET:
http://camera_ip/logging/logstream?filter=uarttrigger&save&wfilter=1
http://camera_ip/logging/logstream?filter=*&save&wfilter=1

Note

- '*' no filtering is applied

2.30. RESTART

Restart the camera

GET:
http://camera_ip/setup/exit?retcode=1

Note

Return codes (retcode) available:

- 1: restart modules (quick restart)
- 3: reboot camera
- 7: reboot and enter recovery mode

2.31. CAMERA MODULES

List all modules currently loaded

GET:
http://camera_ip/stat/system?getall&section=module&wfilter=1

ANSWER:

```
…
mn00=logging/logstream\r\n
mv00=3.1.0.6,0\r\n
md00=Device Log Streamer\r\n
mn01=maintenance/backup\r\n
mv01=3.1.1.4,2\r\n
md01=Backup System\r\n
…
```

General syntax:

```
mnXX=group of the module/ name of module\r\n
mvXX=version of the module\r\n
mdXX=description of the module\r\n
where XX is the index of the module
```

Contact Information

Headquarters:
Adaptive Recognition Hungary Inc.
Alkotás utca 41 HU-
1123 Budapest Hungary
Phone: +36 1 201 9650
Fax: +36 1 201 9651
Web: www.adaptiverecognition.com

Service Address:
Adaptive Recognition Hungary Inc.
Ipari Park HRSZ1113/1 HU
2074 Perbál Hungary
Phone: +36 1 2019650
E-mail: rmarequest@adaptiverecognition.com

Adaptive Recognition Hungary Technical Support System (ATSS) is designed to provide you the fastest and most proficient assistance, so you can quickly get back to business.
For further technical information about our products, please visit our official website.

Information regarding hardware, software, manuals and FAQ are easily accessible for customers who previously registered to enter the dedicated ATSS site. Besides offering assistance, the site is also designed to provide maximum protection while managing your business information and technical solutions utilized.

New User
If this is your first online support request, please create an account by clicking on this link.

Returning User
All registered ATSS customers receive a personal access link via e-mail. If you previously received a confirmation message from ATSS, it contains the embedded link that allows you to securely enter the support site.

If you need assistance with login or registration, please contact atsshelp@adaptiverecognition.com for help.