FOTRIC 600 Series Software

Development Manual

V 1.3

Document Description

date	version	reviser	revised statement	Reviewers involved	Reviewer
8.9.2017	1.0				
2018.10	1.1				
2019.3	1.2				
2019.7	1.3				

Copyright Notice

table of contents

Contents

1 Summary

The FOTRIC 600 series is a thermal imaging product for customable industrial applications, all devices' control interfaces are based on a REST style Web Service, this document focuses on the specific interface definitions and specifications required to control the device.

Video stream capture functions can be developed further by referring to the Demo program in the SDK.

2 Terms and definitions

REST

Representational State Transfer (REST) is a set of architectural constraints and principles. It is important to note that REST is a design style rather than a standard, and is typically based on the use of existing widely popular protocols and standards such as HTTP, URIs, and XML (a subset of the Standard Universal Markup Language) and HTML (an application of the Standard Universal Markup Language).

REST defines a set of architectural principles by which you can design system resource-centric Web services, including how clients have written in different languages handle and transport resource state over HTTP. REST has become the dominant Web service design pattern in recent years considering the number of Web services that use it. The impact of REST on the Web has been so significant that it has widely replaced SOAP and WSDL-based interface design due to its considerable ease of use.

JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is based on a subset of ECMAScript (the JS specification developed by w3c) and uses a text format to store and represent data that is completely independent of the programming language. The simplicity and clarity of the hierarchy make JSON the ideal language for data interchange. The data format is not only easy to read and write by humans, but also easy to parse and generate by machines, and efficient for network transmission.

Black body radiation

Every object is constantly radiating, absorbing, and reflecting electromagnetic waves. The radiated electromagnetic waves are different in each waveband, that is, they have a certain spectral distribution. This spectral distribution is related to the properties of the object itself and its temperature and is therefore called thermal radiation. To study the laws of thermal radiation, physicists have defined an ideal object, the **black body**, as the standard object for the study of thermal radiation.

Emissivity

Emissivity, also called specific emissivity or emissivity coefficient, is the ratio of the emitted radiation flux of an object to the radiation flux of a blackbody at the same temperature. The emissivity of an object is related to the nature of the object, the surface condition (e.g. roughness, color, etc.), and is a function of temperature and wavelength

3

Basic functions

Configuration Set

Many parameters need to be configured by the user on the device, and the user often has to use different parameters for different scenarios. It is often necessary for the user to save and switch groups of parameters collectively.

All parameters contained in a configuration set are configuration set parameters, which can be saved and switched in groups by the user. In addition, configuration sets can support automatic switching via external events.

The current configuration of a device can be saved as a copy, which is named by the user. Users can call the copy by its copy name (see '/admin/cfgsets' for reference). The number of configuration sets a user can save vary depending on the device model.

Device Control Interface

The device interface uses RESTful HTTP web services, which allow for easy secondary development and debugging, and all functionality is implemented through a unified interface.

HTTP Service

The user accesses the device's HTTP service through port 10080, which acts as a proxy to communicate with internal standalone functional modules, and port 10081 to access streaming data (rfc2616). the HTTP service will authenticate the permissions for each access, via the HTTP Digest Access Authentication (rfc2617) protocol for authentication, see 7.

- HTTP request example

http://DEVADDR:10080/path-to-resource?arg1=1&arg2=2

http://DEVADDR:10081/path-to-stream

```
GET /path HTTP/1.1
Host: server
Accept: */*
Connection: keep-alive
Authorization: Digest username="guest",
               realm="REST-API.fotric.co.uk",
               nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
               uri="/admin/info",
               qop=auth,
               nc=00000001,
               cnonce="0a4f113b",
               response="6629fae49393a05397450978507c4ef1",
               opaque="5ccc069c403ebaf9f0171e9517f40e41
```

- Returning data carries error information

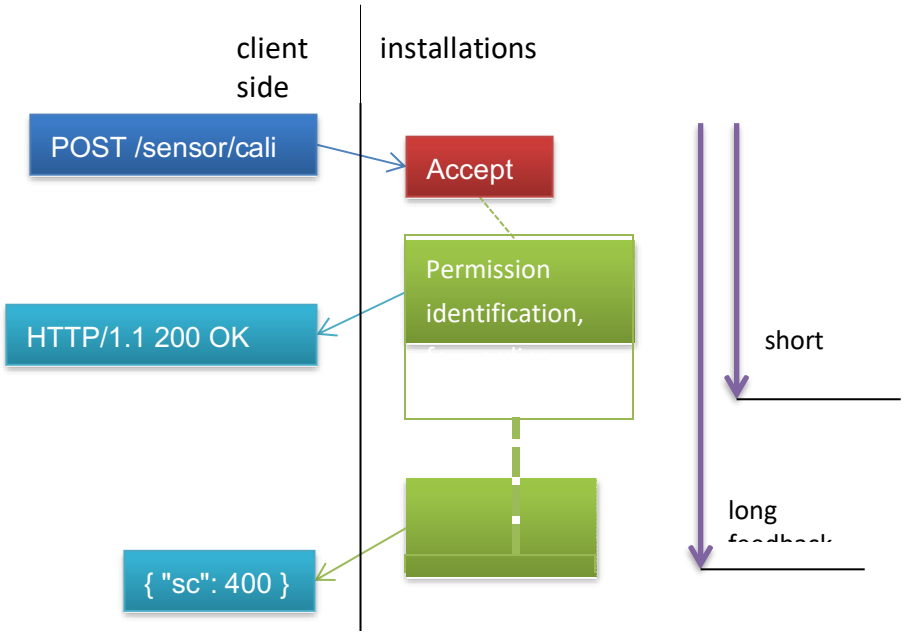
```
HTTP/1.1 200 OK
Content-Type: text/xml
Content-Length: 156
```

```
Connection: close
```

1.1. Response Method Extention

The device's response to the client extends the standard HTTP protocol, and the user receives 2 responses

The HTTP service returns the response message immediately according to the definition of the HTTP protocol, but at this point, the internal module has not yet responded. To use the HTTP approach to device control, the HTTP connection is maintained until the internal module returns or the connection times out. Therefore, the short and long feedback process is completed by first returning the HTTP message header, holding the connection, and then returning the message body.



When the HTTP service forwards the message successfully, the client receives a normal HTTP response with a status code of 200, and if the internal return is correct, the content returned is the body of the interface definition. If the interface returns an error, it will also be in JSON form.

```
{"sc": 400}
```

The value of sc is the same as the HTTP-defined return status code, see Appendix.

1.2. Cache redirection

Most interface commands transfer information in JSON format. When larger data transfers are required, access is redirected as a link, in the style of REST state transfer.

For example, to get a snapshot of a thermal image raw image

GET /isp/snapshot

```
{ "path" : "/file/cache/9aubeEZTjKYBj23k" }
```

The device saves the image in the device's cache, and the client can get the desired snapshot with the GET /file/cache/9aubeEZTjKYBj23k command.

Cached data should be read as soon as possible because the device's cache is finite, and when cache usage reaches its limit, older data is deleted, and data in the cache is lost when the device is turned off and rebooted. Writing to the cache too quickly may cause data to be deleted before

it can be used.

1.3. Null return

When the interface returns no content, the device returns an empty **HTTP** response, which is slightly different from the standard JSON, which will return 'null' as text.

The empty array "[ ]" and the empty object "{ }" are treated as equivalent to null in the interface as a return, so some interfaces may return a body of zero length.

1.4. JSON format extension definition

The string type in JSON used to access the device interface uses UTF-8 encoding instead of the UNICODE wide characters defined in the JSON standard.

1.5. JSON Configuration Tree

The configuration information inside the device will be accessed and modified by the user using different URIs as interfaces. The internal configuration information of the device in this document is defined in terms of JSON, which is called a configuration tree because of its tree-like hierarchy. All the information that can be modified for an interface is called the **Full Tree** for that interface. Removing the parts of the subtree and node in the Full Tree, what's left is the **Partial Tree for** that interface. And the empty tree is the partial Configuration Tree of any configuration tree. If a node in the configuration tree cannot be found in the full configuration tree with the same lookup path, it is called an **Unrelated Node**, and conversely, it is called a **Related Node**.

When the user reads the device information, the user is provided with the full configuration tree and additional read-only status information, and all of the internally available information for the corresponding interface is converted to JSON data.

When a user writes information to a device, the user can write either a full or partial configuration tree. The device first verifies the syntactic correctness of the JSON, then verifies the correctness of the relevant node data in the JSON configuration tree, and finally loads the data into the internal configuration and makes it effective. Configuration information not contained in the configuration tree is left as it was before it was written, while nodeless points written by the user are ignored, and will not cause the device to return an error.

Define the complete configuration tree for admin/ifaces/eth0 as follows.

```
{
    "dhcp" : false,
    "dns" :
    [
      "192.168.1.1"
    ],
    "gateway" :
    [
      "192.168.1.1"
    ],
    "ip" : "192.168.1.115",
    "netmask" : "255.255.255.0"
```

```
}
```

The user can write to parts of its configuration tree.

```
{
      "dhcp" : true
}
```

This turns on the DHCP function of the device. the other information remains the same. but the static IP is no longer functional. When the user writes the following data.

```
{
      "ip": "192.168.1.12",
      "foo":"bar"
}
```

The device's IP will be modified and the 'foo' field will be ignored.

1.6. Common return values

The following list of common error status codes for device interfaces. It applies to all interface paths

Return Code	description
401	Insufficient user privileges to access the interface
500	Internal device exceptions, usually an I/O exception has occurred (e.g., insufficient memory, etc.), or it could be a function module exception
400	Incorrect parameters of the input device

1.7. Rules for naming resources within a device

Users often need to create some resources on the device and read and write information about these resources through the device interface, such as adding user names, creating new zones for temperature measurement, creating new alarm rules, etc. These are identified by their corresponding names.

For example: creating a new user.

```
PUT /user/user1
```

The new user name is user1 and the naming of the user name should satisfy the requirements of the URL (RFC3986). The name of the resource to be created through the device interface shall conform to the following rules.

- Ascii-encoded printable characters
- Cannot use reserved characters: : / ? # [ ] @ ! $ & " ' * ( ) , = + ;
- The length of the resource name should be less than 1024 characters unless otherwise specified.

2. Basic concepts of thermal imaging cameras

All objects radiate electromagnetic waves, which are called thermal radiation. Thermal imagers pick up these waves and transfer them

into a digital image. Unlike the usual color photographs, thermal radiation images have a wide sampling dynamic range and are typically expressed as an integer of 14-16 bits, which we call the AD value (because this is the digital value obtained by digital-to-analog conversion).

According to the blackbody radiation principle, the temperature of an object can be deduced from the magnitude of the thermal radiation, and this is the principle of thermal imaging camera temperature measurement. The relationship between each AD value and the blackbody temperature is carefully calibrated during the production of the camera and is stored in the form of a temperature reference table (LUT) inside each device. Each device has multiple LUTs, which need to be selected correctly before the temperature measurement can be performed.

2.1. Temperature range

If higher temperatures are to be measured, the range of the camera will need to be changed. Choosing the right range for the camera will ensure that image sampling does not become saturated with overexposure.

A list of ranges is available through the following interface.

```
GET /sensor/t-range
```

Listings will vary by model, please consult your supplier

The picture quality will vary at different ranges, as higher temperature responsiveness will make lower temperature objects appear to be less clear.

2.2. Lens

Another factor that affects the temperature measurement is the lens used in the camera. The lens of each camera is calibrated before leaving the factory, so please do not change the pairing of the lens with the device, which may affect the temperature measurement accuracy.

If your model supports more than one lens, you can change lenses. But the device is to be configured. The list of supported lenses can be obtained by the following command

```
GET /sensor/lens
```

Listings will vary by model, please consult your supplier

2.3. Switching the temperature gauge

The temperature can only be measured correctly if the range and lens are selected correctly, the range and lens are selected by the list number, which starts from 1. If the 2nd range and the 1st lens are selected for temperature measurement.

PUT /sensor

```
PUT /sensor/jconfig


{
     "selected-lens" : 1,
     "selected-t-range" : 2
}
```

The user can obtain these range and lens combinations with the temperature gauge as follows.

```
GET /sensor/luts
```

Not all combinations have a corresponding temperature chart, this will need to be checked with the supplier before purchase. Each temperature comparison table can be downloaded separately:

```
GET /sensor/luts/[idx]?list
```

The current temperature gauge selected according to the range and lens affects the temperature output from the temperature measurement tool, and if you need to know which temperature gauge is currently in use, you can use the interface:

```
GET /sensor/lut
```

The user may select a range, lens combination via 'PUT /sensor/jconfig' that does not contain the corresponding factory temperature gauge, then an error will be returned when fetching the current temperature gauge.

2.4. Emissivity

The accurate temperature measurement of the thermal imaging camera depends not only on the collected AD value of the thermal radiation but also on the properties of the measured object itself, the most important property being the emissivity. The blackbody is a theoretical model and has an emissivity of 1. All objects have an emissivity of less than 1, and therefore need to be compensated for by parameter settings. See 4 for details of the settings.

2.5. Use of Temperature Gauges

The temperature table obtained from the device is the control table that converts the AD values

```
[
      {
         "r" : 7000,
         "t" : 10
      },
      {
         "r" : 7500,
         "t" : 20
      },
      {
         "r" : 8000,
         "t" : 30
      }
]
```

to temperature.

where r denotes the AD value and t denotes the corresponding Celsius temperature.

The order of the temperature table is always arranged from the lowest to the highest AD value. If the AD value to be converted is not included in the temperature table, it can be calculated by interpolating with adjacent AD values.

Suppose the adjacent terms of the AD value r in the $_a$temperature table are $r_0$and r_1, and their corresponding temperatures are $t_0$and t_1 then one interpolation gives the temperature t_a

$$t_a = t_0 + \frac{t_1 - t_0}{r_1 - r_0}(r_a - r_0)$$

All calculated temperatures refer to blackbody temperatures with an emissivity of 1.

3. Built-in temperature measurement tool

Each pixel of the thermal radiation image can be used as a temperature sampling point, but the amount of data is very large, and for the convenience of the user, the device provides a number of temperature tools to facilitate the selection of the location of interest for data acquisition.

The device supports three kinds of temperature measurement objects: point, area and line. The device has a global set of temperature measurement compensation parameters, each temperature measurement object can inherit the global parameters, or modify each parameter individually.

3.1. Temperature measurement compensation parameters

The temperature measurement compensation parameters supported within the device include.

- Emissivity
  Related to the nature of the material being measured, which has the greatest effect on the temperature measurement results, the emissivity of impractical materials can be found through the table of common material emissivities.

- ambient-t (ambient temperature)
  In addition to radiation emitted from the measured object itself, some radiations are reflected from the environment. For the unpolished object, this part of the radiation can be considered as a diffuse reflection of the ambient temperature.

- reflect-t (reflective temperature)
  When the measured object is surrounded by hot objects and this effect cannot be avoided, an additional correction for this strong interference is needed, i.e. compensation by the reflect-t parameter.

- distance
  The distance factor only becomes apparent when the object being measured is far away (several kilometers away) when the absorption of thermal radiation by the atmosphere needs to be compensated for, and the distance is measured in meters

- offset
  Offsets in the measured temperature values due to systematic or other influences that are not covered can be adjusted directly. Modification of this parameter requires the user to refer to a standard blackbody.

- lens-transmissivity
  When there are additional optical components in front of the device lens (e.g. the device is placed in a watertight compartment), the absorption of thermal radiation by the optical components affects the temperature measurement results and needs to be compensated for. Modification of this parameter requires the user to refer to a standard blackbody or to contact the supplier for calibration at the time of manufacture.

These temperature measurement parameters are applied to each temperature measurement object, while the device retains a global set of parameters as defaults. The global parameters can

```
PUT /isp/instrument/jconfig

{
     "ambient-t" : 20,
     "distance" : 1,
     "emissivity" : 0.92
}
```

be modified through the following interface.

To get the full global parameters, you can do this.

```
GET /isp/instrument/jconfig
```

3.2. Temperature measurement point

Users can set up temperature measurement points anywhere on the thermal imaging screen

```
PUT /isp/instrument/objects/points/p1


{
      "emissivity" : 0.97,
      "pos" :
      {
         "x" : 150,
         "y" : 50
      },
      "label": "My point 1"
}
```

other than the perimeter

The above command sets a temperature measurement point at column 150, row 50 of the image, and changes the emissivity to 0.97, which will overwrite the global emissivity value.

The label property is used to display a better readable object name on the OSD.

3.3. Temperature measurement area

The temperature measurement area can be defined by polygons or area masks.

Polygons with no less than 3 vertices and no more than 50 vertices, polygonal regions can be

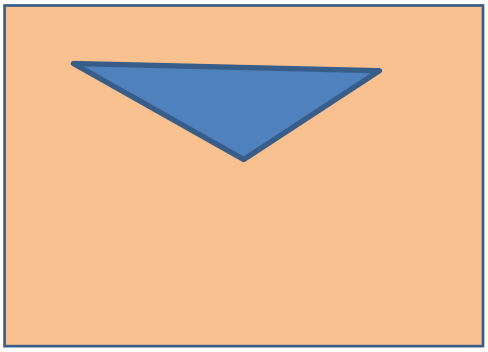
```
PUT /isp/instrument/objects/regions/r1


{
      "polygon" :
      [
         {
             "x" : 30,
             "y" : 50
         },
         {
             "x" : 300,
             "y" : 60
         },
         {
             "x" : 150,
             "y" : 120
         }
      ],
```

added as such:

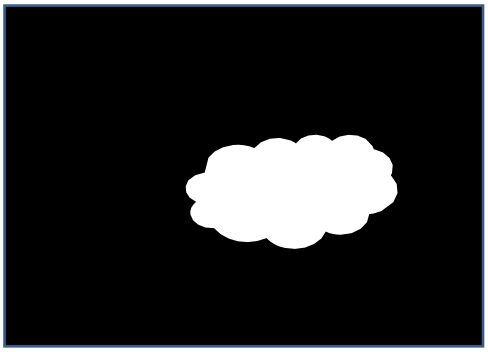
```
    "label" : "My Region1"
}
```

The above command will add the triangular area.



Note that the device only supports adding convex polygons.

The masked region will provide the user with a more flexible way of defining the region by uploading a masked image of a single connected region. The mask image should be the same size as the thermal camera resolution, with each pixel represented by an 8-bit byte, with non-zero values representing inside the region and zero representing outside the region. The masked region can only represent one irregular graph, and it can not be on the edge of the screen.



For a 384x288 device, the user needs to create a 384x288 byte cache and set the pixels in the corresponding area to non-zero values and upload them to the device cache:.

```
POST /file/cache
```

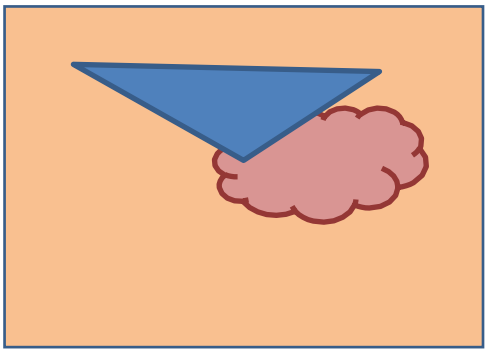
Upon successful upload, the device will return a cache token:

```
{
    "path" : "8kn4VBYmAw0XKUvI"
}
```

At this point, it is possible to create masked areas: the

```
PUT /isp/instrument/objects/regions/m1


{
    "mask": "8kn4VBYmAw0XKUvI"
    "label" : "My Mask Region1"
    "layer" : -1
}
```



In the above command, we have used the layer property, represented by an integer. Each region object has a layer attribute, the default value is 0. When there is overlap between regions, the regions with the same layer attribute will be added in the same order to determine the region's hierarchy, and the part of the region obscured by other regions will not participate in the region object's temperature measurement. For example, if we add regions a,b,c,d with layer 4,0,-1,0 respectively, then their layers from top to bottom are a,b,d,c

3.4. Thermometric polyline

For special applications where only a linear region of temperature sampling is required along a particular line, a polyline object can be utilized. The polyline object defines a polyline region with a series of vertices that are connected at the beginning and end, and the device collects the temperature by following the pixel grid through which the polyline passes (Bresenham algorithm).

The number of vertices of the polyline is not less than 2 and not more than 50, and the polyline

```
PUT /isp/instrument/objects/lines/l1


{
      "polyline" :
      [
         {
             "x" : 30,
             "y" : 50
         },
         {
             "x" : 300,
             "y" : 60
         },
         {
             "x" : 150,
             "y" : 120
         }
      ],
      "label" : "My Line1"
}
```

can be added like adding a polygon:

3.5. Global parameters

The simplest measurement is to get the highest and lowest temperature points on the entire image

```
GET /isp/instrument/objects/global?value
```

The global temperature values are corrected for temperature using the global temperature measurement compensation parameters.

3.6. OSD Display

Each temperature measurement area is overlaying on the video stream by default, and to turn off the OSD display of an object, you can do the following.

```
PUT /isp/instrument/objects/points/p1


{
      "osd-visible":false
}
```

Note that the above command will fail if the p1 object does not exist on the device, because the above command uses part of the configuration tree, and if the object does not exist, the full configuration tree does not exist.

3.7. Obtaining temperature data

Temperature measurement data are available separately for each object.

```
GET /isp/instrument/objects/lines/l1?value
```

Thus, the user can obtain the maximum and minimum temperatures and AD values on this polyline 'l1'

The user can connect a marker stream to obtain the measured value of the temperature measurement object in real-time, via port 10081.

```
/tag/values
```

See 6

3.8. Sampling rate

The temperature measurement tool can vary the rate at which the temperature is sampled,

```
PUT /isp/instrument/sample-rate


2
```

The sampling rate refers to the number of samples taken in 1 second, but of course, setting a rate greater than the frame rate of the camera is not valid.

3.9. Temperature reading via serial port

After setting the desired area through the API, the temperature measurement object can be bound to the MODBUS data address for serial temperature reading, see VI.3

```
PUT /isp/instrument/objects/regions/r1
```

```
{
      "mb-slot":1
}
```

The above command binds the area object r1 to MODBUS data block 1, which is accessed through the device's external serial port1, you can read r1's highest and lowest temperature.

3.10. Temperature Alarm

The equipment has an automatic alarm function, which can output the alarm signal through the optical coupler, and the user can modify the temperature measurement object 's "alarm-t" property to modify the alarm threshold; the alarm is off by default when the temperature measurement object is initially created.

An alarm signal is triggered when the maximum temperature in the m1 area exceeds 120 degrees.

Users can also customize other alarm methods, please contact the supplier for details.

4. Externally triggered video capture

The device can control the capture of thermal video using an external trigger signal. It is worth noting that thermal video capture requires a stable sampling frequency to obtain a stable image, and the image obtained by triggering is the nearest image obtainable after the trigger moment. Therefore, external triggering cannot only guarantee image acquisition being perfectly in sync with the operation.

How to switch to trigger acquisition:

```
PUT /capture/mode


"trigger"
```

In addition to being triggered by an external signal, the user can trigger acquisition by software.

```
POST /capture/trigger
```

See the installation manual for the connection of external signals.

5. HTTP-based real-time data streaming

In addition to redirecting file data, the device provides support for real-time data streaming.

According to the HTTP protocol, the end of the body is marked by the disconnection of the connection (the Connection header must be closed) when no header information defining the body length, such as Content-Length, is defined in the HTTP request header. Therefore, it is possible to use this method to send a continuous stream of data to the client until the client aborts reception or the device is reset.

The data stream is made of consecutive data frames, and to delimit the data frames, the following 8-byte segmented header is defined according to the RTSP/TCP protocol.

0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
'$'	Channel number	Reserved = 0	

Segment length

- The 'channel number' is used as a marker to distinguish different data frames when the data stream is multiplexed.
- The 'segment length' is the length of the data frame excluding the segment header, and the theoretical maximum supported data frame size is 4G bytes.
- The client end distinguishes data frames by segmented headers and determines its reserved buffer size by the data flow description information from the device interface.

5.1. Real-time data frame

Most of the data streams available in the device are real-time data streams, each data frame contains temporal information, and real-time data frames use the SRTP packet header:

0 1 2 3 4 5 6 7				0 1 2 3 4 5 6 7		0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
V=2	P	X	CC	M	PT	Sequence number	
timestamp							

- Consistent with RTP definition (rfc3550)
- The units of the timestamp are determined by the specific data stream's properties

5.2. Video streaming services

The device uses a separate service port to provide video streaming services to clients, port number 10081, and each video stream will have corresponding video stream attributes included in the STREAM control module, for example: /video/raw. The stream attributes for a video stream can be obtained with the 'GET /stream/video/raw' command.

6. Device Access Rights

Access restrictions to device functions are divided in terms of methods into network layer restrictions (TBD) and application layer restrictions. Network layer restrictions are performed by filtering the IP address and MAC address of the party requesting access to the device (TBD).

Application layer restrictions include two parts: username control and production permission formation.

6.1. User Name Access Control

Users' permission to access the device is managed by user names. The device is shipped with an empty user list by default, and the device can be accessed anonymously. Using anonymous users instead of admin/123456 makes it easier to demonstrate and develop, and also to clearly distinguish the device access control mode.

When there is no user name list in the device: (GET /admin/users), the device is in anonymous access mode or permissionless control mode, and all interfaces are accessible. When the user needs enhanced security, adding the username list switches the device state to username permission control mode, but the user can no longer access the device anonymously.

The 'PUT /user/[username]' interface allows users to add usernames and change passwords.

6.2. Privilege groups

When the user list is created, the first user is assigned to the 'root' group by default, and the root user can be deleted only after all other users have been deleted. When the user list is restored to empty, the device enters permissionless control mode. Users in higher privilege groups have the privileges of all lower privilege groups.

The group name of the access permission group. The following table lists the permissions in descending order.

group name	description
root	This group can contain only one user, which can add and modify other users and reset passwords
manager	All operations except ones related to user list
operator	Operators, see interface definition
viewer	Observer, see interface definition

6.3. Password protection

Access to each interface of the device is authenticated using the HTTP Digest Authentication (RFC 2617) for permissions, which allows effective password protection.

When creating a user and password, the user software may encrypt the plaintext password and then write it to the device, and all subsequent login accesses shall use that encryption method to generate equivalent passwords.

4

Interface Definition

Each parameter configuration class interface in the definition uses the full configuration tree, while during the actual access, user can use a partial configuration tree.

The configuration tree that the user gets when reading the interface will have some read-only status nodes that are not included in the full configuration tree and cannot be modified.

1. Command Definition

Access to the device is done through the HTTP protocol, where each command consists of a resource path (URL) and an operation method. Operation methods include: GET, PUT, DELETE, POST

When describing a command, we keep only the path part of the URL.

```
GET /admin/info
```

When the instruction contains parameters, they are italicized to indicate that.

```
PUT /user/user_name
```

The actual command replaces the brackets with real parameters, and the parameter naming rules

```
PUT /user/somebody
GET /sensor/luts/2?list
```

should conform to III.2.8

The body format of the request and response are listed separately in the table **below**, with each available field defined in the description. For 'GET/PUT' corresponded command, the field definition is the same for both operations.

2. Device Management **ADMIN**

This module includes device management functions such as device information, device network interface, user management, access control, firmware upgrade, etc. This module is not included in the configuration set.

2.1. Get device information

GET /admin/info	Permissions**: ALL**
Requests and responses	instructions
{ "create-date" : "", "device-fw" : { "build" : "20161202", "name" : "iir-fw", "tag" : "rc7", "version" : [3, 0, 0, 101] }, "device-id" : "gl9ARQ1Fc/dEzDuOwQvDsA==", "device-model" : "Fotric123",	device-id; device-fw; (firmware informati on) version; (Firmware version) tag; (version tag) build; (generated date) name; (firmware name) device-model ; device-sn0; (device serial number) device-tag0; (whether the device is a prototype) device-tag1; device-tag2 ;sensor;

	(Sensor Type)

"device-sn0" : "00000", "device-sn1" : "", "device-tag0" : "sample", "device-version" : "1.0", "hardware" : "a", "hardware-version" : "2321", "modify-date" : "", "seal-date" : "", "sensor" : "", "sensor-sn" : "2131123", "update-counter" : 4 }	sensor-sn; (Sensor Serial Number) hardware; (Hardware Type) hardware-version; create-date; (Date of device creation) modify-date; (Date of last modification of device information) seal-date; (date of the equipment leaves factory) update-counter; (Device Information Update Count)
status code	
200	

2.2. Get device time

GET /admin/clock	Permissions**: ALL**
Requests and responses	instructions
"20161202071928"	Time is represented as a compressed numeric string
status code	
200	

2.3. Set the device time

PUT /admin/clock	Permissions**: MANAGER**
Requests and responses	instructions
"20161201071900"	The time must be expressed in 14 digits number format and cannot be set below the unit can not be smaller than 'second'
status code	
200	
400	Incorrect time format

2.4. Get a list of NTP servers

GET /admin/ntp	Permissions: **MANAGER**
Requests and responses	instructions
["pool.ntp.org", "time.nist.gov"]	
status code	
200	

2.5. Setting up the NTP server list

PUT /admin/ntp	Permissions: **MANAGER**
Requests and responses	instructions
["pool.ntp.org", "time.nist.gov"]	The list will turn off the NTP feature
status code	
200	

2.6. Obtaining device network information

GET /admin/ifaces	Permissions: **ALL**
Requests and responses	instructions
{ "eth0" : { "dhcp" : false, "dns" : ["192.168.1.1"], "gateway" : [	• ip; device IPv4 address • dhcp; Whether to use the DHCP protocol to automatically obtain network parameters • dns; domain name server address list • gateway; gateway list • netmask; subnet mask • mac; network card address

"192.168.1.1"], "ip" : "192.168.1.115", "mac" : "06:00:01:FE:4F:29", "netmask" : "255.255.255.0", "prefix" : 24 } }	
GET /admin/ifaces/eth0	Permissions**: ALL**

{
"dhcp" : false,
"dns" :
[
"192.168.1.1"
],
"gateway" :
[
"192.168.1.1"
],
"ip" : "192.168.1.115",
"ipv6" : "fe80::400:1ff:fefe:4f29/64",
"mac" : "06:00:01:FE:4F:29",
"netmask" : "255.255.255.0",
"prefix" : 24
}
	ibid

status code	
200	
404	Network device not found

2.7. Setting Device Network Information

PUT /admin/ifaces/*eth0*	Permissions**: MANAGER**
Requests and responses	instructions
{ "dhcp" : false, "dns" : ["192.168.1.1"], "gateway" : [	Listed are the network parameters that can be modified

"192.168.1.1"], "ip" : "192.168.1.115", "netmask" : "255.255.255.0" }	
	vacant
status code	
200	
400	Network parameter error

2.8. Get a list of users

GET /admin/users	Permissions**: MANAGER**
Requests and responses	instructions

{
"admin" : "root",
"user1" : "viewer"
}	

status code	
200	

2.9. Adding and modifying users

PUT */user/user_name* · user_name New user name	Permissions:**ROOT**
Requests and responses	instructions

{
"pw": "EzDuOw",
"group":
"operator"
}
	· pw password
	· group root, manager, operator, viewer

status code	
200	success
403	User already exists or password is too short

2.10. Delete user

DELETE */user/user_name*	Permissions**: ROOT**
Requests and responses	instructions
status code	
200	
403	

2.11. Get Boot ID

GET /admin/boot-id	Permissions**:ALL**
Requests and responses	instructions
"hVHruyn2TY6j7zyAfZ2hSg=="	BASE64 encoding of the UUID
status code	
200	

2.12. Upgrade firmware

POST /admin/update	Permissions**: MANAGER**
Requests and responses	instructions
{ "path" : "/file/cache/9aubeEZTjKYBj23k" }	· path The path in which the firmware upload to the device cache · Executing this command will cause the device to reboot
	(The command returns and the device begins to reboot)
status code	
200	
404	Firmware not found
POST /admin/update	Permissions**: MANAGER**
Requests and responses	instructions
{ "link": "47.88.34.128", "port":80 "path":"/upgrate/firmware/iir-fw-1.0.0.0.bin" }	· link Remote server address server port · port · path Server Path
	(this operation will start an asynchronous firmware download)

status code	
202	Start downloading firmware
503	Downloading firmware now

2.13. Device reboot

POST /admin/reboot	Permissions**: ALL**
Requests and responses	instructions
	(The command returns and the device begins to reboot)
status code	
200	

2.14. Restore factory configuration

POST /admin/reset?factory **POST /admin/reset?restore** · 'factory' does not change network parameters · 'restore' Also restores factory network parameters	Permissions**: ALL**
Requests and responses	instructions
status code	
200	

2.15. Save the current configuration set

PUT */admin/cfgsets/cfg_name* · cfg_name (Configuration set name)	Permissions**:OPERATOR**
Requests and responses	instructions
status code	
200	
403	Exceeding the maximum number of saves

2.16. Get a list of configuration sets

GET /admin/cfgsets	Permissions**: OPERATOR**
Requests and responses	instructions
["cfg1", "cfg2", "cfg3"]	
status code	
200	
403	Exceeding the maximum number of saves

2.17. Load configuration set

POST *<u>/admin/cfgsets/cfg_name</u>* · cfg_name Configuration set name	Permissions**:OPERATOR**
Requests and responses	instructions
status code	
200	
404	Configuration set does not exist

3. Sensor

The sensor module controls the operation of the thermal imaging detector, including obtaining detector information, switching the detector's temperature range, calibration method, and image mirroring. The movement uses an integer index to refer to the list items, the index value should be greater than 0 and the 0th item in the list is a null value.

3.1. Get a list of calibration modes

GET /sensor/cali-modes	Permissions**: VIEWER**
	Configuration set: No
Requests and responses	instructions

	vacant
["auto", "manual"]	· auto auto-calibration · manual manual calibration
status code	
200	
400	

3.2. Get current calibration mode

GET /sensor/cali-mode	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
"auto"	
status code	
200	
400	

3.3. Set the current calibration mode

PUT /sensor/cali-mode	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions
"manual"	Values in the calibration mode list only
status code	
200	
400	Unknown calibration mode

3.4. Perform a calibration

POST /sensor/cali	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions

status code	
200	
400	

3.5. Get sensor image mirroring mode

GET /sensor/mirror	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"h" : 0,
"v" : 0
}
	· h whether to mirror horizontally
	· v whether to mirror vertically

status code	
200	

3.6. Set the sensor image mirroring mode

PUT /sensor/mirror	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions

{
"h" : 0,
"v" : 0
}
	· h whether to mirror horizontally
	· v whether to mirror vertically

status code	
200	

3.7. Get sensor image size

GET /sensor/dimension	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions
{ "h" : 80,	Dimensions in pixels

"w" : 80 }	
status code	
200	

3.8. Get a list of temperature ranges

GET /sensor/t-range	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions

[
null,
{
"high" : 150,
"low" : -20
},
{
"high" : 300,
"low" : 0
},
{
"high" : 650,
"low" : 300
}
]
	Item 0 must be a null value

status code	
200	

3.9. Get a list of lens info

GET /sensor/lens	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions

[
null,
{
"model" : "default"
}
	Item 0 must be a null value

]	
status code	
200	

3.10. Get a list of default look up table

GET /sensor/luts	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions

[
null,
{
"lens" : 1,
"t-range" : 3
},
{
"lens" : 1,
"t-range" : 2
},
{
"lens" : 1,
"t-range" : 3
}
]	

status code	
200	
400	

GET */sensor/luts/idx* · idx list index	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions

{
"lens" : 1,
"t-range" : 3
}	

status code	
200	
404	

3.11. Select temperature comparison table

PUT /sensor/jconfig	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions

{
"selected-lens" : 1,
"selected-t-range" : 2
}
	Index items should be in the range of the corresponding list, with indexes starting at 1

status code	
200	
400	Index overstepping
404	A temperature list does not exist

3.12. Get the current factory temperature cross-reference table index

GET /sensor/lut	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions
1	
status code	
200	
404	Failed to switch the temperature gauge last time

3.13. Get factory temperature comparison table

GET */sensor/luts/idx*?list · idx is the index of the temperature comparison table	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions
[{ "r" : 7455, "t" : -24.700001 }, { "r" : 7504,	· r AD value · t Celsius temperature

"t" : -19.800001 }, { "r" : 7617, "t" : -9.8000002 }, ...	
status code	
200	
404	

4. IMAGE CAPTURE

4.1. Get the current image capture mode

GET /capture/mode	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
"continuous"	
status code	
200	

4.2. Setting the image acquisition mode

PUT /capture/mode	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
"trigger"	・trigger capture by an external signal ・continuous countinuous streaming
status code	
200	

4.3. Trigger one capture

POST /capture/trigger	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
status code	
200	

5. Image Processing (ISP)

The image processing module contains several sub-modules, mainly for visualization of raw thermal image data, analysis of temperature data, intelligent analysis, etc.

5.1. Get image processing parameters

GET /isp/jconfig	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"black-level" : 0,
"contrast" : 1,
"method" : "tpe",
"roi" : "full",
}	

status code	
200	

5.2. Set image processing parameters

PUT /isp/jconfig	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions
{ "black-level" : 0, "contrast" : 1,	• black-level black level [-30 ~ 30] • contrast (0 , 3.0) • method auto gain method: linear, hp, tpe

"method" : "tpe", "roi" : "full" }	· roi auto gain calculation area: full, middle
status code	
200	

5.3. Get Auto Gain Control Mode

GET /isp/agc-mode	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions
	vacant
"continuous"	Mode. · once performs an automatic gain · continuous continuous Auto Gain
status code	
200	

5.4. Setting the Auto Gain Control Mode

PUT /isp/agc-mode	Permissions: **OPERATOR**
	Configuration set: No
Requests and responses	instructions
"continuous"	(This item is not saved when power is down)
status code	
200	
400	

5.5. Get a snapshot of the original image

GET /isp/snapshot	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions
{	Snapshots are in P7 format, see appendix

"path" : "/file/cache/8kn4VBYmAw0XKUvl" }	
status code	
200	

5.6. Get a snapshot of the temperature map

GET /isp/t-snapshot	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions

{
"path" :
"/file/cache/8kn4VBYmAw0XKUvl"
}
	The snapshot is in P7 format, see appendix. The temperature data is expressed as a 16-bit signed integer, where the lowest 3 bits are fixed-point decimal places. For example, 0x007A for Celsius 15.25 degrees

status code	
200	

5.7. Get the temperature of the point on the image

GET /isp/t **GET /isp/t?*x=xpos&y=ypos*** · x,y image coordinates · Default to the center point when there is no parameter	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions

{
"r":7812,
"t":23.234
}
	Use of global temperature compensation parameters

status code	
200	
404	The object does not exist

6. Temperature measurement **ISP/INSTRUMENT**

The temperature measurement module adds different shapes of temperature measurement object to the image according to user settings, calculates the image temperature, and generates temperature measurement events according to triggering conditions.

The temperature measurement tool supports the addition of temperature points, temperature areas, and temperature lines to the image. It can be used to capture the temperature value of the temperature point, the maximum and minimum temperature in the temperature area, and the temperature line.

- The temperature measurement area can be defined by polygons and area masks.
- The number of vertices in polygons or polylines cannot exceed 50.
- The name of the temperature measurement object must not exceed 40 characters in length.

6.1. Obtain global temperature measurement parameters

The global temperature parameters are the basic parameters of the device temperature measurement. The temperature measurement object inherits all global temperature parameters by default, and the temperature measurement object can also set its local temperature parameters.

GET /isp/instrument/jconfig	Permissions: **VIEWER** Configuration set: Yes
Requests and responses	instructions
	vacant

{
"ambient-t" : 20,
"distance" : 1,
"emissivity" : 0.97,
"lens-t" : 20,
"lens-transmissivity" : 1,
"offset" : 0,
"reflect-t" : 20,
"rh" : 0.5
}
	・ambient-t; ambient temperature (> -273.15)
	・distance; to the object (greater than 0)
	・emissivity Emissivity of the measured object (0, 1]
	・lens-t lens temperature (> -273.15)
	・lens-transmissivity lens transmittance (0, 1]
	・offset offset (correction for systematic errors)
	・reflect-t reflection temperature (> -273.15)

status code	
200	

6.2. Set global temperature measurement parameters

PUT /isp/instrument/jconfig	Permissions: **OPERATOR** Configuration set: Yes
Requests and responses	instructions

{ "ambient-t" : 20, "distance" : 1, "emissivity" : 0.97, "lens-t" : 20, "lens-transmissivity" : 1, "offset" : 0, "reflect-t" : 20, "rh" : 0.5	

}	
	vacant
status code	
200	

6.3. Get a list of temperature measurement objects

GET /isp/instrument/objects/points **GET /isp/instrument/objects/regions** **GET /isp/instrument/objects/lines**	Permissions: **VIEWER** Configuration set: Yes
Requests and responses	instructions
["p1", "p2"]	
status code	
200	

6.4. Obtaining temperature measurement object parameters

GET */isp/instrument/objects/points/obj_name* **GET */isp/instrument/objects/regions/obj_name*** **GET */isp/instrument/objects/lines/obj_name*** **GET /isp/instrument/objects/global** ・ obj_name Object name	Permissions: **VIEWER** Configuration set: Yes
Requests and responses	instructions
	vacant
{ "emissivity" : 0.97, "pos" : { "x" : 55, "y" : 50 }, "distance" : 1000, "polygon" : [	Temperature measurement parameters can be modified individually for each object. ・ emissivity ・ reflect-t ・ distance ・ offset ・ osd-visible visible on OSD ・ api-visible whether or not transsfer this with tagging stream ・ mb-slot allocated to Modbus address space ・ lable content of the label used for display

{
"x" : 30,
"y" : 50
},
{
"x" : 35,
"y" : 10
},
{
"x" : 60,
"y" : 60
},
{
"x" : 5,
"y" : 76
}
],
"label" : "My Region1"
}
	· alarm-t temperature alarm threshold
	
	Regions
	· layer regional hierarchy
	· polygon polygon vertices list
	· mask mask image cache token
	
	lines
	· polyline polyline vertices list

status code	
200	
404	Object does not exist

6.5. Create and modify temperature measurement objects

PUT*/isp/instrument/objects/points/obj_name* **PUT*/isp/instrument/objects/regions/obj_name*** **PUT */isp/instrument/objects/lines/obj_name*** **PUT /isp/instrument/objects/global** · obj_name Object name	Permissions: **OPERATOR** Configuration set: Yes
Requests and responses	instructions
{ "emissivity" : 0.97, "pos" : { "x" : 55, "y" : 50 }, "mb-slot" : 1	

}	
status code	
200	
201	new object

6.6. Delete temperature measurement objects of same type

DELETE /isp/instrument/objects/points **DELETE /isp/instrument/objects/regions** **DELETE /isp/instrument/objects/lines**	Permissions: **OPERATOR** Configuration set: Yes

Requests and responses	instructions
status code	
200	
404	The object does not exist

6.7. Delete specific temperature measurement object

DELETE ***/isp/instrument/objects/points/obj_name*** **DELETE*****/isp/instrument/objects/regions/obj_name*** **DELETE*****/isp/instrument/objects/lines/obj_name*** ・obj_name Object name	Permissions: **OPERATOR** Configuration set: Yes

Requests and responses	instructions
status code	
200	
404	The object does not exist

6.8. Get temperature measurement object data

GET*/isp/instrument/objects/points/obj_name*?value **GET*/isp/instrument/objects/regions/obj_name*?value** **GET*/isp/instrument/objects/lines/obj_name*?value** · obj_name Object name	Permissions: **VIEWER** Configuration set: No

Requests and responses	instructions

{
"r":7812,
"t":23.234
}
	· r Sampling value
	· t temperature

status code	
200	
404	The object does not exist

6.9. Get the maximum number of temperature measurement objects

GET /isp/instrument/max-point-num **GET /isp/instrument/max-region-num** **GET /isp/instrument/max-line-num**	Permissions: **VIEWER** Configuration set: No

Requests and responses	instructions
10	
status code	
200	

6.10. Get the global maximum and minimum temperatures

GET /isp/instrument/objects/global?value	Permissions: **VIEWER** Configuration set: No

Requests and responses	instructions
{ "max" :	

```
    {
        "r" : 7848,
        "t" : 0.6256697,
        "x" : 72,
        "y" : 18
    },
    "min" :
    {
        "r" : 8125,
        "t" : 25.35555,
        "x" : 72,
        "y" : 18
    }
}
```


status code	
200	

6.11. Get the sampling rate

GET /isp/instrument/sample-rate	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
2	
status code	
200	

6.12. Set the sampling rate

PUT /isp/instrument/sample-rate	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions
2	
status code	
200	

7. Palette ISP/T-RAY

7.1. Get a list of device predefined palettes

GET /isp/t-ray/presets	Permissions: **OPERATOR**
	Configuration set: No
Requests and responses	instructions
["grey", "iron", "rain", "rainbow", "greyred", "glowbow", "yellow", "midgrey", "midgreen"]	
status code	
200	

7.2. Set the current palette

PUT /isp/t-ray/plt	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions

{
"inverse" : false,
"name" : "grey"
}	

status code	
200	
404	

7.3. Get the current palette

GET /isp/t-ray/plt	Permissions: **VIEWER**
	Configuration set: Yes

Requests and responses	instructions
	vacant

{
"inverse" : false,
"name" : "grey"
}
	· inverse Whether or not invert the palette
	· name Palette name

status code	
200	
404	

7.4. Setting up a custom palette

Please contact the supplier for more details

8. Autofocus ISP/AF

8.1. Perform autofocus

The focus window is in the center of the screen, so place the scene with the feature in the center of the camera

POST /isp/af	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions

{
"x" : 10,
"y" : 10
}
	Center of focus position
	Default to screen center when there is no request

status code	
200	

8.2. Get autofocus results

The focus window is in the center of the screen, so place the scene with the feature in the center of the camera

GET /isp/af/result	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions
"pending"	Results.

	· Pending Focusing · good Normal · nofeat Didn't find feature to focus on · timeout Focus timeout · unknown Unknown error
status code	
200	

9. Screen Display OSD

9.1. Get the OSD list

GET /osd/layout	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
["time", "title"]	
status code	
200	

9.2. Get the OSD object parameters

GET */osd/layout/obj_name* · obj_name Object name	Permissions: **VIEWER** Configuration set: Yes
Requests and responses	instructions

{
"show" : true,
"pos" :{
"x": 5,
"y": 2
},
"align" : "topright"
}	

status code	

200	
404	

9.3. Set the time display

PUT /osd/layout/time	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"show" : true,
"pos" :{
"x": 5,
"y": 2
},
"align" : "topright"
}	

status code	
200	
404	

9.4. Get the currently displaying time zone

GET /osd/tz	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
"CST-8"	
status code	
200	

9.1. Set the currently displaying time zone

PUT /osd/tz	Permissions: **OPERATOR**
	Configuration set: Yes
Requests and responses	instructions
"CST-8"	

status code	
200	

9.2. Set title display

PUT /osd/layout/title	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"show" : true,
"pos" :{
"x": 5,
"y": 2
},
"align" : "topright"
}	

status code	
200	
404	

9.3. Get a snapshot of the video

GET /osd/snapshot	Permissions: **VIEWER**
	Configuration set: No
Requests and responses	instructions

{
"path" :
"/file/cache/8kn4VBYmAw0XKUvl"
}
	Snapshots are in P7 format, see appendix

status code	
200	

10. Data Stream

The Data Flow module provides data flow services using port 10081, with each data flow corresponding to a data flow attribute in the Data Flow module.

10.1. Get the main-code stream properties

GET /stream/video/pri	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"bitrate" : 400000,
"height" : 256,
 	
"max-bitrate" : 400000,
"max-packet-size" : 65536,
"pixel-format" : "yuv420",
"width" : 256,
"max-connection-num" : 0,
"connections": 0
}
	· max-bitrate Maximum allowable bit rate
	· bitrate Current video stream bitrate
	· connections Current connection number downstream

status code	
200	

10.2. Get sub-code stream properties

GET /stream/video/sub	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"bitrate" : 80000,
"height" : 128,
"max-bitrate" : 80000,
"max-packet-size" : 16384,
"pixel-format" : "yuv420",
"width" : 128,
"max-connection-num" : 0,
"connections": 0
}	

status code	
200	

10.3. Get raw stream properties

GET /stream/video/raw	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"fps" : 16.129032,
"height" : 80,
"max-packet-size" : 12800,
"width" : 80
"max-fps" : 0
"max-connection-num" : 0
"pixel-format" : "grey16le",
"connections": 0
}	

status code	
200	

10.4. Set raw stream properties

PUT /stream/video/raw	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"fps" : 5,
}	

status code	
200	

10.5. Get event stream properties

GET /stream/tag/events	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions
{ "heartbeat" : 5,	

"max-packet-size" : 8192 }	
status code	
200	

10.6. Get numeric stream properties

GET /stream/tag/values	Permissions: **VIEWER**
	Configuration set: Yes
Requests and responses	instructions

{
"max-packet-size" : 8192
}	

status code	
200	

11. Peripheral Control(PERI)

11.1. Perform manual focus

POST /peri/focus?*op=op&step=step*	Permissions: **OPERATOR**
· op near/far · step step length 10-1000	Configuration set: No
Requests and responses	instructions
	A step is one-thousandth of the motor's full travel, so 1000 means finished going from closest to farthest.
	vacant
status code	
200	

11.2. Get serial port parameters

GET /peri/serial-port	Permissions: **MANAGER**
	Configuration set: No
Requests and responses	instructions

{
"baudrate" : 9600,
"command-gap" : 0,
"csize" : 8,
"modbus-address" : 1,
"modbus-slave" : false,
"parity" : "none",
"stopbit" : "one"
}
	· baudrate Serial baud-rate
	· csize bytesize
	· parity check bits
	· stopbit stop bit
	· modbus-address modbus address
	· modbus-slave Whether Modbus is enabled
	· command-gap Minimum wait time between consecutive commands

status code	
200	

11.3. Get the serial port baud-rate table

GET /peri/serial-port/baudrates	Permissions: **MANAGER**
	Configuration set: No
Requests and responses	instructions
[150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200]	
status code	
200	

11.4. Get the stop bit table

GET /peri/serial-port/stopbits	Permissions: **MANAGER**
	Configuration set.

Requests and responses	instructions
["one", "onepointfive", "two"]	
status code	
200	

11.5. Get the support parity bit table

GET /peri/serial-port/parities	Permissions: **MANAGER**
	Configuration set.
Requests and responses	instructions
["none", "odd", "even"]	
status code	
200	

11.6. Set the serial port parameters

PUT /peri/serial-port	Permissions.
	Configuration set.
Requests and responses	instructions

{
"baudrate" : 9600,
"command-gap" : 0,
"csize" : 8,
"modbus-address" : 1,
"modbus-slave" : false,
"parity" : "none",
"stopbit" : "one"
}	

status code	

200	

11.7. Control movement of the camera

POST /peri/ptz/move?id=ptz_id&vx=vx& *vy=vy*	Permissions: **OPERATOR**
· ptz_id camera serial port address (Pelco-d) · vx Horizontal velocity · vy Vertical velocity	Configuration set: No

Requests and responses	instructions
status code	
200	

11.8. Set the turret camera's preset position

POST /peri/ptz/set?*id=ptz_id&data=val*	Permissions: **OPERATOR**
· ptz_id Camera serial port address (Pelco-d) · val Preset bit number (0~255)	Configuration set: No

Requests and responses	instructions
status code	
200	

11.9. Change the camera's location according to the preset position

POST /peri/ptz/goto?*id=ptz_id&data=val*	Permissions: **OPERATOR**
· ptz_id Camera's serial port address (Pelco-d) · val Preset bit number (0~255)	Configuration set: No

Requests and responses	instructions
status code	
200	

11.10. Clear camera turret's preset position

POST /peri/ptz/clear?*id=ptz_id&data=val*	Permissions: **OPERATOR**
· ptz_id Camera serial port address (Pelco-d) · val Preset bit number (0~255)	Configuration set: No

Requests and responses	instructions
status code	
200	

11.11. Stop the camera turret

POST /peri/ptz/stop?*id=ptz_id*	Permissions: **OPERATOR**
· ptz_id Camera serial port address (Pelco-d)	Configuration set: No

Requests and responses	instructions
status code	
200	

11.12. Set MODBUS Parameters

PUT /peri/modbus	Permissions: **MANAGER**
	Configuration set: No

Requests and responses	instructions

{
"command-gap" : 0,
"enabled" : true,
"modbus-address" : 1
}	

status code	
200	

11.13. Get MODBUS parameters

GET /peri/modbus	Permissions: **MANAGER**

	Configuration set: No
Requests and responses	instructions

{
"command-gap" : 0,
"enabled" : true,
"modbus-address" : 1
}	

status code	
200	

12. General File Caching Service

12.1. Download file

GET */file/cache/file_name* · file_name File name	Permissions: **VIEWER** Configuration set: No
Requests and responses	instructions
(binary)	(Contents of document)
status code	
200	
404	

12.2. Upload files

After uploading a file the user is given a file ID that can be used as a link in other interfaces that support path fields. The size of the uploaded file must not be larger than 20MB.

POST /file/cache	Permissions: **MANAGER** Configuration set: No
Requests and responses	instructions

{
"path" : "8kn4VBYmAw0XKUvl"
}
	File's Relative Path

status code	
200	

400	

12.3. Get the log file

GET */file/log/log_name* ・ log_name Log file		Permissions: **MANAGER** Configuration set: No
Requests and responses	instructions	
(binary)	File Contents Logs include: messages daemon.log auth.log ...	
status code		
200		
404		

5

Interface Panel

1. Reset key **(RST)**

The reset key is used to restore the device's parameters to the factory state.

After the user has powered up the device, confirm that it has been initialized or wait at least 30 seconds before pressing the reset button for at least 10 seconds.

seconds, the device performs a reset operation and reboots, and the original IP address of the device is restored to its factory value.

The device's user list will not be cleared, so if you forget a user or password, contact your local provider.

6

MODBUS Interface Support

MODBUS protocol is a general protocol applied to electronic controllers, which communicate between devices over physical networks (e.g. serial networks, Ethernet) and is a widely adopted industry standard. On a MODBUS network, each device is assigned a communication address. The protocol defines how the controller accesses other devices with different addresses and how to perform various operations.

(See

http://modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf)

This protocol supports traditional RS-232, RS-422, RS-485, and Ethernet devices. Many

Industrial devices, including PLCs, DCSs, smart meters, etc., are using the Modbus protocol as the communication standard between them.

In addition to the HTTP interface, the in-line camera provides a Modbus interface for data acquisition and control using the RS-485 interface to facilitate communication with other industrial equipment.

The serial-based MODBUS protocol is generally available in RTU and ASCII formats, and this device only supports the RTU format, which has a much smaller data volume.

(11.6) Make changes.

1. Serial **RTU** Protocol

1.1. Frame Format

starting bit	address	command	data	**CRC**	End bit
3.5 words*	8 bits	8 bits	N x 8bits	16 bits	3.5 words*

* Serial communication word length

1.2. CRC check up

See appendix

1.3. Read Holding Registers

command code	**0x03**
(numeric, data) field	Example value (hex) Read address at **0x102C ~ 0x1030** The **5** holding registers
request frame	
address	01
command	03
Start address (high bit)	10
Start address (low bit)	2C
Number of registers (high)	00
Number of registers (low)	04
CRC (low)	40
CRC (high)	C0
response frame	
address	01
command	03
byte count	0A
Register 0x102C (high bit)	00
Register 0x102C (low bit)	02
Register 0x102D (high bit)	00
...	...
Register 0x1030 (low bit)	1A
CRC (low)	-

CRC (high)	-

1.4. Exception return

When an exception is returned, the highest bit of the 'command' field of the response frame is 1, which corresponds to the request command plus 0x80, and the data field returns the corresponding exception code

command code	**0x03**
(numeric, data) field	Example value (hex) Read address at **0x102C ~ 0x1030** The **5** holding registers
request frame	
address	01
command	03
Start address (high)	10
Start address (low bit)	2C
Number of registers (high)	00
Number of registers (low)	04
CRC (low)	40
CRC (high)	C0
response frame	
address	01
command	83
Exception Code	0A
CRC (low)	-
CRC (high)	-
(numeric, data) field	Example value (hex)
address	01
command	83

2. Address Space Definition

The device address is defined on a single contiguous space, use 16bit words as units, starting from 0, with all integers placed big-end first

Address range (hexadecimal)	Length (words)	elements
system information area	**512**	
0000	1	Fixed value 0x4952
0001	1	Protocol version (==1)

0002 ~ 0003	2	Firmware version (0x03010011 indicates 3.1.0.17)
0004	1	Number of measurement blocks supported
0008 ~ 0029	40	model number
0030 ~ 0058	40	product key (software)
Measurement data area	**512**	
0200 ~ 0207	8	Data block 1
0208 ~ 0210	8	Data block 2
...		
0200 + 8 x n ~ 0200 + 8 x (n+1) - 1	8	Data block n
Control area (**TBD**)		

3. Measurement data block

The temperature measurement data is mapped to the data block via the mb-slot field (0) and the measured value is read out via the corresponding data block when the measurement object is associated with the data block. The size of each data block is 8 words. 2 words of a 32-bit signed integer (big-endian) are used as Q15.16 fixed-point integers, the first word being a 16-bit integer part and the second word being a 16-bit fixed-point decimal

See https://en.wikipedia.org/wiki/Q_(number_format)

Take data block **n as** base address: **0x200 + n * 8**.

Address range (hexadecimal)	Length (words)	elements
0000~0001	2	Temperature, maximum temperature
0002~0003	2	minimum temperature

7

Onvif interface and **RTSP** video streaming

Some device models support the Onvif 2.0 interface for accessing mainstream security surveillance systems. RTSP video streams, which are part of the Onvif protocol, can also be used separately. The following is a list of specific interfaces supported by the device.

1. Basic functions **(Core Spec. Ver 2.4.2)**

functional name	description	note
Device Discovery	Device Discovery	When a device go online, there will be a prompt 'Hello'; When a device go offline, there will be no prompt
GetCapabilities	Get the device menu	
SystemReboot	Rebooting the device	

2. Event handling **(Core Spec. Ver 2.4.2)**

functional name	description	note
GetEventProperties	Get event properties	
Subscribe	Event Subscription	
Renew	Event Subscription Updates	
Unsubscribe	Unsubscribe	
Notify	Message Trigger	Sent by the device to the subscriber

3. Streaming media **(Media Service Ver 2.4.2)**

functional name	description	note
GeVideoSources	Get a list of video sources	
GetProfile	Get video configuration	
GetProfiles	Get a list of video configurations	
GetVideoSourceConfigurations	Get video source configuration	
GetVideoEncoderConfiguration	Get the video encoding configuration	
GetVideoEncoderConfigurations	Get a list of video encoding configurations	
GetVideoEncoderConfigurationOptions	Get a list of optional configurations for video encoding	
GetStreamUri	Get the video stream URI	
GetOSDOptions	Get OSD options	
GetOSD	Get OSD	OSD title string
SetOSD	Setting the OSD	
CreateOSD	Creating an OSD	

DeleteOSD	Delete the OSD	

3.1. RTSP Video Streaming

The device will also support video output via RTSP if it supports the Onvif protocol. The device supports RTSP/RTP/TCP

The access path refers to HTTP video streaming IV.10, specifically: Main stream.

```
RTSP://xxx.xxx.xxx.xxx/video/pri
```

Subcode streams.

```
RTSP://xxx.XXX.xxx.xxx/video/sub
```

4. Cloud Station Control **(PTZ spec. Ver 2.4.2)**

functional name	description	note
GetConfigurations	Get all head configuration information	
GetConfiguration	Get Head	
ContinuousMove	Moving Head, Focusing	Currently, only focus is supported
Stop	Stopping the head, focusing	
GetPresets	Get a list of preset bits	256 preset bits
SetPreset	Setting the preset bit	
GotoPreset	Go to preset position	
RemovePreset	Deleting preset bits	

8

Appendix

1. Data Type Definition

types	Length (bytes)	note
UUID	16	
MD5	16	
String	80	utf-8
Integer	4	
Integer (unsigned) Unsigned Integer	4	
Floating Point	4	32-bit IEEE 754 floating-point number
Enumeration	4	Use of integers (unsigned)
Date	16	

2. Common **HTTP** Return Status Codes

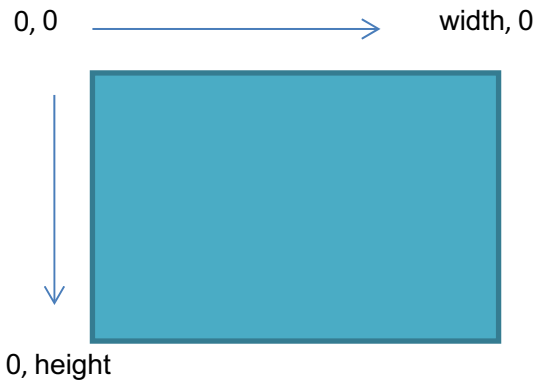
status code	types	instructions
200	success	OK.
201	success	Created
301	redirects	Moved Permanently
302	redirects	Found
400	Client Error	Bad Request
401	Client Error	Unauthorized
403	Client Error	Forbidden
404	Client Error	Not Found
500	Server Error	Internal Server Error
501	Server Error	Not Implemented
502	Server Error	Bad Gateway

3. Snapshot Image Format

Image header (16 bytes).

<table>
<tr><td>0 1 2 3 4 5 6 7</td><td>0 1 2 3 4 5 6 7</td><td>0 1 2 3 4 5 6 7</td><td>0 1 2 3 4 5 6 7</td></tr>
<tr><td>'P'</td><td>'7'</td><td colspan="2">Image width width (16bit)</td></tr>
<tr><td colspan="2">Image heightheight (16bit)</td><td>Bit Depth</td><td>C</td><td>E</td><td>Image type type</td></tr>
<tr><td colspan="6">Line size
linesize(bytes)</td></tr>
<tr><td colspan="6">retain</td></tr>
</table>

Image data, stored row-first, each row defined by the linesize



Number of bytes per line of image, image offset per line calculated by.

offset of i_{th} line = ptr + $linesize * i$

- Bit depth: number of bytes per pixel (not bits)
- Image type: 0 means original image, one image plane; 1 means yuv420p mode
- C: Image block identification
- E: Extension header

Extended header (64 bytes)

0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7	0 1 2 3 4 5 6 7
E	Type=1	Retai n	
Image data offset Data offset			
Fixed chunk list size			
Retain (52 bytes)			

4. MODBUS Exception Codes Table

exception code	name (of a thing)	instructions
01	Illegal orders	The device cannot recognize the command code in the request frame
02	illegal address	The device cannot access the requested address
03	Data error	Incorrect data value in the request frame
04	Internal errors	An error occurred while processing a command within the device
05	handshake	
06	Device is busy	

5. MODBUS CRC Generation Code

From Modicon Modbus Protocol Reference Guide (PI-MBUS-300)

```
/*
    MODBUS CRC Generation fucntion
    Input Paramers:
      puchMsg: message to calculate CRC upon
      usDataLen: quantity of bytes in message
*/
unsigned short CRC16(unsigned char *puchMsg, unsigned short
usDataLen)
{
     unsigned char uchCRCHi = 0xFF ; /* high byte of CRC initialized
     */  unsigned  char  uchCRCLo  =  0xFF  ;  /*  low  byte  of  CRC
     initialized */ unsigned uIndex ; /* will index into CRC lookup
     table */
     while (usDataLen--) /* pass through message buffer */
     {
        uIndex = uchCRCHi ^ *puchMsgg++ ; /* calculate the CRC */
        uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex} ;
        uchCRCLo = auchCRCLo[uIndex] ;
     }
```

```
/* Table of CRC values for high-order byte */
static unsigned char auchCRCHi[] = {
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80,
0x41,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1,
0x81,
0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00,
0xC1,
0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80,
0x41, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0,
0x80, 0x41, 0x01,
```

```
0x40, 0x01, 0xC0,
0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1,
0x81, 0x40, 0x01,
0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81,
0x40, 0x01, 0xC0,
0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
0x80, 0x41, 0x01,
0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01,
0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
0x00, 0xC1, 0x81,
0x40
} ;

/* Table of CRC values for low-order byte */
static char auchCRCLo[] = {
0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06, 0x07, 0xC7,
0x05, 0xC5, 0xC4,
0x04, 0xCC, 0x0C, 0x0D, 0xCD, 0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB,
0x0B, 0xC9, 0x09,
0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A, 0x1E, 0xDE,
0xDF, 0x1F, 0xDD,
0x1D, 0x1C, 0xDC, 0x14, 0xD4, 0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2,
0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3, 0xF2, 0x32,
0x36, 0xF6, 0xF7,
0x37, 0xF5, 0x35, 0x34, 0xF4, 0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E,
0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29, 0xEB, 0x2B,
0x2A, 0xEA, 0xEE,
0x2E, 0x2F, 0xEF, 0x2D, 0xED, 0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27,
0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60, 0x61, 0xA1,
0x63, 0xA3, 0xA2,
0x62, 0x66, 0xA6, 0xA7, 0x67, 0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD,
0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68, 0x78, 0xB8,
0xB9, 0x79, 0xBB,
0x7B, 0x7A, 0xBA, 0xBE, 0x7E, 0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4,
0x74, 0x75, 0xB5,
```

```
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71, 0x70,
0xB0,

0x50, 0x90, 0x91,

0x51, 0x93, 0x53, 0x52, 0x92, 0x96, 0x56, 0x57, 0x97, 0x55, 0x95,
0x94,

0x54, 0x9C, 0x5C,

0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B, 0x99,
0x59,

0x58, 0x98, 0x88,

0x48, 0x49, 0x89, 0x4B, 0x8B, 0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F,
0x8D,

0x4D, 0x4C, 0x8C,

0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42, 0x43,
0x83,
```