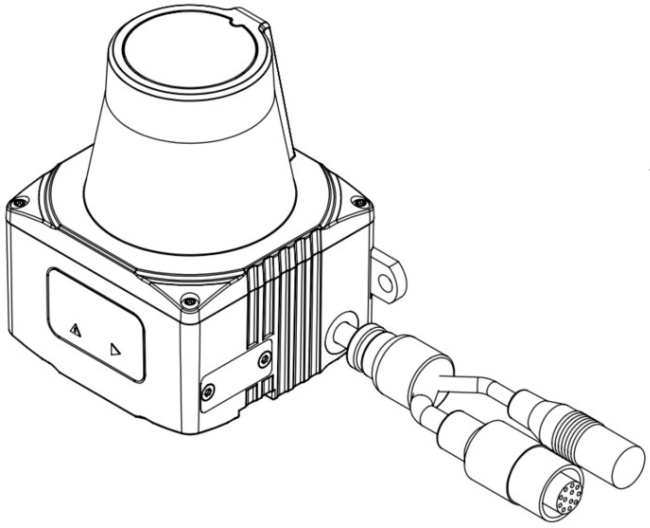
LiDAR User Manual

January 2023



Contents

Preface

This user manual contains the introduction, use and maintenance of the LiDAR. Please read this manual carefully before formal use, and strictly follow the steps described in the manual during use to avoid product damage, property loss, personal injury or/and violation of product warranty terms.
If you encounter problems that cannot be solved during use, please contact sentiacu staff for assistance.

Contact Details

Official website: www.sentiacu.com
Contact number: 400-880-9610
For technical questions, please contact: support@sentiacu.com

Copyright Notice

DisclaimerThe LiDAR product is constantly being improved, and its specifications and parameters will undergo iterative changes. Please refer to the official website for latest version.

Version history

Version 0.1

Release Date: 2021-03-18

Version Log:Initial release.

Version 1.0

Release Date: 2021-10-11

Version Log:Update the LiDAR parameters.

Version 1.1

Release Date:2021-11-09

Version Log:Add Data Block analysis instructions and examples.

Version 1.2

Release Date:2021-11-15

Version Log:Add instructions of Coordinate System of lidar.

Version 1.3

Release Date:2021-12-14

Version Log:Modify the electrical interface instructions and dimension drawing.

Version 1.4

Release Date:2022-01-07

Version Log:Add the RESTful API instruction and sensor maintenance instruction.

Version 1.5

Release Date:2022-04-06

Version Log:Add lidar filtering algorithm instrution and settings.

Version 1.6

Release Date:2022-04-24

Version Log:Add the lidar light spot size instruction, update the markings of lidar's optical cover dimension drawing.

Version 1.8.1

Release Date:2022-09-15

1 Safety Notices

Make sure to read through this user manual before your first use.LiDAR shall not be held responsible for any direct or indirect loss caused by abnormal working or damage of the products due to users' failure to store and use our products in strict accordance with the requirements of the latest product manual or to accept guidance or reference from a third party.

- **Retain Manual**
  Please keep the user manual properly,and it must be handed over together with product.

- **Follow Instructions**
  Please carefully check and follow the working rating of this product before use, operation beyond the rated range will cause permanent damage to this product.

- **Flammable and Explosive**
  Do not use the measuring instrument in a potentially explosive environment with flammable liquid, gas or dust. The product may generate sparks which ignite dust and gas.

- **Prohibition of Disassembly**
  In order to ensure user safety and avoid equipment damage，
  do not modify or disassemble the product without permission.

- **Do not View Directly**
  When the device is running, infrared laser is emitted continuously.
  To ensure safety, do not view the optical window for a long time.

Special Warnings

International Electrotechnical Commission (IEC) and Food and Drug Administration (FDA) have classified laser equipment according to the magnitude of the laser output value:

The IEC standard classifies laser equipment into four classes, called Class1, Class2, Class3, Class4.Among which, they can be subdivided into the following classes:

Class 1 laser equipment, for example, is a safety device under "foreseeable operating conditions"; The use of Class 4 laser equipment, however, is likely to generate harmful diffuse reflectance, which can cause skin burns and even fire, and should be used with special care.

The FDA standard classifies laser equipment into five classes, from I to V.

I laser products have no biological hazards. Any beam that may be viewed is shielded, and the laser system is interlocked during laser exposure.

II laser products have an output power of 1 mW. It won't burn your skin and won't cause a fire. Because eye reflexes can prevent some eye damage, such lasers are not considered dangerous optical devices.

IIIa laser products have output power from 1 mW to 5 mW. It won't burn the skin. Under certain conditions, such lasers can cause blindness and other damage to the eye.

IIIb laser products have output power ranging from 5 mW to 500 mW. At high power levels, these laser products can scorch skin. This kind of laser product is clearly defined as harmful to eyes, especially when the power is relatively high, it will cause eye damage.

V laser products have an output power greater than 500 mW. Such laser products can certainly cause eye damage. Just as it burns skin and sets clothing on fire, it can also ignite other materials.

This product belongs to Class 1 laser equipment and has no biological hazard.

Laser Safety

2 Product Overview

LiDAR product is a miniaturized high-performance lidar based on the principle of direct Time of Flight (dToF). This product is mainly used in robot-environment perception, UAV mapping, industrial process monitoring, security and other fields.

The product uses a laser diode with a high speed continuously rotating mirror to realize the scanning measurement of the environmental profile within 270 degrees of the horizontal plane. The wavelength of laser is 940nm which is invisible and laser satisfies the Class 1 standard of IEC60825-1 and ensure the safety of the eyes when the product is working. High quality motor and light weight scanning mirror ensures the product can work steadily for a long time and create maximum value for users.

The product provides Ethernet connection based on 10Base-T/100Base-TX, the DC power supply supports an ultra-wide voltage range of 9-36V, and provides a USB2.0 Type-C interface (support USB Type-A to USB Type-C and USB Type-C to USB Type-C cable connection),which can directly provide power supply and establish data connection for the product.The USB Type-C interface of the product strictly follows the operating system standards and supports RNDIS (Windows/Linux), CDC-ECM (Mac OSX), which can be plug-and-play without driver.

The built-in web server of the product provides user-friendly dashboard functions. Users can easily access the product dashboard through a web browser with mobile phones or computers. User can check the operating status of the product, set working parameters, configure network information and other functions through the Web Server, and can easily update the device firmware for more functions.

3 Operating Principle

LiDAR product is one kind of lidar using pulsed Time-of-Flight (pToF) methodology, which consists of laser diode, optical detector, optical system and Time-to-Digital converter. The core ranging principle of product: laser diode emits a beam of by short laser pulses, meanwhile the Time-to-Digital converter starts, object's surface produces a diffuse reflection after the beam hitting the target object and the returning beam is detected by the optical detector, and then Time-to-Digital converter stops. Then we can get the time between laser emission and receipt, which is the travel time of the laser beam. The travel distance of the laser beam can be calculated by multiplying the speed of light and travel time of the laser beam, accordingly distance of sensor to target object can be calculated,as is shown in figure 3.1.

travel distance of the laser beam: $$d = c \times \Delta t \tag{3.1}$$

distance of sensor to target object: $$d = \frac{c \times \Delta t}{2} \tag{3.2}$$

The method above only achieves point-to-point ranging, in order to realize point-to-plane 2-dimensional Scanning, LiDAR product has a built-in rotating mirror to realize ranging of different azimuth with rotating mirror by the motor driving. The ranging results obtained by rotation of the mirror are combined in sequence, which are the set of all the ranging results in a plane.

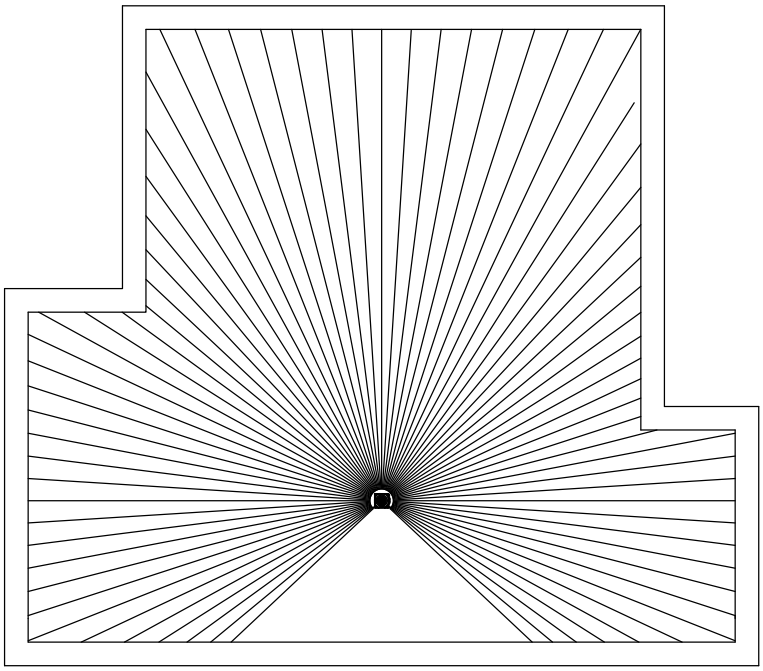


Figure 3.1: 2-Dimensional Scanning Diagram

LiDAR product outputs azimuth,distance of strongest return,RSSI of strongest return in every single mearurement.

4 Electrical Interface

LiDAR product has a built-in Type-C USB interface,which supports power supply (5V only) and data transmission through the USB interface.

Please check and follow the working rating of product before use,operation out of the rated range will cause permanent damage to product.

4.1 Mechanical Interface

The specific connected definition of LiDAR product cable interface is as follows:

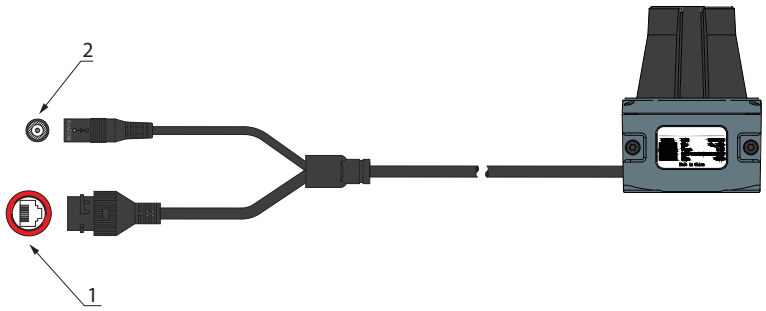


Figure 4.1: Lidar Cable Diagram

Port	Interface	Function
1	RJ45 Socket	Standard Ethernet Port
2	DC5.5-2.1 Socket	Power Supply

Table 4.1: Lidar Cable Interface Function

5 System Confirguration

In addition to USB connection, LiDAR product also provides a cable with RJ45 socket and DC 5.5-2.1 socket, users can use the cable to establish power and data connection with the product. Connect the RJ45 socket to the network through the Ethernet cable. To ensure connection stability, select Ethernet cables of CAT.5E or higher specifications. Connect the power supply to DC 5.5-2.1 socket.

5.1 Establish Connection

LiDAR product has a built-in web server. When using this product for the first time, connect the network cable and power with the computer, the default network configuration of the product is static mode, and the factory settings of the product IP address is 192.168.198.2 (Device). Users need to manually change the IP address of the Ethernet port connecting the lidar to the computer to 192.168.198.1 (Host). Then users can access the web browser http://192.168.198.2 to the built-in web service of the product.

The product can also be connected to the computer through USB Type-C cable. It takes approximately 30 seconds to boot up and self-check after properly establishing the connection. After self-check, the computer will recognize a USB mass storage device and an RNDIS network equipment (Windows/Linux).

Users need to create an additional file named osx.apple in the root directory of the identified USB mass storage device if using macOS. After the product is safely ejected and restarted with power off, macOS will recognize a CDC-ECM device.

Both the RNDIS device and the CDC-ECM device finally provide a virtual Ethernet adapter on the client through USB. The product internally assigns a fixed IP address of 192.168.8.1 (Host) to the virtual Ethernet adapter through the built-in DHCP server. The IP address of the product is fixed at 192.168.8.2 (Device).Then users can access the built-in web service of the product by accessing http://192.168.8.2.

The built-in web service provides dashboard, lidar configuration, firmware update, and other functions.

5.2 Dashboard

The Dashboard page provides system status monitoring(including system load_average,memory usage and system uptime), lidar status monitoring(including input voltage, input current, system voltage, core temperature, APD bias voltage, motor speed, laser ranging state), Ethernet adapter information, device information.

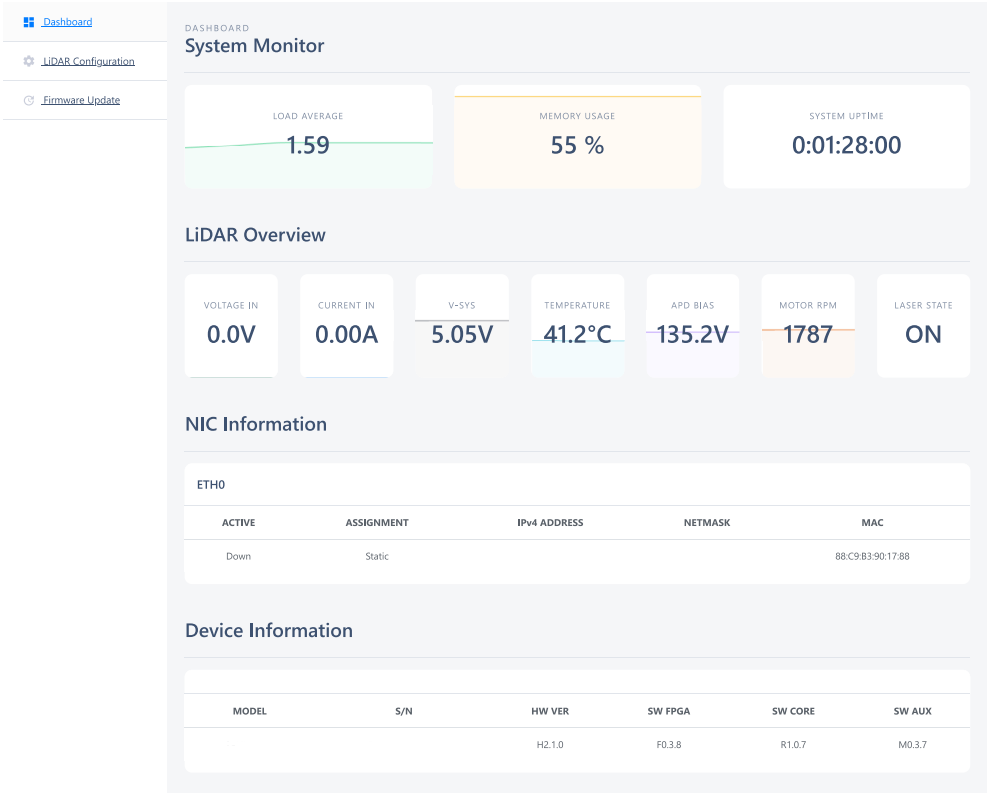


Figure 5.1: Dashboard Page

5.3 Lidar Configuration

LiDAR Configuration page provides the configuration related to Scanning (including scanning frequency settings, settings of ranging start and stop); Data Processing Configuration (including output data range settings and general filtering settings); Data sending target configuration (including IP address and data port of host); Network Configuration (including mode switching between DHCP and static IP address, static IP address settings). The new network configuration information will be automatically applied after 5-10 seconds.

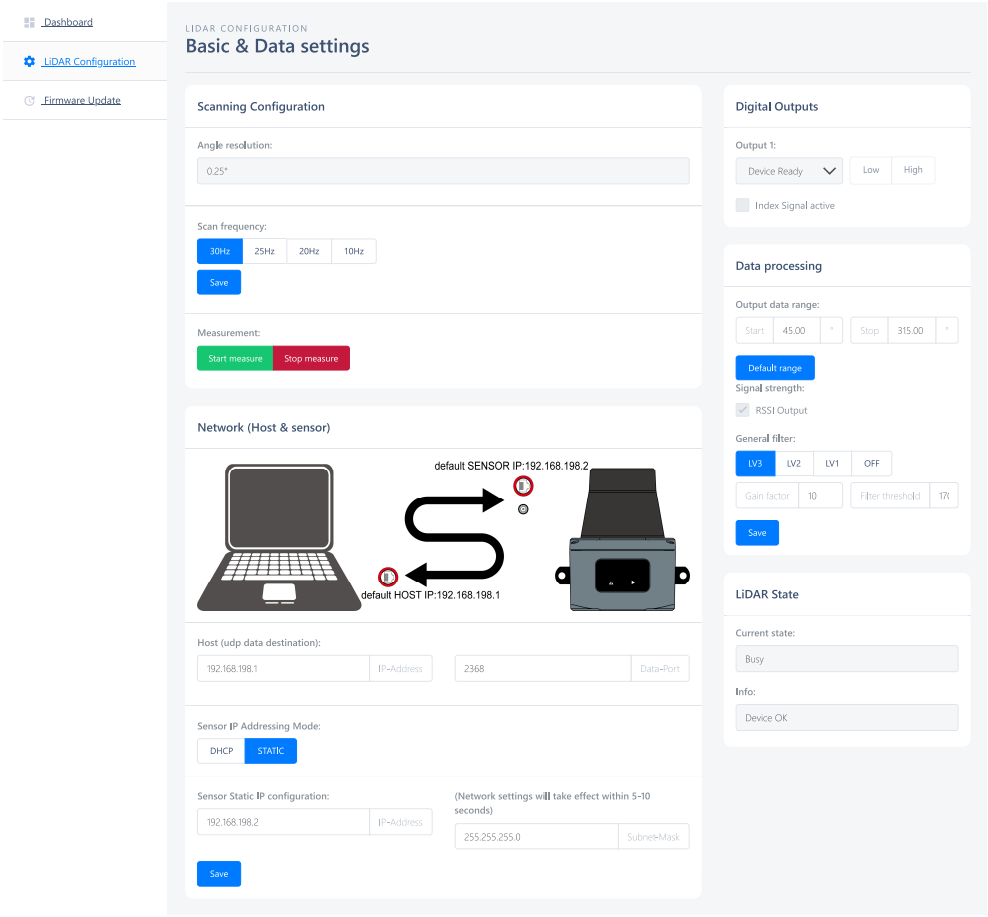


Figure 5.2: Lidar Configuration Page

5.4 Data Processing

The ideal light spot is a point when laser hiting the target, but in fact the laser emission has a divergence angle, the light spot is a surface when hitting the object. Hence, when there are two objects in front and one behind, and the laser just hit the edge of the object in front, it is possible that part of the same laser hit the object behind, then the reflected laser is the superposition of the reflected light of the two spots. In this case, the lidar will judge that the measurement target is between the two surfaces, resulting in veiling effect. Therefore, veiling effect is common in lidar, as shown in Figure 5.3.

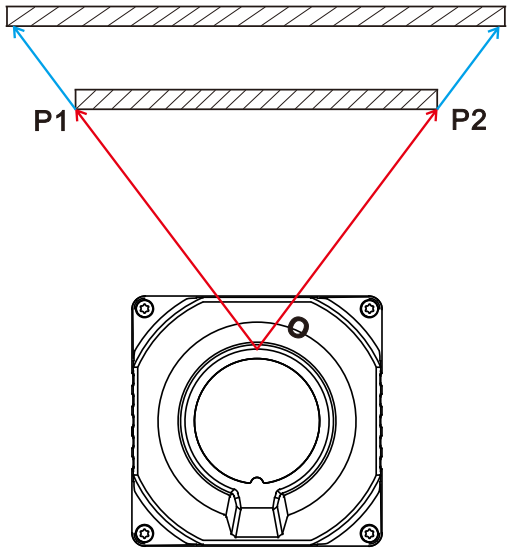


Figure 5.3: Principle of Lidar Veiling Effect

LiDAR1(L) not only provides filters to noise and jitter of the pointcloud, but also provides filters to veiling effect. Users can turn on and off the filter, and adjust filter type and intensity through the setting page of web server, as shown in Figure 5.4. Off indicates filter is disabled, LV1 indicates smoothing filter is enabled only, LV2 indicates that scan shadow filter is enabled only, and LV3 indicates that scan shadow and smoothing filters are enabled simultaneously.

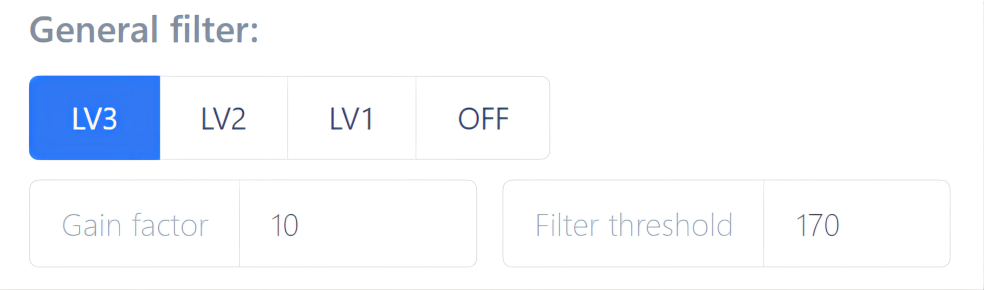


Figure 5.4: Lidar General Filter Configuration

Parameters of LiDAR1(L) veiling effect filter algorithm is as follows:
Gain_factor：indicating the maximum distance to control the filtering of veiling effect and point cloud
Filter_threshold：threshold of veiling effect filter;
Parameter setting (for reference only): Gain_factor：10
Filter_threshold：170

5.5 Firmware Update

Firmware Update page provides functions of lidar informations' display (Product Model, Serial Number, Hardware Version,FPGA Software Version, Firmware Version,AUX Software Version), firmware upload and lidar reset.

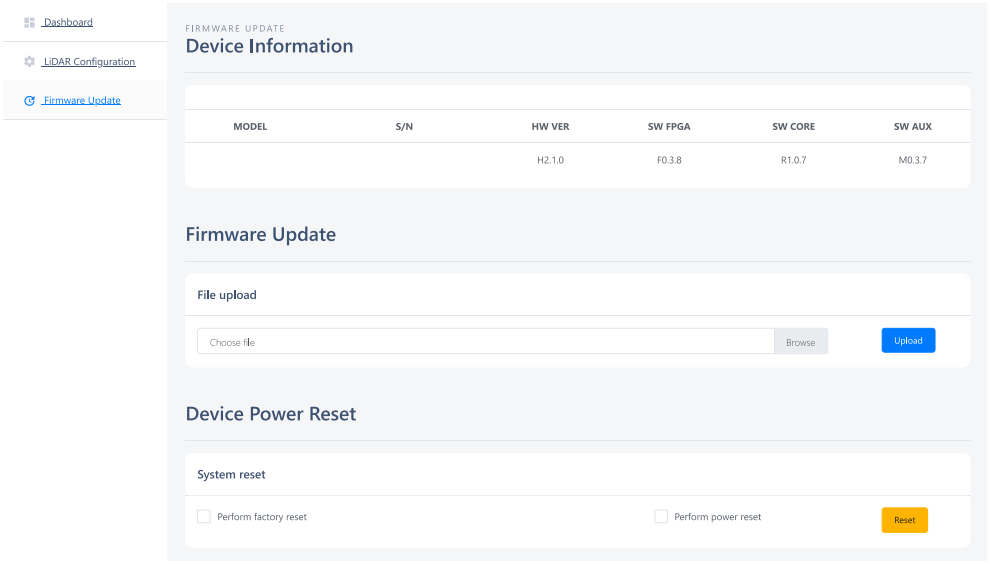


Figure 5.5: Firmware Update Page

6 Data Interface

LiDAR product will send ranging data to the Host continously when starting measurement. The communication adopts Ethernet as the transmission medium, and the packet is based on UDP protocol. LiDAR product also supports HTTP-based RESTful(Representational State Transfer) interface that allow users to proactively request lidar data and set the current lidar status.

6.1 MSOP

MSOP: Main data Stream Output Protocol. I/O type: device output data, computer parse data. Default port number is 2368.

MSOP packet outputs data information of lidar ranging,each packet consists of measuring distance, RSSI, azimuth and a time stamp.Each MSOP packet payload is 1248 byte long and consists of a 42-byte UDP header,a 1200-byte data block (a total of 12 100-byte Data Block) and a 6-byte tail,each tail contains a 4-byte timestamp and a 2-byte factory information.

The basic data structure of a MSOP packet is as shown in Figure 6.1:

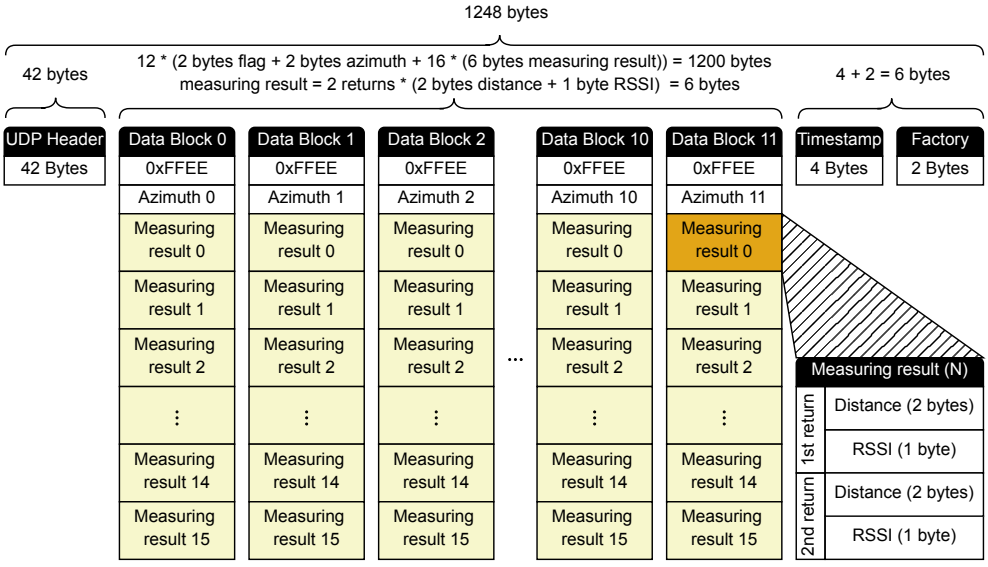


Figure 6.1: MSOP Packet Structure

One of the things to note is that in the MSOP packets output on each turn of the lidar, the last MSOP packet is not exactly the same as the contents of the other MSOP packets. In the case of 20 Hz, 25 Hz and 30 Hz, when one turn of data (1440 sets of data at 20 Hz, 25 Hz and 30 Hz) is output, there are still 96 sets of data left in the last MSOP packet. That is, the data in the last 6 Data Blocks are not filled, so all data from Data Block 6 to Data Block 11 is invalid data, and the flag bit and Azimuth are 0xFFFF, and all invalid range data is also 0xFFFF, as shown in Figure 6.2:

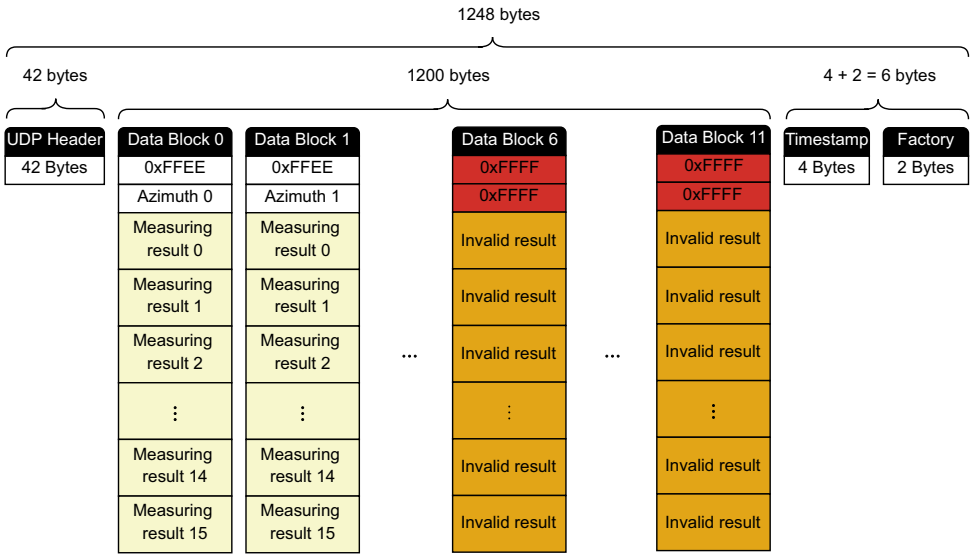


Figure 6.2: MSOP Last Packet Structure

6.2 Data Block

Data Block is a part of MSOP packet that contains lidar's measurement data,1200 bytes in total. It consists of 12 Data Blocks, each Block is 100bytes and represents a complete set of ranging data. The 100bytes Data Block contains: 2bytes flag bits: 0xFFEE; 2bytes Azimuth: horizontal rotation angle, each angle corresponds to 16 ranging data. In each Block, the horizontal angle of each ranging data can be calculated as follows: First, α_1 is the difference of the angle values of the two adjacent Blocks, then the angle increasment of two adjacent ranging data in each data Block is $\frac{\alpha_1}{16}$, the Azimuth of the Nth($0 \leq N \leq 15$) ranging data in the current Block can be calculated accordingly.

The horizontal angle value of the Nth ranging data in a Block (Block Angle value is Azimuth) is:

$$Azimuth_N = Azimuth + \frac{\alpha_1}{16} \times N \tag{6.1}$$

Each of the 16 ranging data contained in each Data Block is composed of 6bytes data. The first three bytes are the Strongest Return and the last three bytes are the Last Return. Each echo contains 2bytes of distance information and 1byte of RSSI, and the distance information is in mm.

In the calculation of Block azimuth, distance and timestamp, the data in MSOP package are big endian.

For example,in Figure 6.3, the azimuth of first Data Block in hexadecimal is: 0x80，0x25.

Combine to a 16 bit, unsigned integer: 0x2580.

Convert to decimal: 9600.

Divided by 100, then get the result: 96.00 degrees.

Hence, the horizontal angle of first Data Bloack is 96.00 degrees.

Similarly, the azimuth of second Data Block in hexadecimal is: 0x10，0x27, then get the result: 100.00 degrees.

Hence, the incremental value of the angle in Data Block is: (100 - 96) ÷ 16 = 0.25°.

The ranging data and intensity of first Data Block in hexadecimal is: 0xFC，0x08，0x31，0x00，0x00，0xFF. The last three bytes are the last return data, if the ranging information in last return data is 0, that indicates no return data. Therefore the demonstration only parses data of stronget return.

RSSI of stronget return is: 0x31, convert to decimal: 49.Hence, the RSSI of stronget return is 49.

Then parse the distance of stronget return, combine to a 16 bit, unsigned integer: 0x08FC.

Convert to decimal: 2300.

Divided by 100 and convert to meter: 2.3m.

azimuth：$96.00 + 0.25 \times 1 = 96.25$. Hence, the second ranging data in first Data Block is: azimuth: 96.25°, distance: 2.3m, RSSI: 49.

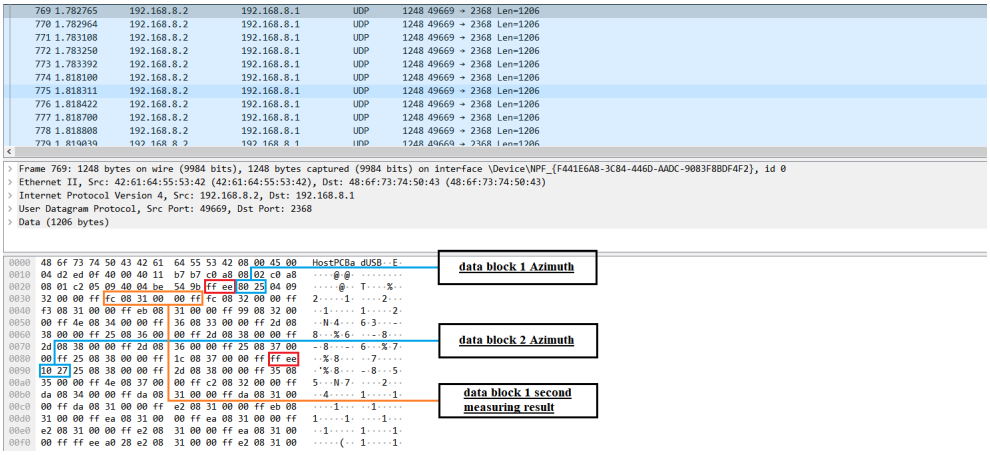


Figure 6.3: MSOP Packet Demonstration Data 1

6.3 Timestamp and Factory Bytes

Each MSOP package has 6bytes information which contains 4bytes of timestamp and 2bytes of factory bytes information at the end. The timestamp is used to record the system time with a resolution of 1us.

For example, in Figure 6.4, the 2bytes at the end of MSOP in hexadecimal is factory bytes information: 0x37，0x40.

The timestamp in hexadecimal is: 0x87，0x20，0xA8，0x0E。

Combine to a 32 bit, unsigned integer: 0x0EA82087。

Convert to decimal: 245899399。

Hence, the timestamp of this MSOP Packect is: 245899399us.

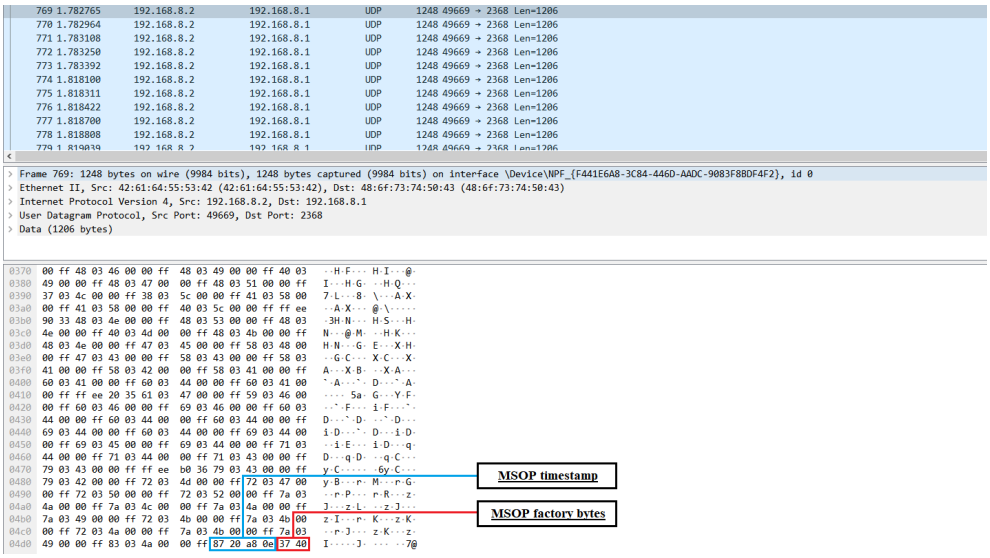


Figure 6.4: MSOP Packet Demonstration Data 2

6.4 RESTful API

LiDAR product supports HTTP-based RESTful(Representational State Transfer) interface. Users not only can connect to the lidar, view information, or modify settings through web server, but also can obtain the current lidar information or modify the current lidar settings through RESTful interface. The URL, HTTP verb, and HTTP status code of each specified resource of RESTful interface, see Appendix C: RESTful Interface instruction.

7 Assembly Guide

In order to ensure reliability and stability of the product, please strictly comply with the following product features and interface requirements during use. To avoid mutual interference of beams and influence on measuring accuracy, when assembling multiple lidars, ensure that the laser beam is not received by another lidar. If multiple products are assembled on the same plane, tilt the device downward or set the output data range on the lidar configuration page to avoid laser beam received by other products.When assembling multiple lidars, ensure that the laser beam is not received by another lidar.If multiple lidars are assembled on the same plane, tilt the device downward or set the scanning range on the configuration page to avoid mutual interference of beams, as shown in Figure 7.1.When multiple lidars are assembled in parallel on different planes, it is recommended to stagger the laser emitting position of the lidar by a certain distance to avoid mutual interference of beams, as shown in Figure 7.2. (The blue areas in Figure 7.1 and 7.2 represent expanded beam.)

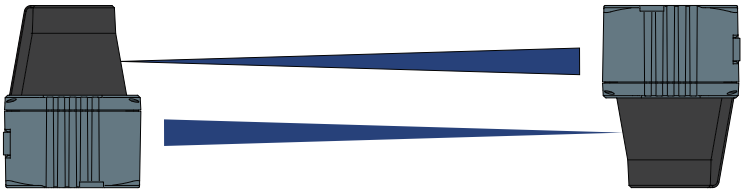


Figure 7.1: Parallel Offset Assembly

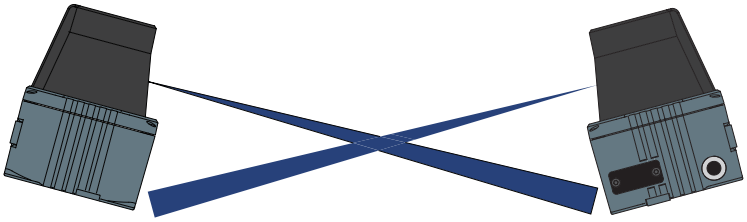


Figure 7.2: Parallel Assembly

7.1 Coordinate Transformation

Lidar's MSOP packet exports just azimuth value and distance data. But to transform a 2-dimensional pointcloud effect, a transformation of the azimuth value and distance data into x, y, z coordinates in accordance to Cartesian Coordinate System is necessary, as is shown in figure 7.3.

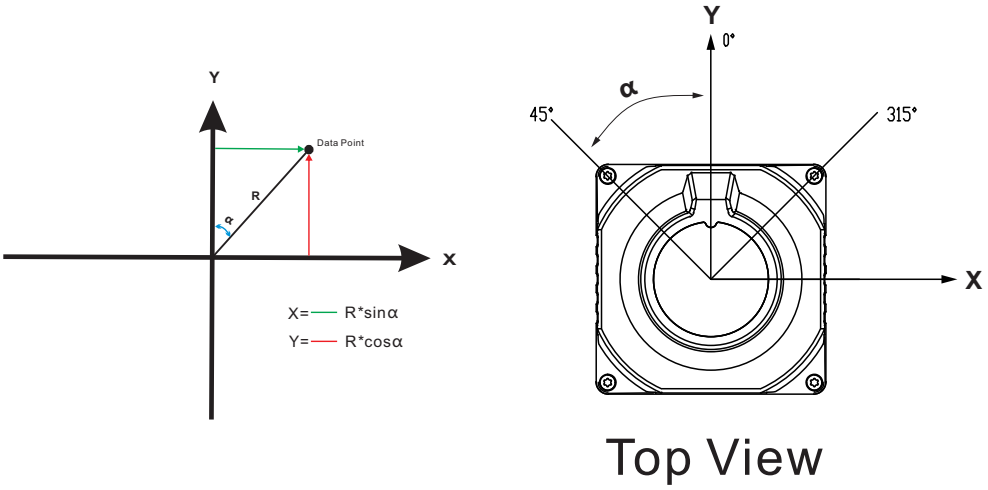


Figure 7.3: Lidar Polar Coordinate and X, Y, Z Coordinate Mapping

7.2 Optical Characteristics

LiDAR product has a slight deviation of laser vertical emission angle. The deviation range of laser vertical emission angle of each product is ±1° refered to horizontal plane. And the laser emission height is 56.75mm refered to the bottom surface of the product.

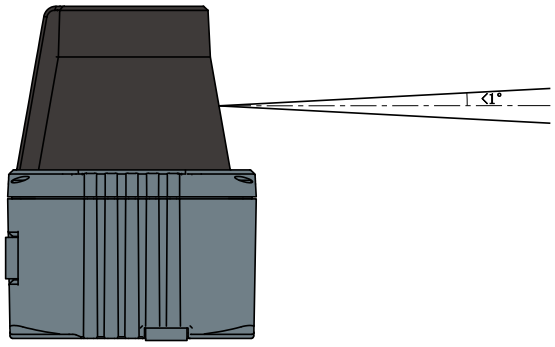


Figure 7.4: Vertical Emission Angle

The product has a scanning range of 45° to 315°, 270° in total.

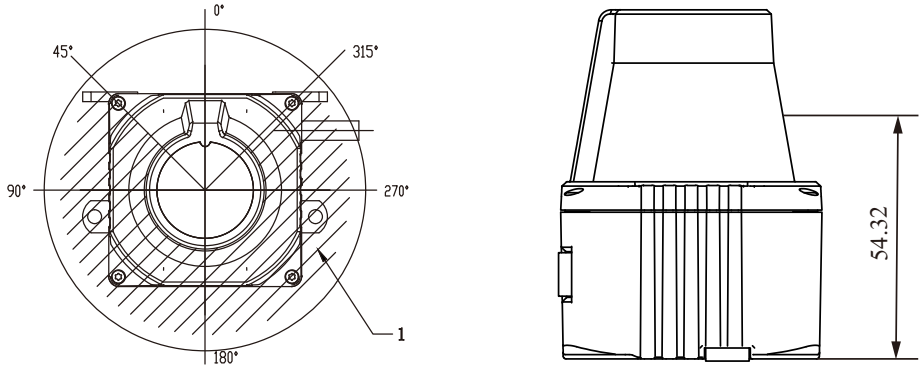


Figure 7.5: Lidar Horizontal Scanning Range and Vertical Beam Emission Height

1.Horizontal scanning filed is 270°.

7.3 Light Spot Size

As the distance from the lidar to the measured object increases, the laser beam expands.As a result, the diameter of the light spot on the surface of the object increases.In figure 8.4,the blue area represents expanded beam.

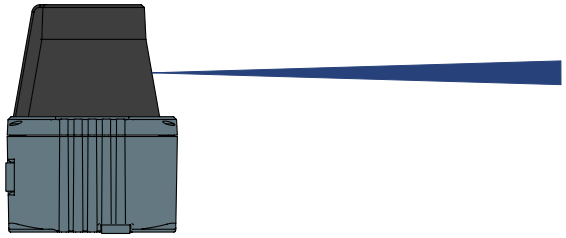


Figure 7.6: Beam Expansion Diagram

Required values for calculating the light spot size and minimum object size:
• Light spot size on the black cover lens:6mm (rounded up)
• Light spot divergence per single pulse:0.487 deg (8.5 mrad)
Formula for calculating the light spot width:
Light spot divergence [mrad] × distance [mm] + light spot size on the device black cover lens [mm] = Light spot diameter [mm]
Example calculation of the light spot diameter at a distance of 4 m:
$8.5 mrad \times 4,000 mm + 6 mm = 40 mm$

For reliable measurement, an object needs to be hit several times.Therefore， both the lidar and the object should be fixed.

7.4 Mechanical Interface

The following mechanical dimensions are in mm unless specified.

- **Lidar**

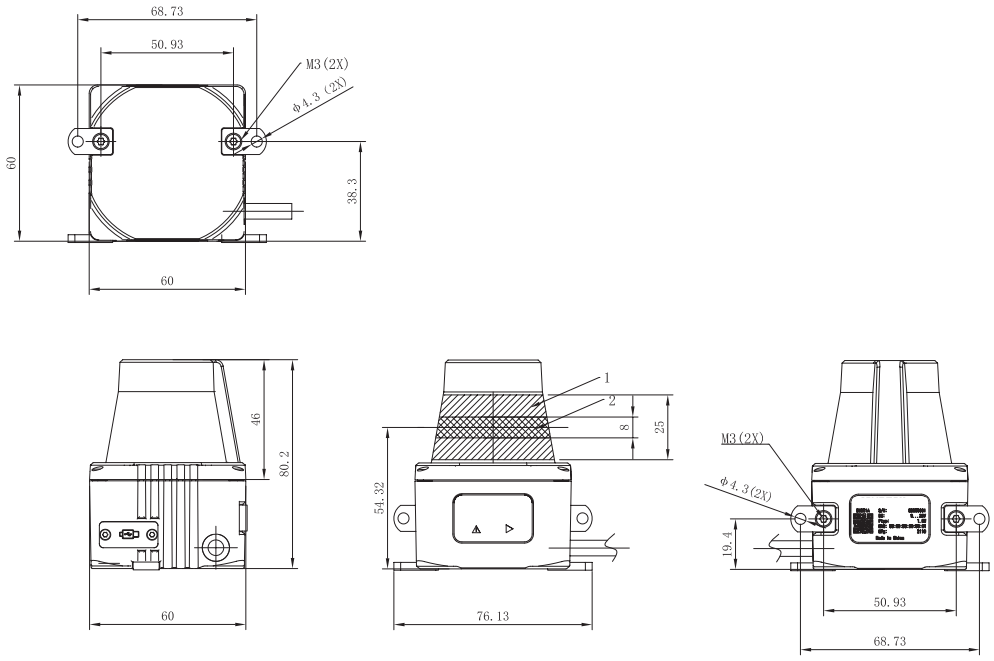


Figure 7.7: Lidar Dimension Drawing

1.Laser receiving range 2.Laser transmission range。

A Fault Diagnosis

B Sensor Maintenance Guide

Stains on the lidar's black cover lens, such as dirt, fingerprints,mud and oil,can negatively affect point cloud data quality after the lidar scans objects,the lidar should be cleaned specifically at this time. Please perform the following steps to remove the stains:

Never use sharp or rough objects to wipe the cover lens as this may cause damage to the device.

Please use recommended cleaning agents and tools to clean.

Please do not reuse cleaning tools, replace cleaning tools when cleaning the lidar several times or when they are contaminated. If the cleaning tool is contaminated, replace it promptly to ensure effective cleaning.

1) If the cover lens of lidar is just covered by dust or dirt, users only need to wear a pair of powder-free PVC gloves and gently wipe the cover lens with a piece of lint-free wipe or optical cleaning paper/mirror wipe paper.
2) If the cover lens of lidar is covered by stains such as fingerprints and oil, users need to wear a pair of powder-free PVC gloves, clean with a piece of lint-free wipe or optical cleaning paper/mirror wipe paper dipped in a little isopropyl alcohol and anhydrous ethanol solvent, wipe gently until to remove stains, and take a new clean piece of lint-free wipe or optical clean paper/mirror wipe paper wipe until the cover lens is completely no other stains.
3) If the cover lens of lidar is caked with mud or bugs, users need to wear a pair of powder-free PVC gloves, loose any debris with clean, warm water in a spray bottle, then clean with a piece of lint-free wipe or optical cleaning paper/mirror wipe paper dipped in a little isopropyl alcohol and anhydrous ethanol solvent, wipe gently until to remove stains, and take a new clean piece of lint-free wipe or optical clean paper/mirror wipe paper wipe until the cover lens is completely no other stains.

C RESTful API Instruction

Appendix C provides instruction and examples of RESTful interface to help users understand the LiDAR product and better use and develop the product.

System/Firmware

GET /api/v1/system/firmware

GET 192.168.8.2/api/v1/system/firmware
Get the lidar firmware information, which contains:product model,product serial number,hardware version,FPGA software version,firmware version and AUX software version.

```
GET /api/v1/system/firmware HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "model": "BM661",
    "sn": "BM000000",
    "hw": "1.0.0",
    "fpga": "F0.2.2",
    "core": "R0.4.4",
    "aux": "M0.1.9"
}
```

Response JSON Object

model(string) - Product Model
sn(string) - Product Serial Number
hw(string) - Hardware Version
fpga(string) - FPGA Software Version
core(string) - Firmware Version
aux(string) - AUX Software Version

Status Code

200 OK - No Error

System/Monitor

GET /api/v1/system/monitor

GET 192.168.8.2/api/v1/system/monitor
GET system monitor data,which contains:system load_average,memory usage and system uptime.

```
GET /api/v1/system/monitor HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "load_average": 0.19,
    "mem_useage": 51.72,
    "uptime": 4260.36
}
```

Response JSON Object

load_average(float) - System Load_Average
mem_useage(float) - Memory Usage
uptime(float) - System Uptime

Status Code

200 OK - No Error

System/Network

GET /api/v1/system/network

GET 192.168.8.2/api/v1/system/network
GET system network configuration, contains: state of Ethernet link,duplex mode of Ethernet link,ethernet hardware (MAC) address,hostname of the sensor and IPv4 network configuration.

```
GET /api/v1/system/network HTTP/1.1
Host: 192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "carrier": true,
    "duplex": "full",
    "ethaddr": "88:c9:b3:90:00:00",
    "hostname": "LiDAR",
    "ipv4": {
        "dhcp": false,
        "addr": "192.168.1.211/24",
        "override": "192.168.1.211/24"
    }
}
```

Response JSON Object

carrier(boolean) - State of Ethernet link, true when physical layer is connected.

duplex(string) - Duplex mode of Ethernet link, half or full.

ethaddr(string) - Ethernet hardware (MAC) address.

hostname(string) - Hostname of the sensor.

ipv4(object) - IPv4 network configuration

dhcp(boolean) - State of DHCP.

addr(string) - Static IP address.

override(string) - Static IP override value. This value will be null when unset and operating in DHCP modes.

speed(integer) - Ethernet physical layer speed in Mbps, should be 100 Mbps.

Status Code

200 OK - No Error

GET /api/v1/system/network/override

GET 192.168.8.2/api/v1/system/network/override
Get the current IPv4 static IP override value.

```
GET /api/v1/system/network/override HTTP/1.1
Host: 192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "override": "192.168.1.211/24"
}
```

Response JSON Object

override(string) - Static IP override value, this should match addr. This value will be null when unset and operating in DHCP mode.

Status Code

200 OK - No Error

PUT /api/v1/system/network/override

PUT 192.168.8.2/api/v1/system/network/override
Override the default dynamic behavior and set a static IP address.

PUT /api/v1/system/network/override HTTP/1.1 Host: 192.168.8.2 "192.168.1.211/24"
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 "192.168.1.211/24"

Request JSON Object

string - Static IP override value with subnet mask.

Response JSON Object

string - Static IP override value that system will set after a short delay.

Status Code

200 OK - No Error

DELETE /api/v1/system/network/override

DELETE 192.168.8.2/api/v1/system/network/override
Delete the static IP override value and return to dynamic configuration.

DELETE /api/v1/system/network/override HTTP/1.1 Host: 192.168.8.2
HTTP/1.1 204 No Content Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8

Status Code

204 No Content - No Error，no content

System/Reset

PUT /api/v1/system/reset

PUT 192.168.8.2/api/v1/system/reset
Set system reset.

```
PUT /api/v1/system/reset HTTP/1.1
Host:192.168.8.2
"reset"
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "cmd":"okay"
}
```

Request JSON Object

string - Set system reset.

Response JSON Object

string - This value will be null when unset.

Status Code

200 OK - No Error

Sensor/Overview

GET /api/v1/sensor/overview

GET 192.168.8.2/api/v1/sensor/overview
Get Lidar overview information, contains: scan frequency, real time motor_rpm, laser state, resolution, scan_range, filter level, Host IP and port.

```
GET /api/v1/sensor/overview HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "scanfreq": 30,
    "motor_rpm": 1795,
    "laser_enable": true,
    "resolution": 0.25,
    "scan_range": {
        "start": 45,
        "stop": 315
    }
    "filter": 0
    "host": {
        "ip": "192.168.8.1",
        "port": 2368
    }
}
```

Response JSON Object

scanfreq(integer) - Scan frequency, 4 gears in total:10Hz, 20Hz, 25Hz, 30Hz, unit: Hz.

motor_rpm(integer) - Motor real time speed, the theoretical speed corresponding to each scanning frequency is: 10Hz/600RPM，20Hz/1200RPM，25Hz/1500RPM，30Hz/1800RPM，unit: RPM(Revolutions Per minute).

laser_enable(boolean) - laser state, on or off.

resolution(float) - Sensor Resolution(0.1°or 0.25°).

scan_range(object) - Sensor scan range of angle.

start(integer) - Sensor start angle of scanning, min: 45°, max: 315°.

stop(integer) - Sensor stop angle of scanning, min: 45°, max: 315°.

filter(integer) - Current filter level, 3 gears in total: 0 is off.

host(object) - Host information.

ip(string) - Host IP address.

port(integer) - Host port.

Status Code

200 OK - No Error

Sensor/Scanfreq

GET /api/v1/sensor/scanfreq

GET 192.168.8.2/api/v1/sensor/scanfreq
GET sensor scan frequency, unit: mm.

```
GET /api/v1/sensor/scanfreq HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "scanfreq": 30
}
```

Response JSON Object

scanfreq(integer) - Sensor scan frequency,4 gears in total:10Hz, 20Hz, 25Hz, 30Hz.

Status Code

200 OK - No Error

PUT /api/v1/sensor/scanfreq

PUT 192.168.8.2/api/v1/sensor/scanfreq
Set sensor scan frequency, unit: Hz.

PUT /api/v1/sensor/scanfreq HTTP/1.1 Host:192.168.8.2 30
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 30

Request JSON Object

integer - Set sensor frequency.

Response JSON Object

integer - Motor speed will change slowly until it fluctuates near the setting value when sensor scan frequency is set.

Status Code

200 OK - No Error

Sensor/Motor_rpm

GET /api/v1/sensor/motor_rpm

GET 192.168.8.2/api/v1/sensor/motor_rpm
Get sensor motor real time spped, unit: RPM.

```
GET /api/v1/sensor/motor_rpm HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "motor_rpm": 1795
}
```

Response JSON Object

motor_rpm(integer) - Sensor motor real time speed, the theoretical speed corresponding to each scanning frequency is: 10Hz/600RPM，20Hz/1200RPM，25Hz/1500RPM，30Hz/1800RPM，unit: RPM(Revolutions Per minute).

Status Code

200 OK - No Error

Sensor/Laser_enable

GET /api/v1/sensor/laser_enable

GET 192.168.8.2/api/v1/sensor/laser_enable
Get sensor laser state.

```
GET /api/v1/sensor/laser_enable HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "laser_enable": true
}
```

Response JSON Object

laser_enable(boolean) - Sensor laser state.Start mearsurement when laser state is on only.

Status Code

200 OK - No Error

PUT /api/v1/sensor/laser_enable

PUT 192.168.8.2/api/v1/sensor/laser_enable
Switch sensor laser state.

PUT /api/v1/sensor/laser_enable HTTP/1.1 Host:192.168.8.2 true
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 true

Request JSON Object

boolean - Switch sensor laser state.

Response JSON Object

boolean - Sensor will switch to its corresponding state when swtiching to another state.

Status Code

200 OK - No Error

Sensor/Resolution

GET /api/v1/sensor/resolution

GET 192.168.8.2/api/v1/sensor/resolution
Get sensor resolution.

```
GET /api/v1/sensor/resolution HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "resolution": 0.25
}
```

Response JSON Object

resolution(float) - current sensor resolution, resolution corresponding to each scanning frequency is: 10Hz/0.1°, 20Hz/0.25°, 25Hz/0.25°, 30Hz/0.25°

Status Code

200 OK - No Error

Sensor/Scan_range

GET /api/v1/sensor/scan_range

GET 192.168.8.2/api/v1/sensor/scan_range
Get sensor scan range of angle.

```
GET /api/v1/sensor/scan_range HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "scan_range": {
        "start": 45,
        "stop": 315
    }
}
```

Response JSON Object

scan_range(object) - Sensor scan range of angle.
start(integer) - Sensor start angle of scanning.
stop(integer) - Sensor stop angle of scanning.

Status Code

200 OK - No Error

GET /api/v1/sensor/scan_range/start

GET 192.168.8.2/api/v1/sensor/scan_range/start
Get sensor start angle of scan range.

```
GET /api/v1/sensor/scan_range/start HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "start": 45,
}
```

Response JSON Object

start(integer) - Start angle value.

Status Code

200 OK - No Error

GET /api/v1/sensor/scan_range/stop

GET 192.168.8.2/api/v1/sensor/scan_range/stop
Get sensor stop angle of scan range.

```
GET /api/v1/sensor/scan_range/stop HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "stop": 315,
}
```

Response JSON Object

stop(integer) - Stop angle value.

Status Code

200 OK - No Error

PUT /api/v1/sensor/scan_range/start

PUT 192.168.8.2/api/v1/sensor/scan_range/start
Set sensor start angle of scan range.

PUT /api/v1/sensor/scan_range/start HTTP/1.1 Host:192.168.8.2 45
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 45

Request JSON Object

integer - Start angle value.

Response JSON Object

integer - Sensor start angle of scanning, min: 45°, max: 315°.

Status Code

200 OK - No Error

PUT /api/v1/sensor/scan_range/stop

PUT 192.168.8.2/api/v1/sensor/scan_range/stop
Set sensor stop angle of scan range.

PUT /api/v1/sensor/scan_range/stop HTTP/1.1 Host:192.168.8.2 315
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 315

Request JSON Object

integer - Stop angle value.

Response JSON Object

integer - Sensor stop angle of scanning, min: 45°, max: 315°.

Status Code

200 OK - No Error

Sensor/Filter

GET /api/v1/sensor/filter

GET 192.168.8.2/api/v1/sensor/filter

Get current sensor filter level.

```
GET /api/v1/sensor/filter HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "filter": 3
}
```

Response JSON Object

filter(integer) - Current filter level, 3 gears in total: 0 is off.

Status Code

200 OK - No Error

PUT /api/v1/sensor/filter

PUT 192.168.8.2/api/v1/sensor/filter
Set sensor filter level.

PUT /api/v1/sensor/filter HTTP/1.1 Host:192.168.8.2 3
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 3

Request JSON Object

integer - Set sensor filter level value.

Response JSON Object

integer - value of filter level, 3 gears in total: 0 is off.

Status Code

200 OK - No Error

Sensor/Host

GET /api/v1/sensor/host

GET 192.168.8.2/api/v1/sensor/host
Get current sensor host information.

```
GET /api/v1/sensor/host HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "host": {
        "ip": "192.168.8.1",
        "port": 2368
    }
}
```

Response JSON Object

host(object) - Host information.
ip(string) - Host IP address.
port(integer) - Host port.

Status Code

200 OK - No Error

GET /api/v1/sensor/host/ip

GET 192.168.8.2/api/v1/sensor/host/ip
Get current sensor Host IP address.

```
GET /api/v1/sensor/host/ip HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
      "ip": "192.168.8.1",
}
```

Response JSON Object

ip(string) - Host IP address.

Status Code

200 OK - No Error

GET /api/v1/sensor/host/port

GET 192.168.8.2/api/v1/sensor/host/port
Get current sensor Host port.

```
GET /api/v1/sensor/host/port HTTP/1.1
Host:192.168.8.2
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Content-Type: application/json; charset=UTF-8
{
    "port": 2368,
}
```

Response JSON Object

port(integer) - Host port

Status Code

200 OK - No Error

PUT /api/v1/sensor/host/ip

PUT 192.168.8.2/api/v1/sensor/host/ip
Set sensor Host IP address.

PUT /api/v1/sensor/host/ip HTTP/1.1 Host:192.168.8.2 192.168.8.1
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 192.168.8.1

Request JSON Object

string - Set sensor Host IP address.

Response JSON Object

string - Host IP address.

Status Code

200 OK - No Error

PUT /api/v1/sensor/host/port

PUT 192.168.8.2/api/v1/sensor/host/port

Set sensor Host IP port.

PUT /api/v1/sensor/host/port HTTP/1.1 Host:192.168.8.2 2368
HTTP/1.1 200 OK Server: nginx/1.18.0 Content-Type: application/json; charset=UTF-8 2368

Request JSON Object

integer - Set sensor Host IP port.

Response JSON Object

integer - Host IP port.

Status Code

200 OK - No Error