Date:2025/11/05

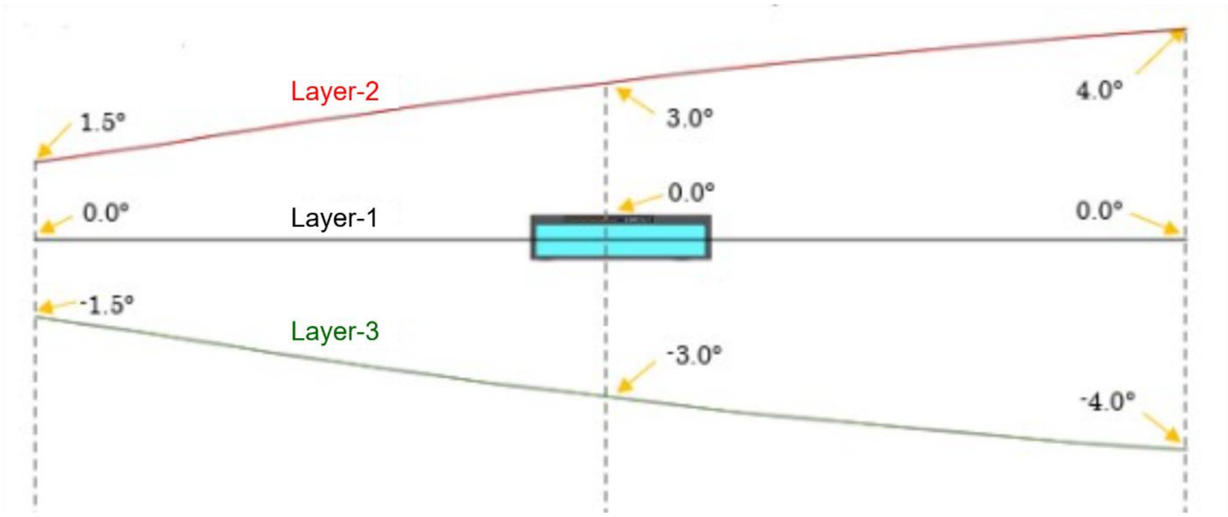
UCT series VSSP Communication Protocol Specification

SYMBOL	AMENDED REASON	PAGES	DATE	AMENDED BY	REFERENCE

APPROVED	CHECKED	DRAFTED	DESIGNED	TITLE	UCT series VSSP Communication Protocol Specification	
T.kamitani	Hosoda	Momodori	Momodori	DRAWING NO.	C-42-04610	1／25

1 Overview

The UCT series is a 3D scanning sensor that combines scanning by a mirror and a motor, as shown in the figure below, and supports communication using the Volumetric Scanning Sensor Protocol (VSSP) Version 2.3.
This document describes the VSSP communication specification for the UCT series.



2 Communication Method

The session layer uses TCP/IP. Error detection and correction are handled by the layers below the session layer and are not performed by VSSP itself.

Factory default network settings are as follows:

Item	Default	Notes
IP Address	192.168.0.10	(Changeable via IPDiscovery, URGBenri)
Subnet Mask	255.255.255.0	(Changeable via IPDiscovery, URGBenri)
Default Gateway	192.168.0.1	(Changeable via IPDiscovery, URGBenri)
Port	10940 (fixed)	—

3 Definition of Measurement Data

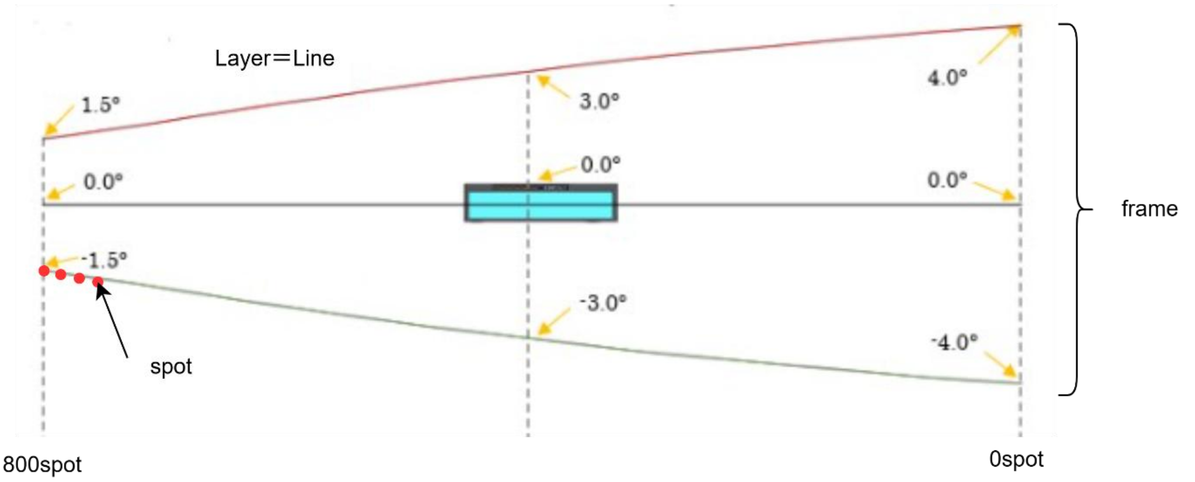
The unit of measurement data obtained by scanning is defined as follows.

This sensor performs measurements in three layers with different vertical angles. In the VSSP protocol, the set of one layer is called a "line", and the set of three lines (three layers) is called a "frame".

The number of spots per line is 801 spots, from spot 0 to spot 800. When viewing the sensor from the front, the right side corresponds to spot 0.

Names and meanings of measurement data:

Notation	Meaning
Echo	One measurement in a specific direction.
Spot	All echoes in a specific direction (a group of multiple echoes, equivalent to a "step" in SCIP).
Line	A group of spots for one mirror (one layer).
Frame	A group of three lines corresponding to one motor scan cycle.



4 Vertical Interlacing

To obtain measurement data for all three layers, the vertical interlace count must be set to "3".

The number of vertical interlaces that can be set can be obtained from the parameter "spec.remInterlaceCount". In multi-layer measurement mode it is "3", and in single-layer measurement mode it is "1".

The measurement mode can be changed using AreaDesignerPrime, and the setting is retained even when the power is turned off.

In VSSP, layers 1, 2, and 3 are referred to as vertical fields 1, 2, and 3 respectively.

5 Data Representation Format

VSSP uses both human-readable character strings and binary data. In this document, the notation of data is defined as follows.

Notation	Meaning
‘A’	A fixed readable character of 1 byte (for example, the character 'A').
“ABC”	A fixed character string.
C(n)	A readable character string of n bytes (for example, C(4): a 4-character readable string).
C(*)	A variable-length readable character string.
LF	Line feed = 0x0A.
CR	Carriage return = 0x0D.
RT	Terminator = LF, CR, or LF + CR.
U8	Unsigned 1-byte integer.
U16	Unsigned 2-byte integer, little-endian.
S16	Signed 2-byte integer, little-endian.
U32	Unsigned 4-byte integer, little-endian.
S32	Signed 4-byte integer, little-endian.
RSV	Reserved area. Empty data = 0x00.

Note: Data is aligned so that the total size is a multiple of 4 bytes (4-byte alignment).

6 Commands

Commands consist of request commands sent from the host to the sensor and response commands sent from the sensor to the host. VSSP response commands are classified into three types:

· Handshake type

· Continuous data transmission type

· Spontaneous error notification type

6.1 Basic Format

Basic format of a request command:

【Request Command】

command code(C(3))	parameters(C(*))	RT

- command code: A 3-character string that indicates the command to execute.
- parameters: Parameters for the command. The length and meaning depend on the command.
- RT: Request terminator (one character of CR or LF, or two characters of CR+LF). When a request command is echoed back in a response command, the terminator of the echoed request is always LF.

Basic format of a response command:

【Response Command】

<table>
<tr><td colspan="2">"VSSP"</td><td>packet type(C(3))</td><td>':'</td><td>status(C(3))</td><td>LF</td></tr>
<tr><td>header bytes(U16)</td><td>response bytes(U16)</td><td colspan="2">request timestamp(U32)</td><td colspan="2">response timestamp(U32)</td></tr>
<tr><td colspan="6">data</td></tr>
</table>

VSSP common header (the first two rows)

- packet type: Type of packet.
- status: Status code.

Category	Code	Description
Normal	"000"	Works well
	"021"	Internal process overload timeout
	"022"	Transmission timeout
	"098"	Sleeping
	"099"	Initializing…
Request command error	"101"	Command is unknown
	"102"	Command structure is mismatch
	"103"	Command parameter is mismatch
	"104"	Command parameter is out of range
	"105"	Element count is mismatch
	"106"	Couldn't get this parameter
System error	"201"	System boot failed
	"202"	System fault

- header bytes: Number of bytes in the header (24 fixed).
- response bytes: Total number of bytes in the response command.
- request timestamp: Timestamp when the request command was received [ms].
- response timestamp: Timestamp when the response command was sent [ms].
- data: Response data for each command (some commands have no data).

6.2 Response Check (PNG Command)

This command is used only to confirm that a response can be returned; it does not return any special information.

【Request Command】

"PNG"	RT

【Response Command】

VSSP common header

- VSSP common header (packet type: "PNG")

6.3 Version Information (VER Command)

This command obtains a set of version-related parameters that can also be obtained by the GET command.

【Request Command】

"VER"	RT

【Response Command】

VSSP common header			
"vend"(C(4))	':'	vendor name(C(*))	LF
"prod"(C(4))	':'	product name(C(*))	LF
"firm"(C(4))	':'	firmware version(C(*))	LF
"prot"(C(4))	':'	protocol version(C(*))	LF
"seri"(C(4))	':'	serial number(C(*))	LF

- VSSP common header (packet type: "VER")
- vendor name: Vendor name.
- product name: Product model.
- firmware version: Firmware version.
- protocol version: Protocol version.
- serial number: Serial number.

6.4 Get Parameters (GET Command)

This command reads the value of a parameter. See the parameter table on the next pages for the types of parameters.

【Request Command】

"GET"	':'	parameter name(C(*))	RT

- parameter name: Parameter name (see parameter table).

【Response Command】

VSSP common header	
echo back(C(*))	LF
parameter value(C(*))	LF
parameter value(C(*))	LF
…	

- VSSP common header (packet type: "GET")
- echo back: Echoed request command (terminator converted to LF).
- parameter value: Parameter values (at least one line; may span multiple lines).

6.5 Set Parameters (SET command)

This command writes a value to a parameter. Parameters that can be written are listed in the parameter table.

【Request Command】

"SET"	':'	parameter name (C(*))	'='	parameter value (C(*))	RT

- parameter name: Parameter name (RW parameters in the parameter table).
- parameter value: Value to be written to the parameter.

【Response Command】

VSSP common header	
echo back(C(*))	LF

- VSSP common header (packet type: "SET")
- echo back: Echoed request command (terminator converted to LF).

- Parameter Table

There are two types of parameters: read-only parameters and read/write parameters.

In the parameter list they are indicated as follows.

Type	Meaning
RO	Read-only parameter (read only parameter)
RW	Read/write parameter (read/write parameter)

The parameters are listed below.

<table>
<tr><th>parameter</th><th>Type</th><th>Description</th></tr>
<tr><td>“vend”</td><td>RO</td><td>Vendor name.</td></tr>
<tr><td>“prod”</td><td>RO</td><td>Product name.</td></tr>
<tr><td>“firm”</td><td>RO</td><td>Firmware version.</td></tr>
<tr><td>“prot”</td><td>RO</td><td>Protocol version.</td></tr>
<tr><td>“seri”</td><td>RO</td><td>Serial number.</td></tr>
<tr><td>“_itv”</td><td>RW</td><td>Vertical interlacing setting.
The following character strings specify each setting:
<table>
<tr><th>Setting value</th><th>Meaning</th></tr>
<tr><td>“0,01”</td><td>Vertical interlace count 1</td></tr>
<tr><td>“0,02”</td><td>Vertical interlace count 2</td></tr>
<tr><td>“0,03”</td><td>Vertical interlace count 3</td></tr>
</table>
The available setting range depends on the sensor. For this sensor it is as follows:
<table>
<tr><th>Sensor</th><th>Available setting range</th></tr>
<tr><td>UCT series</td><td>1～3</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td>"tblh[00]"
"tblh[01]"
"tblh[02]"
"tblh[03]"</td><td>RO</td><td>Vertical direction coordinate conversion table (single layer)
<table>
<tr><td>Usage</td><td>Used for calculating the vertical direction (angle) of each spot in a line.The index n corresponds to a spot group divided into 256 spots.
Example:
tblh[01]:vertical-direction coordinate conversion table for spots 256–511.</td></tr>
<tr><td>Numerical value</td><td>For each spot in the line, this is the vertical offset from the first spot in the line,
represented as a 16-bit value where the first spot is 0 and the last spot is 65535 (relative-angle table).</td></tr>
<tr><td>Value representation</td><td>Numeric representation: readable hexadecimal characters
Separator: "," (comma)
Spot groups:
[00] 0–255 spots
[01] 256–511 spots
[02] 512–767 spots
[03] 768–800 spots</td></tr>
</table></td></tr>
<tr><td>"tblv[00]"
"tblv[01]"
"tblv[02]"
"tblv[03]"</td><td>RO</td><td>Horizontal direction coordinate conversion table (common to single-layer / multi-layer
<table>
<tr><td>Usage</td><td>Horizontal direction coordinate conversion table (common to single-layer / multi-layer)
Example: tblv[01] : horizontal-direction coordinate conversion table for spots 256–511.</td></tr>
<tr><td>Numerical value</td><td>The horizontal direction is expressed as the absolute angle seen from the sensor, with 0° in the vertical direction and the left direction taken as positive, covering 0–360° represented as a 16-bit value (absolute-angle table).</td></tr>
<tr><td>Value representation</td><td>Numeric representation: readable hexadecimal characters
Separator: "," (comma)
Spot groups:
[00] 0–255 spots
[01] 256–511 spots
[02] 512–767 spots
[03] 768–800 spots</td></tr>
</table></td></tr>
</table>

<table>
<tr><td>“tv00[00]”
“tv00[01]”
“tv00[02]”
“tv00[03]”
“tv01[00]”
“tv01[01]”
“tv01[02]”
“tv01[03]”
“tv02[00]”
“tv02[01]”
“tv02[02]”
“tv02[03]”</td><td>RO</td><td>Vertical direction coordinate conversion table (multi-layer)
<table>
<tr><td>Usage</td><td>Used for calculating the vertical direction (sensor rotation direction) of each spot in a line.The characters that follow tv indicate the vertical interlace number. The index n indicates a spot group divided into 256 spots.
Example: tv01[01] : vertical-direction coordinate conversion table for Layer 2, spots 256–511.</td></tr>
<tr><td>Numerical value</td><td>For each spot in the line, this is the vertical offset from the first spot in the line.It is represented as a 16-bit value whose first spot is 0 and last spot is 65535 (relative-angle table).</td></tr>
<tr><td>Value representation</td><td>Numeric representation: readable hexadecimal characters
Separator: “,” (comma)
Spot groups:
[00] 0–255 spots
[01] 256–511 spots
[02] 512–767 spots
[03] 768–800 spots</td></tr>
</table></td></tr>
<tr><td>“data”</td><td>RO</td><td>Returns the transmission status of the measurement data.
For details of the measurement data, see Section 6.6.
<table>
<tr><th>Value</th><th>Meaning</th></tr>
<tr><td>‘0’</td><td>Transmission stopped</td></tr>
<tr><td>‘1’</td><td>Transmitting</td></tr>
</table>
Example:When only _ro (distance-only data) is being transmitted
_ro=1
_ri=0
_ax=0</td></tr>
</table>

<table>
<tr><td>"stat"</td><td>RO</td><td>Returns whether measurement of the data is possible.
For details of the measurement data, see Section 6.6.
<table><tr><th>Setting value</th><th>Meaning</th></tr><tr><td>"000;Works well"</td><td>Measurement possible</td></tr><tr><td>"099;Initializing..."</td><td>Measurement not possible (during startup)</td></tr><tr><td>"105;Command is not currently available"</td><td>Measurement not possible (sensor in standby mode)</td></tr><tr><td>"202;System fault"</td><td>Measurement not possible (sensor failure)</td></tr></table>
Example) When all data can be measured
_ro=000;Works well
_ri=000;Works well
_ax=000;Works well</td></tr>
<tr><td>"swdr"</td><td>RO</td><td>Represents the scanning direction. For this sensor the value is fixed to 1.</td></tr>
<tr><td>"spec.*"</td><td>RO</td><td>Parameter for obtaining sensor specification information.
By substituting the * in spec.* with one of the specification item names shown below, each piece of information can be retrieved.
<table><tr><th>Name</th><th>Type</th><th>Meaning</th><th>Value</th></tr><tr><td>"spec.remInterlaceCount"</td><td>Decimal integer</td><td>Maximum vertical interlace count</td><td>3</td></tr><tr><td>"spec.firstLineNumber"</td><td>Decimal integer</td><td>Measurement start line number</td><td>0</td></tr><tr><td>"spec.lastLineNumber"</td><td>Decimal integer</td><td>Measurement end line number</td><td>0</td></tr><tr><td>"spec.spotCount"</td><td>Decimal integer</td><td>Maximum number of spots</td><td>801</td></tr><tr><td>"spec.echoCount"</td><td>Decimal integer</td><td>Maximum number of echoes</td><td>3</td></tr><tr><td>"spec.accelScale"</td><td>Decimal integer</td><td>Accelerometer scale [g]</td><td>16</td></tr><tr><td>"spec.gyroScale"</td><td>Decimal integer</td><td>Gyroscope scale[dps]</td><td>2000</td></tr></table></td></tr>
</table>

6.6 Start/Stop Measurement Data Acquisition (DAT Command)

This command starts or stops acquisition of measurement data. After the response to the DAT command, the requested measurement data is sent from the sensor to the host in continuous transmission format. Only the first response contains the VSSP common header and an echo back of the request command; subsequent responses are sent only in the continuous data formats described in Sections 6.6.1 and 6.6.2.

【Request Command】

"DAT"	':'	data type(C(2))	'='	state(C(1))	RT

- data type : Measurement data format

data type	Meaning
"ro"	distance-only
"ri"	distance+intensity
"ax"	auxiliary

- state : Start/stop state. A value that selects whether to start or stop transmission of the specified data type.

state	Meaning	Remarks
'0'	Stop	・For ro and ri, specifying either one stops acquisition of data in both formats. ・For ax, only acquisition of ax data is stopped (it does not affect the acquisition state of ro or ri).
'1'	Start	・For ri and ro, the setting that is configured later takes effect. ・Starting ax begins acquisition of ax data (it does not affect the acquisition state of ro or ri).

【Response Command】

VSSP common header	
echo back(C(*))	LF

- VSSP common header (packet type: "DAT")
- echo back of the request command (terminator converted to LF)

6.6.1 ro/ri Format Data (_ro / _ri Response Command)

These are the distance or distance+intensity data that are sent continuously in response to a DAT start request. The distance-only format is returned as an "_ro" packet and the distance+intensity format is returned as an "_ri" packet.

【Response Command】

VSSP common header
Distance measurement data header
Echo index array
Measurement data array

- VSSP common header (packet type: "_ro"/"_ri")

【Distance measurement data header】

header bytes(U16)	head timestamp(U32)			tail timestamp(U32)			head direction(U16)
tail direction(U16)	frame num(U8)	horizontal field num(U8)	line num(U16)	head spot num(U16)	vertical field num(U8)	vertical interlace num(U8)	RSV(U16)

- header bytes: Number of bytes in the distance measurement data header.
- head timestamp: Timestamp of the first spot in the line [ms].
- tail timestamp: Timestamp of the last spot in the line [ms].
- head direction: Vertical angle of the first spot in the line. The front direction is 0°, upward is positive, and the range 0–360° is represented as a 16-bit value.
- tail direction: Vertical angle of the last spot in the line. The front direction is 0°, upward is positive, and the range 0–360° is represented as a 16-bit value.
- frame num: Frame number (fixed to 0 for this sensor).
- horizontal field num: Horizontal field number (fixed to 0 for this sensor).
- line num: Line number (fixed to 1 for this sensor).
- head spot num: Spot number of the first data in this line.
- vertical field num: Vertical field index. This field is present only when the vertical interlace count is 2 or greater.
- vertical interlace num: Vertical interlace count. This field is present only when the vertical interlace count is 2 or greater.

【Echo index array】

echo index array num(U16)	spot num(U16)	spot position[0](U16)	spot position[1](U16)	…
…	spot position[n](U16)	all echo num(U16)	RSV*1	RSV*1

- echo index array num: Byte length of the echo index array.
- spot num: Number of spots included in this data.
- spot position[0], spot position[1], …, spot position[n]: For each spot, the position of the first echo in the expanded measurement data array. The number of elements is equal to spot num.
- all echo num: Total number of echoes in this line.

*1 The echo index array may include reserved (padding) bytes so that the total size of the array is aligned in 32-bit units, depending on the number of spots.

【Measurement data array】

measurement data[0](U16)	measurement data[1](U16)	measurement data[2](U16)	…	measurement data[n](U16)

For _ro (distance-only) format, the measurement data array contains distance values only.
For _ri (distance+intensity) format, the measurement data array contains alternating distance and intensity values.

Specifically:
・_ro format:　measurement_data[0], measurement_data[1], …, measurement_data[n]
Each element is a distance value [mm]. A total of all_echo_num distances are stored.
・_ri format:　measurement_data[0], measurement_data[1], …, measurement_data[2 * n + 1]
Even-indexed elements (0, 2, 4, …) are distance values [mm].
Odd-indexed elements (1, 3, 5, …) are intensity values.

A concrete example of an _ri response and detailed explanation of the echo index array and measurement data array is given in the Appendix (section 7.1).

6.6.2 ax Format Data (_ax Response Command)

Auxiliary sensor data (such as angular velocity and acceleration) is sent continuously in response to a DAT start request for auxiliary data. This data is returned in packets with packet type "_ax".

【Response Command】

VSSP common header
Auxiliary sensor data header
Auxiliary sensor data

- VSSP common header (packet type: "_ax")

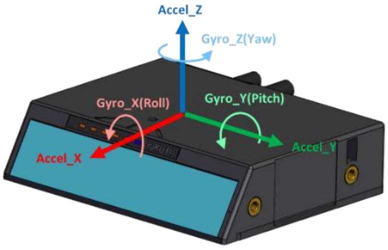
【Auxiliary sensor data header】

header bytes(U16)	head timestamp(U32)	data type(U32)	data num(U8)	period(U8)

- header bytes: Number of bytes in the auxiliary sensor data header (12 bytes).
- head timestamp: Timestamp of the first auxiliary data sample [ms].
- data type: Bit field that represents the type of data contained in the auxiliary data array. For this sensor, bits 32–27 are used, so the data type is always “0xFC000000”.

Bit	Description
32	Angular velocity X
31	Angular velocity Y
30	Angular velocity Z
29	Acceleration X
28	Acceleration Y
27	Acceleration Z
26～1	Reserved

The axes for angular velocity and acceleration are defined so that the sensor’s front, left, and upward directions correspond to the positive X, Y, and Z axes respectively.



- data num: Number of auxiliary data samples contained in this packet.
- period: Time interval between auxiliary data samples [ms].

【Auxiliary sensor data】

data[0](S32)	data[1](S32)	…	data[5](S32)

Data name	Contents	Remarks
data[0]	Angular velocity X	A value in the range ±2000dps is represented as a signed 16-bit integer (-32768 to 32767). Example when the value is 10000: 10000 × 2000 / 32768 = 610.35 [dps].
data[1]	Angular velocity Y	A value in the range ±2000dps is represented as a signed 16-bit integer (-32768 to 32767).
data[2]	Angular velocity Z	A value in the range ±2000dps is represented as a signed 16-bit integer (-32768 to 32767).
data[3]	Acceleration X	A value in the range ±16g is represented as a signed 16-bit integer (-32768 to 32767). Example when the value is 10000: 10000 × 16 / 32768 = 4.88[g]
data[4]	Acceleration Y	A value in the range ±16g is represented as a signed 16-bit integer (-32768 to 32767).
data[5]	Acceleration Z	A value in the range ±16g is represented as a signed 16-bit integer (-32768 to 32767).

6.7 Reboot (RST Command)

This command restarts the sensor. The behavior is equivalent to turning the power off and on again.

【Request Command】

“RST”	RT

【Response Command】

VSSP common header

- VSSP common header (packet type: "RST")

6.8 Error Information (ERR Command)

When the sensor receives an undefined command, a command with an invalid format, or when it cannot return a normal response to a request command because of its internal state, it returns an "ERR" response command instead of the normal response.

【Response Command】

VSSP common header	
echo back(C(*))	LF
information(C(*))	LF
…	
information(C(*))	LF

- VSSP common header
  (packet type: "ERR")
  (status: a non-"0**" status code indicating the error condition; see the status list)
- echo back: Echoed request command (with the terminator converted to LF).
- information: Error information. Contains one or more lines of text and may span multiple lines.

6.9 er Format Data (_er Response Command)

This is a spontaneous notification that the sensor sends when its internal error state changes. It is not a response to a specific request command.

【Response Command】

VSSP common header	
information(C(*))	LF
…	
information(C(*))	LF

- VSSP common header
  (packet type: "_er")
  (status: a non-"0**" status code indicating the current error condition)
- information: Error information. Contains one or more lines of text and may span multiple lines.

7 Appendix

7.1 Explanation of Measurement Data Acquisition Commands

Section 6.6.1 describes the format of the measurement data. This appendix shows a concrete example of an _ri-format distance+intensity data packet in both binary and ASCII, and explains each part. Only the continuous-transmission response format is described here.

An _ri response packet consists of the following four blocks in order:

1) VSSP common header

2) Distance measurement data header

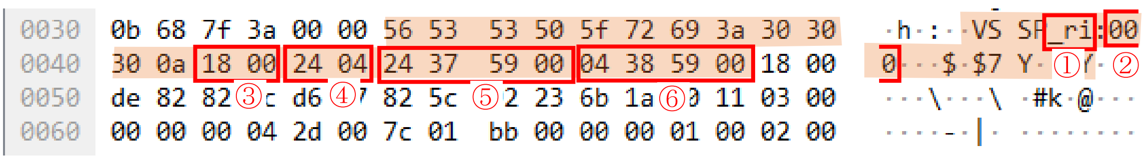
3) Echo index array

4) Measurement data array

All data from block (3) onward is binary. Multi-byte values are encoded in little endian.

7.1.1 VSSP common header

In the example communication log shown in the original document, the highlighted portion corresponds to the VSSP common header. The left column is a hexadecimal byte index, the middle column is binary data, and the right column is the ASCII representation.



In this example:

① packet type = "_ri"

② status = "000" (normal)

③ header bytes = 0x0018 = 24 bytes (20 bytes for this sensor)

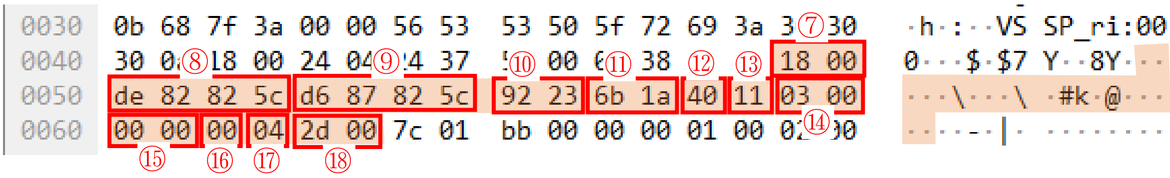
④ response bytes = 0x0424 = 1060 bytes

⑤ request timestamp = 0x00593724

⑥ response timestamp= 0x00593804

7.1.2 Distance measurement data header

In the example log, the distance measurement data header immediately follows the common header. It contains fields such as head timestamp, tail timestamp, head direction, tail direction, frame number, horizontal field number, line number, head spot number, vertical field number, and vertical interlace number.



In this example:

⑦ Header byte count 0x0018 = 24 (Note 1)

⑧ Timestamp of the first spot in the line 0x5c8282de

⑨ Timestamp of the last spot in the line 0x5c8287d6

⑩ Vertical direction of the first spot in the line 0x2392 = 9106 → 9106 × 360 / 65535 = 50.0° (Note2)

⑪ Vertical direction of the last spot in the line 0x1a6b = 6763 → 6763 × 360 / 65535 = 37.15° (Note2)

⑫ Frame number
0x40 = 64

⑬ Horizontal field number
0x11 = 17 (for this sensor it is fixed to 0)

⑭ Line number
0x0003 = 3 (for this sensor it is fixed to 1)

⑮ Head spot number of the line
0x0000 = 0

⑯ Vertical field number
0x00 = 0

⑰ Vertical interlace count
0x04 = 4 (maximum is 3 for this sensor)

⑱ Reserved

Notes

Note 1: When the horizontal interlace count is 1, the size is 20 bytes and items ⑯, ⑰, and ⑱ are omitted.

Note 2: Angles are defined with the front direction as 0°, and upward as the positive direction.

7.1.3 Echo index array

The echo index array follows the distance header. In the example log, the highlighted part shows the echo index array, whose structure is:

```
0030  0b 68 7f 3a 00 00 56 53  53 50 5f 72 69 3a 30 30   ·h·:··VS SP_ri:00
0040  30 0a 18 00 24 04 24 37  59 00 04 38 59 00 18 00   0···$·$7 Y··8Y···
0050  de 82 82 5c d6 87 82⑲c  9⑳23 6b㉑la 40 11 03 00   ···\···\ ·#k·@···
0060  00 00 00 04 2d 00 7c 01  bb 00 00 00 01 00 02 00   ····-·|· ········
0070  03 00 04 00 05 00 06 00  07 00 08 00 09 00 0a 00   ········ ········
                               ⋮
01d0  96 00 97 00 98 00 99 00  9a 00 9b 00 9c 00 9d 00   ········ ········
01e0  9e 00 6d 04 dc 01 6f 04  e0 01 70 04 e8 01 64㉒04   ··m···o· ··p···d·
01f0  e4㉓01 6c 04 e4 01 53 04  e8 01 4a 04 d4 01 6c 04   ··l···S· ··J···l·
0200  e4 01 60 04 e0 01 6a 04  dc 01 66 04 e0 01 6a 04   ··`···j· ··f···j·
```

⑲ echo index array bytes = 0x017C = 380

⑳ spot num = 0x00BB = 187

㉑ spot position[i] = first echo position for spot i in the expanded data array

㉒ all echo num = total number of echoes

㉓ reserved = padding so that the array length is aligned to 32-bit units

The spot position array is used to locate the echo data for each spot in the measurement data array. Depending on the number of spots, reserved bytes are used so that the echo index array is aligned to a 32-bit boundary.

7.1.4 Measurement data array and access procedure

For an _ri distance+intensity packet, the measurement data array contains alternating distance and intensity values. The echo index array and data array can be used together to access all echoes for any given spot. Consider the following example:

line = 1
head spot number = 0
index[] = {0, 1, 3, 5, 5, …}
data[]　= {100, 30, 150, 20, 180, 35, 102, 22, 103, 31, 111, 27, …}

For _ri data, multiplying each index element by 2 gives the start position of each spot's data in the measurement data array: {0, 2, 6, 10, 10, …}. (For _ro data, the index values are used directly without multiplying by 2.)

line1											
spot0		spot1				spot2				spot4	
echo1		echo1		echo2		echo1		echo2		echo1	
range	intensity	range	intensity	range	intensity	range	intensity	range	intensity	range	intensity
100	30	150	20	180	35	102	22	103	31	111	27

Thus, the data for each spot can be obtained as follows (C-like pseudocode):

```
/* Number of echoes in spot i */
echo_count = index[i + 1] - index[i];

/* If index[i] == index[i + 1], there is no echo for this spot. */

/* Access all echoes of spot i */
for (p = index[i], echo = 0; p < index[i + 1]; ++p, ++echo) {
    /* Distance */
    distance[i][echo]   = data[p * 2];
    /* Intensity */
    intensity[i][echo] = data[p * 2 + 1];
}
```

7.2 Three-Dimensional Transformations

Using the distance data contained in the packets and the coordinate conversion tables, the 3D coordinates of each measured point can be calculated.

In this document, a right-handed coordinate system is used in which the sensor's front, left, and up directions correspond to the positive X, Y, and Z axes respectively. Length units are omitted, and angles are expressed in radians.

7.2.1 Computation of horizontal and vertical angles for each spot

i : spot index

tblv[i] : value of the horizontal coordinate conversion table for spot i

(obtained with the GET command "tv**")

tblh[i] : value of the vertical coordinate conversion table for spot i

(obtained with the GET command "tblh")

φ_head : vertical angle of the first spot in the line (head direction)

φ_tail : vertical angle of the last spot in the line (tail direction)

[Horizontal angle θ_i]

$$\theta_i = tblv[i] \times \frac{2\pi}{65535}$$

The horizontal measurement range is common to all lines, so the horizontal coordinate conversion table tblv can be used to compute the horizontal angle for each spot.

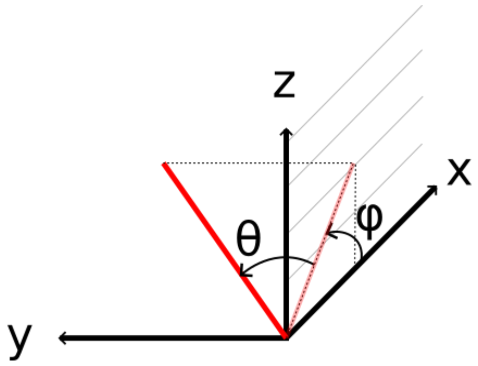
[Vertical angle φ_i]

$$\phi_i = \left(\phi_{head} + (\phi_{tail} - \phi_{head}) \times \frac{tblh[i]}{65535}\right) \times \frac{2\pi}{65535}$$

The vertical measurement range varies from line to line. Therefore, the vertical angle of each spot is calculated from the absolute angles of the first and last spots in the line (φ_head and φ_tail) and the vertical coordinate conversion table tblh, which represents the relative vertical position of each spot within the line.

7.2.2 Conversion to Cartesian coordinates

Each spot can be represented in a spherical coordinate system as shown in the figure in the original document.



Let r_i be the distance of spot i, φ_i its vertical angle, and θ_i its horizontal angle. The Cartesian coordinates (x_i, y_i, z_i) are given by:

$x_i = r_i \cdot \cos(\varphi_i) \cdot \cos(\theta_i)$

$y_i = r_i \cdot \cos(\varphi_i) \cdot \sin(\theta_i)$

$z_i = r_i \cdot \sin(\varphi_i)$