2019-1-24

Delta-1A communication interface protocol

[Delta-1A]

Shenzhen Shanchuan Robot Co., Ltd.

Table of contents

one. Introduction to Radar Communication

Delta-1A LiDAR communicates with external devices through UART TTL level and only supports simplex

Communication (that is, the lidar actively sends data frames to external devices), and the external devices only need to extract valid data from the data frames.

The data is enough, no response is required, all the data in the communication frame are data in hexadecimal format.

The radar rotates and measures for one week, and scans to obtain the information of evenly distributed points around it (the angle and distance of the points).

Leave). The sdk is to receive and analyze the data and get the information of each lap point. A circle of 360° is divided into 16 equal parts

The frame reports the scanning information (see the command word list below) frame, so the starting angles of each frame of 16 frames are obtained respectively.

Is 0° (zero point - see specification for position), 22.5°, 45°, 67.5°, 90°...270°, 292.5°, 315,

337.5°, 360°. 16 frames of data add up to a complete circle, the total number of points in one circle = 16 * the number of points per frame;

The total number of points in each frame can be obtained by calculating the number of distances according to the scanned information frame (number of distances = total number of points). per frame

Data point information (angle and distance): The distance of the Nth point in a frame is the N distance value in the scan information frame,

The angle corresponding to the distance of the Nth point in that frame = the starting angle of this frame + (N-1)*22.5/(the total of each frame

Points), so that a frame of point information (angle and distance) is available.

According to the communication protocol defined in this article, the communication data can be parsed, and the real-time measurement information and

Health status information.

two. Communication frame structure

The communication frame consists of frame header, frame length, frame type, command word, parameter length, parameter, and check code.

Mainly used for lidar to actively upload measurement information, fault information, etc. to the external host, the host only needs to

The effective data can be extracted from the communication frame uploaded by the radar, and no response is required.

The command frame format is as follows:

Frame Header Frame Length Protocol Version Frame Type Command Word Parameter Length Parameter Check Code								

Frame header: The frame header field occupies 1 Byte and is fixed at 0xAA.

Frame **length:** The frame length field occupies 2Bytes. The frame length is calculated from the frame header to the byte before the check code.

Highs come first, lows come after.

Protocol version: The address code field occupies 1Byte, and the default is 0x00.

Frame Type: The frame type field occupies 1 Byte and is fixed at 0x61.

Command word: The command word field occupies 1 Byte and is an identifier to distinguish different commands.

Parameter length: The parameter length occupies 2 Bytes, which is the length of the valid data in the data frame.

Parameters : The parameter field is the valid data for the command.

Check code: The check code field is the cumulative sum of 16 bits, occupying two bytes, with the high order in the front and the low order in the back.

Calculation: The accumulated sum from the beginning of the frame to the previous byte of the check code.

Command word list:

command word description		parameter length	Parameter Description
0xA9	Measurement information (3N+5)Bytes		0Bytes: radar speed value, 8 bits unsigned number, the minimum resolution is 0.05r/s (that is, the speed value is 1, and the corresponding speed is 0.05r/s) 1ÿ2Bytes: zero offset, 16 bits signed number, high bit first, low bit The minimum resolution is 0.01° (zero offset: radar debugging information **information, not used after parsing)** 3 ~ 4Bytes:

			The starting angle value of this data frame, 16 bits unsigned number, high-order first, low 5 Bytes: Signal value corresponding to distance value 1, 8 bits unsigned number (signal **Value: radar debugging information, not used after parsing)** 6ÿ7Bytes: Distance value 1,16 bits unsigned number, high order first, low order last 8Bytes: Signal value corresponding to distance value 2, 8 bits unsigned number (signal value: radar **Debug information, not used after parsing)** 9ÿ10Bytes: Distance value 2,16 bits unsigned number, high order first, low order last ..... 3N + 2Bytes: the signal value corresponding to the distance value N, 8 bits unsigned number **(Signal value: radar debugging information, not used after analysis)** 3N + 3 ~ 3N + 4Bytes: Distance value N, 16 bits unsigned number, high order first, low order last Remark:

			1. Angle value range: 0 ~ 36000 2. Angular resolution: 0.01° (that is, the angle value is 1, and the corresponding angle is 0.01°) The distance resolution is 0.25mm (that is, the distance value is 1, and the corresponding actual distance is 0.25mm) 3. Angle calculation: Example: Distance n (n is 1 ~N, N is the distance points in this frame) corresponding angle calculation: N = (parameter length - 5)/3 Angle from distance n = starting angle value + 22.5°*(n - 1)/N
0xAE Device	Health Letter interest	1Byte	Equipment speed failure Speed value, 8 bits unsigned number, the minimum resolution is 0.05r/s

three. Check code calculation

The communication frame check algorithm of this protocol adopts 16-bit cumulative sum. The following is the routine for calculating the check code. for reference.

```
//================================================================================================
// checksum calculation
// *Start_Byte: start byte
```

```
// Num_Bytes: the length of the calculated data

// Return value: 16-bit check code

//=============================================
====================================

u16 CRC16(u8 *Start_Byte, u16 Num_Bytes)
{
    u16 Checksum = 0;
    while (Num_Bytes--)
    { // Calculate CRC
        Checksum += *Start_Byte++;
    }
    return Checksum;
}
```

Four. Communication frame instance analysis

0. Resolution in the protocol: actual measurement data = value in communication * resolution

Actual speed = speed value in communication * resolution (0.05r/s)

Actual distance = distance value in communication * resolution (0.25mm)

Actual angle = angle value in communication * resolution (0.01°)

1. Measure the data frame:

AA 00 4F 00 61 AD 00 47 79 00 40 72 42 3C 05 6D 37 05 8A 3A 05 93

34 05 9C 35 05 AD 35 05 B8 35 05 C6 3505 D5 34 05 E5 36 05 F2 31

06 07 2D 06 16 2E 06 2B 2E 06 40 36 06 52 35 06 67 32 06 61 2D 06

45 2B 06 222B 06 03 31 05 DF 30 05 C6 13 3B

AA: frame header

00 4F: The frame length is 0x004F (note: only the frame length of the instance frame, not the actual length of the radar)

00: Protocol version

61: Frame Type

AD: command word

00 47: Valid data length 0x0047

79: Radar speed 121*0.05r/s (resolution)=6.05r/s

00 40: Zero offset 64*0.01° (resolution) = 0.64°

72 42: starting angle 29250*0.01°(resolution)=292.5°

3C: Signal letter 1

05 6D: Distance value 1 is 1389*0.25mm (resolution)=347.25mm

37: Signal value 2

05 8A: Distance value 2 is 1418*0.25mm (resolution)=354.5mm........

30: Signal value 22

05 C6: Distance value 22 is 1478*0.25mm (resolution)=369.5mm

13 3B: Check code 0x133B= (AA+00+4F+00+…+DF+30+05+C6)

2. Radar RPM fault frame:

AA 00 09 00 61 AE 00 01 69 02 2C

AA: Frame header identification.

00 09: The frame length is 0x0009 (ie 9) bytes (excluding CRC code)

00: Protocol version

61: Frame Type

AE: command word

00 01: Valid data length 0x0001

C9: radar speed 0xC9, that is, 201*0.05r/s (resolution) = 10.05r/s

02 2C: Check code 0x022c