Contents

Overview 3
Note 4
Specification 5
Pin Interface Definition 6
Single Module Test 7
  Preparation Before Test 7
  Connect to Radar 7
  Data Real-time Display 8
Gimbal Control Module 10
Data Protocol 11
  Radar Serial Port Output Protocol 11
  Radar Output CRC8 Calibration 12
  77G Radar Gimbal Protocol 13
System Obstacle Avoidance Solution 16
  Open Source Flight Control Platform 16
FAQ 21
Overview

The 77GHz millimeter-wave obstacle avoidance radar is especially designed for industrial UAV. The obstacle avoidance radar can be easily integrated with existing platforms, like PIXHAWK, DJI A3,N3 etc. It is a single axis obstacle avoidance radar system, which is able to maintain a self-stabilizing state during flight, and always detecting forward, can not be affected by the flight action of the drone.
Note

(1) The power supply pin needs to be externally connected to 5VDC;
(2) The front of the module is consistent with the direction of the drone head when installing, and there is no obstruction in front of the module;
(3) Pay attention to the installation mode of the gimbal, the outgoing line side is the front side.
## Specification

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transmission Frequency</td>
<td>76-77GHz</td>
</tr>
<tr>
<td>Detection Range</td>
<td>Greater Than 100m</td>
</tr>
<tr>
<td>Detection Accuracy</td>
<td>± 0.18m</td>
</tr>
<tr>
<td>Wave Speed Width</td>
<td>110°(Yaw) and 15.6°(pitch)</td>
</tr>
<tr>
<td>Interface</td>
<td>UART</td>
</tr>
<tr>
<td>Data Output Frequency</td>
<td>50Hz</td>
</tr>
<tr>
<td>Voltage</td>
<td>5V(DC)</td>
</tr>
<tr>
<td>Gimbal Angle Range</td>
<td>180°</td>
</tr>
<tr>
<td>Working Temperature</td>
<td>-20°C</td>
</tr>
<tr>
<td>Weight</td>
<td>120g</td>
</tr>
<tr>
<td>Size</td>
<td>70x102x34mm</td>
</tr>
</tbody>
</table>
Pin Interface Definition

<table>
<thead>
<tr>
<th>Pin Interface Definition</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>VCC(Red)</td>
<td>5V DC</td>
<td>-</td>
</tr>
<tr>
<td>GND(Black)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>UART_RX(Yellow)</td>
<td>TTL 3.3 DC</td>
<td>Radar Port</td>
</tr>
<tr>
<td>UART_TX(Green)</td>
<td>TTL 3.3 DC</td>
<td>Radar Port</td>
</tr>
<tr>
<td>UART_RX(White)</td>
<td>TTL 3.3 DC</td>
<td>Gimbal Port</td>
</tr>
<tr>
<td>UART_TX(Blue)</td>
<td>TTL 3.3 DC</td>
<td>Gimbal Port</td>
</tr>
</tbody>
</table>

Adjust the self-stabilizing angle through the serial port.
Single Module Test

Preparation Before Test

Use USB to serial port device to connect the radar output serial port, the USB port is connected to the PC serial port assistant debugging software, you can see the output data, or observe the radar output data more intuitively through the company’s developed software “UAV radar obstacle avoidance expert system”. Please refer to the serial data protocol description for specific data. The testing using tools or software is shown in the following table:

<table>
<thead>
<tr>
<th>No.</th>
<th>Equipent</th>
<th>Quant.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>77G Radar</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>PC</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>USB to TTL Adapter</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>5V Power Adapter</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>Serial port debugging software/radar obstacle avoidance expert system.</td>
<td>1</td>
</tr>
</tbody>
</table>

Please connect the TX of the adapter to the RX of the radar, and the RX to the TX.

Connect to Radar

Connect the radar serial port to PC according to the connecting method above, open the "radar obstacle avoidance expert system", and you will see the radar output data in the sector area. The interface is as below:
The default connecting method of this software is COM. After connect the radar to PC, open the software and it will automatically identify the port, chose the radar type, click “save the setting”.

Data Real-time Display

The real-time data display module is based on the successful connection of the device, and displays the real-time received radar data in dynamic line graph and sector orientation:
- In the above figure 1, the radar altimeter, barometer and other data are displayed in real time in a dynamic line diagram;
- In the above figure 2, the sector data of the obstacle avoidance radar is displayed in real time in the sector orientation map, and the obstacle avoidance display range can be dynamically adjusted in the above figure 3, the range is 0-120m, the default is 30m, and according to the return The obstacle distance is displayed in an arc of a different color in the corresponding sector, and the distance of the obstacle is displayed under the corresponding sector;
- According to the data algorithm returned by the radar, you can freely check the CRC8 algorithm on the left side of Figure 3. The default is not checked. After checking, the check value will be calculated by the custom CRC8 algorithm, which is mainly used for parameter setting.
Gimbal Control Module

The gimbal control module is based on the successful connection of the radar. It is necessary to connect the gimbal serial port to the PC and control the gimbal to achieve stable radar obstacle avoidance reliability. The operation commands are as shown in the red box below:

- For the forward and reverse of the actual installation of the gimbal, the default is to check the “Forward” checkbox, that is, the gimbal installation is its own forward direction, otherwise the installation is reversed, then the “Forward” checkbox is unchecked;
- The gimbal control mode is divided into fixed mode and self-stabilized mode. In each mode, there are parameter setting and serial port control. The system can read the current gimbal control mode in real time, and can be freely set according to requirements.
- The system can read the gimbal angle in real time. This angle distinguishes different modes. The fixed mode is the gimbal fixed angle, the self-stabilizing mode is the gimbal stable angle. The angle range is -90 degrees to 90 degrees. At the same time, you can freely drag the arrow position progress bar or change the text value to change the angle.

💡 The Radar has default parameters, which generally do not need to be changed. If necessary, it can be modified according to the 77G radar head protocol.
Data Protocol

The 77G radar is a dual serial port output, one outputs radar data and the other is a gimbal output serial port. The serial port settings are 115200bps, 8N1.

Radar Serial Port Output Protocol

77G Radar serial port output frequency is 50Hz, the specific protocol format is as follows:
head byte D1 D2 D3 D4 D5 D6 D7 D8 CRC8

<table>
<thead>
<tr>
<th>Byte</th>
<th>Parametric Description</th>
<th>Type</th>
<th>Unit</th>
<th>Explanation</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Lead Byte 1</td>
<td>uint8_t</td>
<td></td>
<td>Fixed as ‘T’, namely 0x54</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Lead Byte 2</td>
<td>uint8_t</td>
<td></td>
<td>Fixed as ‘H’, namely 0x48</td>
<td></td>
</tr>
<tr>
<td>2-3</td>
<td>D1</td>
<td>uint16_t</td>
<td>cm</td>
<td>0 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>4-5</td>
<td>D2</td>
<td>uint16_t</td>
<td>cm</td>
<td>45 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>6-7</td>
<td>D3</td>
<td>uint16_t</td>
<td>cm</td>
<td>90 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>8-9</td>
<td>D4</td>
<td>uint16_t</td>
<td>cm</td>
<td>0135 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>10-11</td>
<td>D5</td>
<td>uint16_t</td>
<td>cm</td>
<td>180 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>12-15</td>
<td>D6</td>
<td>uint16_t</td>
<td>cm</td>
<td>225 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>14-15</td>
<td>D7</td>
<td>uint16_t</td>
<td>cm</td>
<td>270 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>16-17</td>
<td>D8</td>
<td>uint16_t</td>
<td>cm</td>
<td>315 degree sector obstacle distance</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>CRC8</td>
<td>uint8_t</td>
<td></td>
<td>CRC8 verification</td>
<td>See below description</td>
</tr>
</tbody>
</table>

Obstacle distance: unit: cm; The high 8 bits are in front and the low 8 bits are behind, such as 0-degree sector obstacles distance 0x07D0, byte 2=0x07, byte 3=0xD0, the actual distance is 20m.

⚠️ Send data no matter if it has radar data or not. When the data is invalid, DX fills in 0xFFFF. 77G radar obstacle avoidance system output D1, D2, D8 sector obstacle distance, other sectors are invalid data, filled with 0xFFFF.
Crc.cpp
static const uint8_t crc8_table[] = {
  0x00, 0x07, 0x0e, 0x09, 0x1c, 0x1b, 0x12, 0x15, 0x38, 0x3f, 0x36, 0x31,
  0x24, 0x23, 0x2a, 0x2d, 0x70, 0x77, 0x7e, 0x79, 0x6c, 0x6b, 0x62, 0x65,
  0x48, 0x4f, 0x46, 0x41, 0x54, 0x55, 0x5a, 0x5d, 0xe0, 0xe7, 0xe2, 0xe9,
  0xfc, 0xfb, 0xf2, 0xf5, 0xd8, 0xdf, 0xd6, 0xd1, 0xc4, 0xc3, 0xca, 0xcd,
  0xc0, 0x97, 0x9e, 0x99, 0x8c, 0x8b, 0x82, 0x85, 0xa8, 0xaf, 0xa6, 0xa1,
  0xb4, 0xb3, 0xba, 0xbd, 0xc7, 0xc0, 0xc9, 0xce, 0xdb, 0xdc, 0xd5, 0xd2,
  0xff, 0xf8, 0xf1, 0xf6, 0xe3, 0xe4, 0xed, 0xea, 0xb7, 0xb0, 0xb9, 0xbe,
  0xab, 0xac, 0xa5, 0xa2, 0x8f, 0x88, 0x81, 0x86, 0x93, 0x94, 0x9d, 0x9a,
  0x27, 0x20, 0x29, 0x2e, 0x3b, 0x3c, 0x35, 0x32, 0x1f, 0x18, 0x11, 0x16,
  0x03, 0x04, 0x0d, 0x0a, 0x57, 0x50, 0x59, 0x5e, 0x4b, 0x4c, 0x45, 0x42,
  0x6f, 0x68, 0x61, 0x66, 0x73, 0x74, 0x7d, 0x7a, 0x89, 0x8e, 0x87, 0x80,
  0x95, 0x92, 0x9b, 0x9c, 0xb1, 0xb6, 0xbf, 0xb8, 0xad, 0xaa, 0xa3, 0xa4,
  0x9f, 0xfe, 0xf7, 0xf0, 0xe5, 0xe2, 0xeb, 0xec, 0xc1, 0xc6, 0xcf, 0xc8,
  0xdd, 0xda, 0xd3, 0xd4, 0x69, 0x6e, 0x67, 0x60, 0x75, 0x72, 0x7b, 0x7c,
  0x51, 0x56, 0x5f, 0x58, 0x4d, 0x4a, 0x43, 0x44, 0x19, 0x1e, 0x17, 0x10,
  0x05, 0x02, 0x0b, 0x0c, 0x21, 0x26, 0x2f, 0x28, 0x3d, 0x3a, 0x33, 0x34,
  0x4e, 0x49, 0x40, 0x47, 0x52, 0x55, 0x5c, 0x5b, 0x56, 0x77, 0x71, 0x78,
  0x7f, 0x6a, 0x6d, 0x64, 0x63, 0x3e, 0x39, 0x30, 0x37, 0x22, 0x25, 0x2c, 0x2b,
  0x06, 0x01, 0x08, 0x0f, 0x1a, 0x1d, 0x14, 0x13, 0xae, 0xa9, 0xa0, 0xa7,
  0xb2, 0xb5, 0xbc, 0xbb, 0x96, 0x91, 0x98, 0x9f, 0x8a, 0x8d, 0x84, 0x83,
  0xde, 0xd9, 0xd0, 0xd7, 0xc2, 0xc5, 0xcc, 0xcb, 0xe6, 0xe1, 0xe8, 0xef,
  0xfa, 0xfd, 0xf4, 0xf3
};

uint8_t crc_crc8(const uint8_t *p, uint8_t len)
{
  uint16_t i;
  uint16_t crc = 0x0;
  while (len--)
  {
    i = (crc ^ *p++) & 0xFF;
    crc = (crc8_table[i] ^ (crc << 8)) & 0xFF;
  }
  return crc & 0xFF;
}
77G Radar Gimbal Protocol

Configuring Gimbal Parameters

gimbal parameter protocol setting by the PC-side serial port is shown in the following table. A single instruction has no return value, and the status is continuously sent by the gimbal.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Parametric Description</th>
<th>Type</th>
<th>Range</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>First Byte</td>
<td>uint8_t</td>
<td>0x48</td>
<td>Fixed as ‘K’, namely 0x48</td>
</tr>
</tbody>
</table>
| 1    | mounting type         | uint8_t | 0~4, default 0x00 | 0-Installation direction by parameter settings  
1-forward installation;  
2-reverse forward-installation;  
3-forward downward-installation;  
4-Reverse downward-installation; |
| 2    | gimbal control mode   | uint8_t | 0~3, default 0x02 | 0-fixed mode, the fixed angle of the gimbal is controlled by parameters;  
1-fixed mode, the fixed angle of the gimbal is controlled by the serial port;  
2-Self-stabilizing mode, the stability angle of the gimbal is set by parameters;  
3-Self-stabilizing mode, the stability angle of the gimbal is controlled by the serial port; |
| 3~4  | gimbal angle           | Short | -900~900, default 0 | -90 degrees to 90 degrees, the unit is 0.1 degrees, the upper 8 bytes are in the front and the lower 8 bytes are in the back;  
-If the gimbal control mode is 1 fixed mode, the angle is a fixed angle of the gimbal;  
-If the gimbal control mode is 3 self-stabilizing mode, the angle is the gimbal stabilizing angle;  
-Other gimbal control mode, this angle is invalid |
| 5    | CRC8 verification      | uint8_t |     | From the installation mode to the gimbal stable angle byte CRC8 check.  
The verification algorithm is described below. |
### Gimbal Delivery Status

The gimbal delivery status protocol is as shown in the following table. The output frequency is 100Hz.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Parametric Description</th>
<th>Type</th>
<th>Range</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>First Byte</td>
<td>uint8_t</td>
<td>0x48</td>
<td>Fixed as 'K', namely 0x48</td>
</tr>
<tr>
<td>1</td>
<td>mounting type</td>
<td>uint8_t</td>
<td>0–48</td>
<td>0-installation direction by parameter settings</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1-forward installation;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2-reverse forward-installation;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3-forward downward-installation;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>4-Reverse downward-installation;</td>
</tr>
<tr>
<td>2</td>
<td>gimbal control mode</td>
<td>uint8_t</td>
<td>1–4</td>
<td>0-fixed mode, the fixed angle of the gimbal is controlled by parameters;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1-fixed mode, the fixed angle of the gimbal is controlled by the serial port;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2-Self-stabilizing mode, the stability angle of the gimbal is set by parameters;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3-Self-stabilizing mode, the stability angle of the gimbal is controlled by the serial port;</td>
</tr>
<tr>
<td>3–4</td>
<td>gimbal current attitude angle</td>
<td>Short</td>
<td>0–3</td>
<td>-90 degrees to 90 degrees, the unit is 0.1 degrees, the high 8-bit bytes are in the front and the low 8 bytes are in the back;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-if the gimbal control mode is 1 fixed mode, the angle is a fixed angle of the gimbal;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-if the gimbal control mode is 3 self-stabilizing mode, the angle is the gimbal stabilizing angle;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-Other gimbal control mode, this angle is invalid.</td>
</tr>
<tr>
<td>5–6</td>
<td>gimbal control angle</td>
<td>Short</td>
<td>-900–900</td>
<td>-90 degrees to 90 degrees, the unit is 0.1 degrees, the high 8-bit bytes are in the front and the low 8 bytes are in the back;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-if the gimbal control mode is 1 fixed mode, the angle is a fixed angle of the gimbal;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-if the gimbal control mode is 3 self-stabilizing mode, the angle is the gimbal stabilizing angle;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-Other gimbal control mode, this angle is invalid.</td>
</tr>
<tr>
<td>7–8</td>
<td>current motor control PWM value</td>
<td>uint16_t</td>
<td>500–2500</td>
<td>The high 8-bit bytes are in the front and the low 8 bytes are in the back;</td>
</tr>
<tr>
<td>9</td>
<td>CRC8 verification</td>
<td>uint8_t</td>
<td></td>
<td>From the installation mode to current motor control PWM value byte CRC8 check,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>The verification algorithm is described below.</td>
</tr>
</tbody>
</table>
PTZ CRC8 Check

PTZ CRC8 check procedure.

Crc.cpp

static const uint8_t crc8_table[] = {
    0x39, 0x61, 0x58, 0x09, 0x1c, 0x1b, 0x12, 0x15, 0x38, 0x3f, 0x36, 0x31,
    0x24, 0x23, 0x2a, 0x2d, 0x70, 0x77, 0x7e, 0x79, 0x6c, 0x6b, 0x62, 0x65,
    0x68, 0x4f, 0x46, 0x41, 0x54, 0x5a, 0x5d, 0xe0, 0xe7, 0xee, 0xe9, 0xfc,
    0xfb, 0xf2, 0xf5, 0xd8, 0xdf, 0xd6, 0xd1, 0xc4, 0xc3, 0xca, 0xcd,
    0x90, 0x97, 0x9e, 0x99, 0x8c, 0x8b, 0x82, 0x85, 0xa8, 0xaf, 0xa6, 0xa1,
    0xb4, 0xb3, 0xba, 0xbc, 0xc7, 0xc0, 0xc9, 0xce, 0xdb, 0xdc, 0xd5, 0xd2,
    0xff, 0xf8, 0xf1, 0xf6, 0xe3, 0xe4, 0xed, 0xea, 0xb7, 0xb0, 0xb9, 0xbe,
    0xab, 0xac, 0xa5, 0xa2, 0x8f, 0x88, 0x81, 0x86, 0x93, 0x94, 0x9d, 0x9a,
    0x27, 0x20, 0x29, 0x2e, 0x3b, 0x3c, 0x35, 0x32, 0x1f, 0x18, 0x11, 0x16,
    0x03, 0x04, 0x0d, 0xa0, 0x57, 0x50, 0x59, 0x5e, 0x4b, 0x4c, 0x45, 0x42,
    0x6f, 0x68, 0x61, 0x66, 0x73, 0x7e, 0x74, 0x7d, 0x7a, 0x89, 0xe8, 0x87,
    0x80, 0x95, 0x92, 0x9c, 0xb1, 0xb6, 0xbf, 0xb8, 0xad, 0xaa, 0xa3, 0xa4,
    0xf9, 0xfe, 0xf7, 0xf0, 0xe5, 0xe2, 0xeb, 0xe8, 0xc1, 0xc6, 0xc6, 0xc8,
    0xdd, 0xda, 0xd3, 0xd4, 0x69, 0x66, 0x67, 0x60, 0x75, 0x72, 0x7b, 0x7c,
    0x51, 0x56, 0x5f, 0x58, 0x5d, 0x4a, 0x43, 0x44, 0x19, 0x1e, 0x17, 0x10,
    0x05, 0x02, 0x0b, 0x0c, 0x21, 0x26, 0x2f, 0x28, 0x3d, 0x3a, 0x33, 0x34,
    0x4e, 0x49, 0x40, 0x47, 0x52, 0x55, 0x5c, 0x5b, 0x76, 0x71, 0x78, 0x71,
    0x6a, 0x6d, 0x64, 0x63, 0x3e, 0x39, 0x30, 0x37, 0x22, 0x25, 0x2c, 0x2b,
    0x7e, 0x7f, 0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77, 0x78, 0x79,
    0x7a, 0x7b, 0x7c, 0x7d, 0x7e, 0x7f, 0x80, 0x81, 0x82, 0x83, 0x84, 0x85,
    0x86, 0x87, 0x88, 0x89, 0x8a, 0x8b, 0x8c, 0x8d, 0x8e, 0x8f, 0x90, 0x91,
    0x92, 0x93, 0x94, 0x95, 0x96, 0x97, 0x98, 0x99, 0x9a, 0x9b, 0x9c, 0x9d,
    0x9e, 0x9f, 0xa0, 0xa1, 0xa2, 0xa3, 0xa4, 0xa5, 0xa6, 0xa7, 0xa8, 0xa9,
    0xaa, 0xab, 0xac, 0xad, 0xae, 0xaf, 0xb0, 0xb1, 0xb2, 0xb3, 0xb4,
    0xb5, 0xb6, 0xb7, 0xb8, 0xb9, 0xba, 0xbb, 0xbc, 0xbd, 0xbe, 0xbf, 0xc0,
    0xc1, 0xc2, 0xc3, 0xc4, 0xc5, 0xc6, 0xc7, 0xc8, 0xc9, 0xca, 0xcb, 0xcc,
    0xd0, 0xd1, 0xd2, 0xd3, 0xd4, 0xd5, 0xd6, 0xd7, 0xd8, 0xd9, 0xda, 0xdb,
    0xdc, 0xdd, 0xde, 0xdf, 0xe0, 0xe1, 0xe2, 0xe3, 0xe4, 0xe5, 0xe6, 0xe7,
    0xe8, 0xe9, 0xea, 0xeb, 0xec, 0xed, 0xee, 0xef, 0xf0, 0xf1, 0xf2, 0xf3
};
/*
crc8 from trone driver by Luis Rodrigues
*/
uint8_t crc_crc8(const uint8_t *p, uint8_t len)
{
    uint16_t crc = 0x0030;
    while (len--)
    {
        i = (crc ^ *p++) & 0xFF;
        crc = (crc8_table[i] ^ (crc << 8)) & 0xFF;
    }
    return crc & 0xFF;
}
System Obstacle Avoidance Solution

In order to facilitate the faster and better integrated use with the radar module, we have proposed the following system solutions for the mainstream flight control platforms currently on the market.

Open Source Flight Control Platform

The radar is compatible with the open source flight control obstacle avoidance protocol and can be directly connected to the general open source flight control platform. The following is a brief description of the integrated application settings of this radar on the APM flight control platform.

Flight Control Hardware: PixhawkV3
Flight Control Software: ArduPilot Copter 3.5.5
Ground station software: MissionPlanner 1.3.62

Radar Installation and Wiring

The radar serial port is connected to the pixhawk TELEM2 interface. The radar needs to be powered separately. The interface definition is as shown below:

<table>
<thead>
<tr>
<th>Pin#</th>
<th>Name</th>
<th>DIR</th>
<th>Wire Color</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VCC_5V</td>
<td>out</td>
<td>red/gray</td>
<td>Supply to GPS from AP</td>
</tr>
<tr>
<td>2</td>
<td>MCU_TX</td>
<td>out</td>
<td>yellow/black</td>
<td>3.3V-5.0V TTL Level, TX of AP</td>
</tr>
<tr>
<td>3</td>
<td>MCU_RX</td>
<td>in</td>
<td>green/black</td>
<td>3.3V-5.0V TTL Level, RX of AP</td>
</tr>
<tr>
<td>4</td>
<td>MCU_CTS(TX)</td>
<td>out</td>
<td>gray/black</td>
<td>3.3V-5.0V TTL Level, TX of AP</td>
</tr>
<tr>
<td>5</td>
<td>MCU_RTS(RX)</td>
<td>in</td>
<td>gray/black</td>
<td>3.3V-5.0V TTL Level, RX of AP</td>
</tr>
<tr>
<td>6</td>
<td>GND</td>
<td>-</td>
<td>black</td>
<td>GND Connection</td>
</tr>
</tbody>
</table>
The installation method can refer to the following figure:

Flight Control Source Code Modification

The obstacle avoidance radar uses the Lidar360 protocol. Because the maximum measurement distance is limited, you need to modify the "AP_Proximity_TeraRangerTower.cpp" file in the AP_Proximity library in the libraries directory.
As shown below:
(1) Modify the maximum measurement distance to 100m
(2) Modify the measured data unit to cm.

MissionPlanner Ground Station Flight Control Parameter Settings

(1) Set the TELEM2 serial port parameter, the SERIAL2 baud rate is set to 115200bit/s (SERIAL2_BAUD is set to 115) and the communication protocol is set to Lidar360 (SERIAL2_PROTOCOL is set to 11) as shown below:
(2) Set the obstacle avoidance sensor protocol to TeraRangerTower (PRX_TYPE is set to 3), as shown below:

(3) Set the obstacle avoidance type AVOID_ENABLE to UseProximitySensor, the obstacle avoidance maximum tilt angle is 10 degrees (AVOID_ANGLE_MAX is set to 1000), the obstacle avoidance action mode is stop (AVOID_BEHAVE is set to 1), and the obstacle avoidance distance in gps mode is 3m (AVOID_MARGIN is set to 3), the obstacle avoidance distance in fixed height mode is 10m (AVOID_DIST_MAX is set to 10), as shown below:
(4) After setting the above parameters and saving, then restart flight control, connect MissionPlanner, CTRL+F to open the debugging window to see the obstacle avoidance distance, as shown below: (target is in 0 sector 3.4m)

After the above settings are completed, then can perform the outdoor test. When the drone is less than 3m away from the obstacle in gps mode (less than 10m in the fixed height mode), the drone will have a brake action, and the joystick cannot make the drone continue to fly forward.
FAQ

Q: Can 77G radar only avoid obstacles in one direction?

A: In the open source flight control platform, this 77G radar currently only supports obstacle avoidance in one direction (forward obstacle avoidance); in the DJI flight control platform, this 77G radar can output data of multiple sectors, and subsequently cooperate with Daxiang Zhiyin, that support obstacle avoidance in multi-direction, and it coming soon.

This content is subject to change.

Download the latest version from
https://www.foxtechfpv.com/77ghz-millimeter-wave-obstacle-avoidance-radar.html

For everyday updates, please follow
Foxtech Facebook page: https://www.facebook.com/foxtechhobby
YouTube Channel: https://www.youtube.com/user/foxtechonline/featured?view_as=subscriber