通信接口	UART
默认波特率	115200
数据位	8
停止位	1
奇偶校验	None

标准数据输出格式（默认）:

数据结构：数据格式为 16 进制（HEX）。具体数据编码详见表

Byte0-1	Byte2	ByteType	ByteData	ByteType	ByteData	……	Bytelast
0x49 0x39	Type_Num	Type	Date	Type	Data	……	0x55 0x39

注：杭州电子科技大学 hzdzkjdx 九键为 49395539

数据编码解释	
Byte0	0x49，每一帧都相同
Byte1	0x39，每一帧都相同
Byte2	Type_Num 帧状态，前 4 位表示数据种类，后 4 位为数据个数，例：0x15 表示数据类型为①，发送五个数据 数据种类分四类：①int16_t②int32_t③uint16_t④uint32_t
ByteType	数据类型，八位
ByteData	Data,若数据类型为 1、3 则为 16 位，若数据类型为 2、4 则为 32 位
Bytelast2	0x55，每一帧都相同
Bytelast	0x39，每一帧都相同

/**********connect.h**************************/

#ifndef _connect_h

```
#define _connect_h
#include "include.h"
typedef struct
{
	uint8_t Type;
	uint8_t Date_1;
	uint8_t Date_2;
	uint8_t Date_3;
	uint8_t Date_4;
}Connect_TypedefU;
typedef struct
{
	uint8_t Type;
	int8_t Date_1;
	int8_t Date_2;
	int8_t Date_3;
	int8_t Date_4;
}Connect_Typedef;
enum
{
	SVEL=1,
	SREMOTE,
	SLAS,
	SSTATE
};
extern Connect_TypedefU unsigned_connet;
extern Connect_Typedef signed_connet;
extern uint8_t SEND_STA;
void Set_TypeU(Connect_TypedefU *co,uint8_t type);
void Set_Type(Connect_Typedef *co,uint8_t type);
void Set_Datau32(Connect_TypedefU *co,uint32_t data);
void Set_Datau16(Connect_TypedefU *co,uint16_t data);
void Set_Data32(Connect_Typedef *co,int32_t data);
void Set_Data16(Connect_Typedef *co,int16_t data);
void Send_Begin(uint8_t STA);
void Send_End(void);
void Send_MessageU16(Connect_TypedefU *co);
void Send_Message16(Connect_Typedef *co);
void Send_MessageU32(Connect_TypedefU *co);
void Send_Message32(Connect_Typedef *co);
void publish_vel(void);
void publish_remote(void);
void publish_las(void);
void publish_state(void);
```

```
#define MSG_vel_x              0X01
#define MSG_vel_y              0X02
#define MSG_vel_z              0X03
#define MSG_distance_f      0X04          //前距离
#define MSG_distance_b      0X05          //后距离
#define MSG_distance_l      0X06          //左距离
#define MSG_distance_r      0X07          //右距离
#define MSG_remote_ch0      0x08     //遥控器 ch0
#define MSG_remote_ch1      0x08     //遥控器 ch1
#define MSG_remote_ch2      0x09     //遥控器 ch2
#define MSG_remote_ch3      0x0A     //遥控器 ch3
#define MSG_remote_s1       0x0B     //遥控器 s1
#define MSG_remote_s2       0x0C     //遥控器 s2
#endif
/*****************************************************************************/


/***************connect.c*****************************************/
#include "connect.h"
Connect_TypedefU unsigned_connet;
Connect_Typedef signed_connet;
uint8_t SEND_STA;
void Set_TypeU(Connect_TypedefU *co,uint8_t type)
{
    co->Type=type;
}
void Set_Type(Connect_Typedef *co,uint8_t type)
{
    co->Type=type;
}

void Set_Datau32(Connect_TypedefU *co,uint32_t data)
{
    uint8_t data1=(uint8_t)data&0x000000FF;
    uint8_t data2=(uint8_t)(data>>8)&0x000000FF;
    uint8_t data3=(uint8_t)(data>>16)&0x000000FF;
    uint8_t data4=(uint8_t)(data>>24)&0x000000FF;

    co->Date_1=data1;
    co->Date_2=data2;
    co->Date_3=data3;
    co->Date_4=data4;
}
void Set_Datau16(Connect_TypedefU *co,uint16_t data)
```

```
{
	uint8_t data1=(uint8_t)data&0x00FF;
	uint8_t data2=(uint8_t)(data>>8)&0x00FF;
	co->Date_1=data1;
	co->Date_2=data2;
}
void Set_Data32(Connect_Typedef *co,int32_t data)
{
	int8_t data1=(int8_t)data&0x000000FF;
	int8_t data2=(int8_t)(data>>8)&0x000000FF;
	int8_t data3=(int8_t)(data>>16)&0x000000FF;
	int8_t data4=(int8_t)(data>>24)&0x000000FF;

	co->Date_1=data1;
	co->Date_2=data2;
	co->Date_3=data3;
	co->Date_4=data4;
}

void Set_Data16(Connect_Typedef *co,int16_t data)
{
	int8_t data1=(int8_t)data&0x00FF;
	int8_t data2=(int8_t)(data>>8)&0x00FF;
	co->Date_1=data1;
	co->Date_2=data2;
}
void Send_MessageU16(Connect_TypedefU *co)
{
	usart1_write(co->Type);
	usart1_write(co->Date_1);
	usart1_write(co->Date_2);
}
void Send_MessageU32(Connect_TypedefU *co)
{
	usart1_write(co->Type);
	usart1_write(co->Date_1);
	usart1_write(co->Date_2);
	usart1_write(co->Date_3);
	usart1_write(co->Date_4);
}
void Send_Message16(Connect_Typedef *co)
{
	usart1_write(co->Type);
	usart1_write(co->Date_1);
```

```
	usart1_write(co->Date_2);
}
void Send_Message32(Connect_Typedef *co)
{
	usart1_write(co->Type);
	usart1_write(co->Date_1);
	usart1_write(co->Date_2);
	usart1_write(co->Date_3);
	usart1_write(co->Date_4);
}
void Send_Begin(uint8_t STA)
{
	usart1_write(0x49);
	usart1_write(0x39);
	switch(STA)
	{
		case SVEL:
			usart1_write(0x13);SEND_STA=True;
			break;
		case SREMOTE:
			usart1_write(0x36);SEND_STA=True;
			break;
		case SLAS:
			usart1_write(0x34);SEND_STA=True;
			break;
		case SSTATE:
			usart1_write(0x31);SEND_STA=True;
			break;
		default:
			SEND_STA=False;break;
	}

}
void Send_End(void)
{
	usart1_write(0x55);
	usart1_write(0x39);
	SEND_STA=False;
}
void publish_vel(void)
{
	Send_Begin(SVEL);
	if(SEND_STA)
	{
```

```
            Set_Type(&signed_connet,MSG_vel_x);
            Set_Data16(&signed_connet,(int16_t)required_linear_vel_x*100);
            Send_Message16(&signed_connet);
            Set_Type(&signed_connet,MSG_vel_y);
            Set_Data16(&signed_connet,(int16_t)required_linear_vel_y*100);
            Send_Message16(&signed_connet);
            Set_Type(&signed_connet,MSG_vel_z);
            Set_Data16(&signed_connet,(int16_t)required_angular_vel*100);
            Send_Message16(&signed_connet);
        }
        Send_End();
}

void publish_remote(void)
{
        Send_Begin(SREMOTE);
        if(SEND_STA)
        {
            Set_TypeU(&unsigned_connet,MSG_remote_ch0);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.ch0);
            Send_MessageU16(&unsigned_connet);
            Set_TypeU(&unsigned_connet,MSG_remote_ch1);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.ch1);
            Send_MessageU16(&unsigned_connet);
            Set_TypeU(&unsigned_connet,MSG_remote_ch2);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.ch2);
            Send_MessageU16(&unsigned_connet);
            Set_TypeU(&unsigned_connet,MSG_remote_ch3);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.ch3);
            Send_MessageU16(&unsigned_connet);
            Set_TypeU(&unsigned_connet,MSG_remote_s1);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.s1);
            Send_MessageU16(&unsigned_connet);
            Set_TypeU(&unsigned_connet,MSG_remote_s2);
            Set_Datau16(&unsigned_connet,(uint16_t)RC_CtrlData.s2);
            Send_MessageU16(&unsigned_connet);
        }
        Send_End();
}

void publish_las(void)
{
        Send_Begin(SVEL);
        if(SEND_STA)
```

```
    {
        Set_TypeU(&unsigned_connet,MSG_distance_f);
        Set_Datau16(&unsigned_connet,(uint16_t)Distance_F);
        Send_MessageU16(&unsigned_connet);
        Set_TypeU(&unsigned_connet,MSG_distance_b);
        Set_Datau16(&unsigned_connet,(uint16_t)Distance_B);
        Send_MessageU16(&unsigned_connet);
        Set_TypeU(&unsigned_connet,MSG_distance_l);
        Set_Datau16(&unsigned_connet,(uint16_t)Distance_L);
        Send_MessageU16(&unsigned_connet);
        Set_TypeU(&unsigned_connet,MSG_distance_r);
        Set_Datau16(&unsigned_connet,(uint16_t)Distance_R);
        Send_MessageU16(&unsigned_connet);
    }
    Send_End();
}
void publish_state(void)
{
    Send_Begin(SSTATE);
    if(SEND_STA)
    {
        Set_TypeU(&unsigned_connet,IsFinsh);
        Set_Datau16(&unsigned_connet,(uint16_t)IsFinsh);
        Send_MessageU16(&unsigned_connet);
    }
    Send_End();
}
```