PPT Switching With Two TFmini-Plus

1. Introduction of the PPT automatic switching module

This module is designed by using two TFmini Plus of Benewake(BEJ)Co.Ltd, an Arduino DUE boards, a low-power infrared laser indicator, two LED lamps, a one-way switch, a two-way switch and a connection line

Function: By detecting the gesture of the person, the last page and the next page of the PPT are switched.

2. System Elements and Wiring

2.1 System Elements

- **Benewake Standard TFmini Plus**

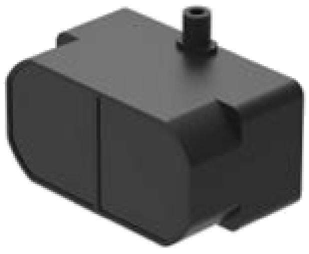

Fig.2.1 TFmini Plus

TFmini Plus(as shown in Figure 2.1). For detailed information, please refer to the TFmini Plus instruction.

This module is used to detect gestures of people. When the hand or arm is swept, the ranging status changes.

- **Arduino DUE Board**

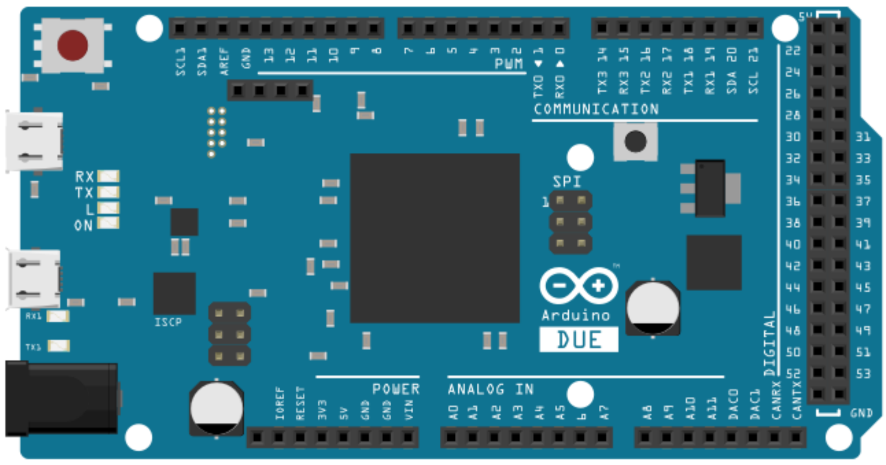

Fig.2.2 Arduino DUE

This module uses DUE board to detect the ranging status of LiDAR 1 and LiDAR 2, and realizes HID function by using USB communication.

Arduino UNO board（as shown in Figure 2.2）. For detailed introductions, please refer to the following two websites:

Chinese Community：http://www.arduino.cn/；

English Official Website：http://www.arduino.cc/ 。

- **Laser indicator**

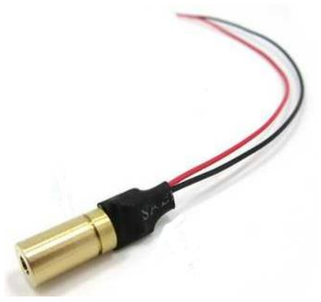

Fig.2.3 Infrared laser indicator

As shown in Figure 2.3, it is an infrared laser indicator that indicates the working direction and approximate area of the LiDAR.

For detailed introductions, please refer to：https://m.tb.cn/h.eml6WNY?sm=032032

- **LED**

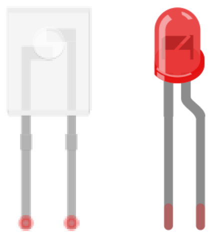

Fig.2.4 Function indicator

As shown in Figure 2.4, the LED is used to indicate the functional status of the module: the green light is turned on for the function, and the red light is turned off.

- **One-way switch**

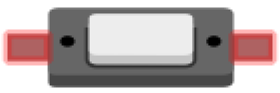

Fig.2.5 One-way switch

As shown in Figure 2.5, it is a one-way switch to control the opening and closing of the infrared laser indicator.

- **two-way switch**

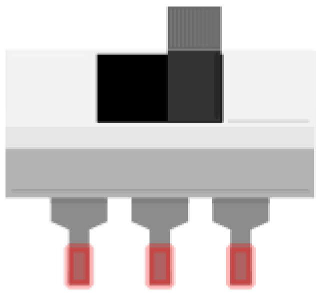

Fig.2.6 Two-way switch

As shown in Figure 2.6, it is a two-way switch used to control the opening and closing of the function of the PPT switching module.

- **Computer**

Fig.2.7 Computer

As shown in Figure 2.3, it is the host computer of the system, which is used to write programs and upload them to the DUE board and power the modules.

- **Cables**

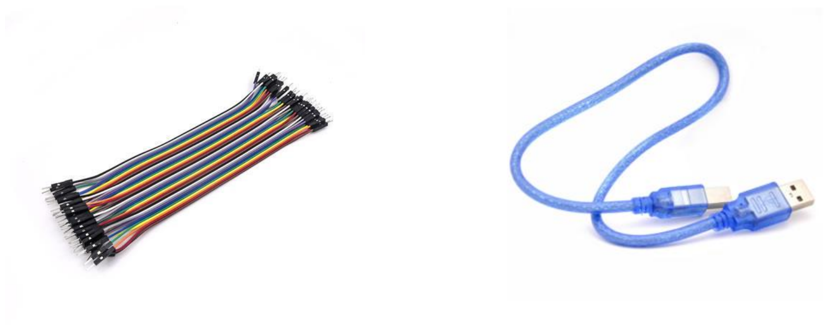

Fig.2.8 cables

Dupont line – connect the TFmini Plus and UNO board;

USB data cable - used for UNO board and computer communication and power supply.

2.2 Wiring

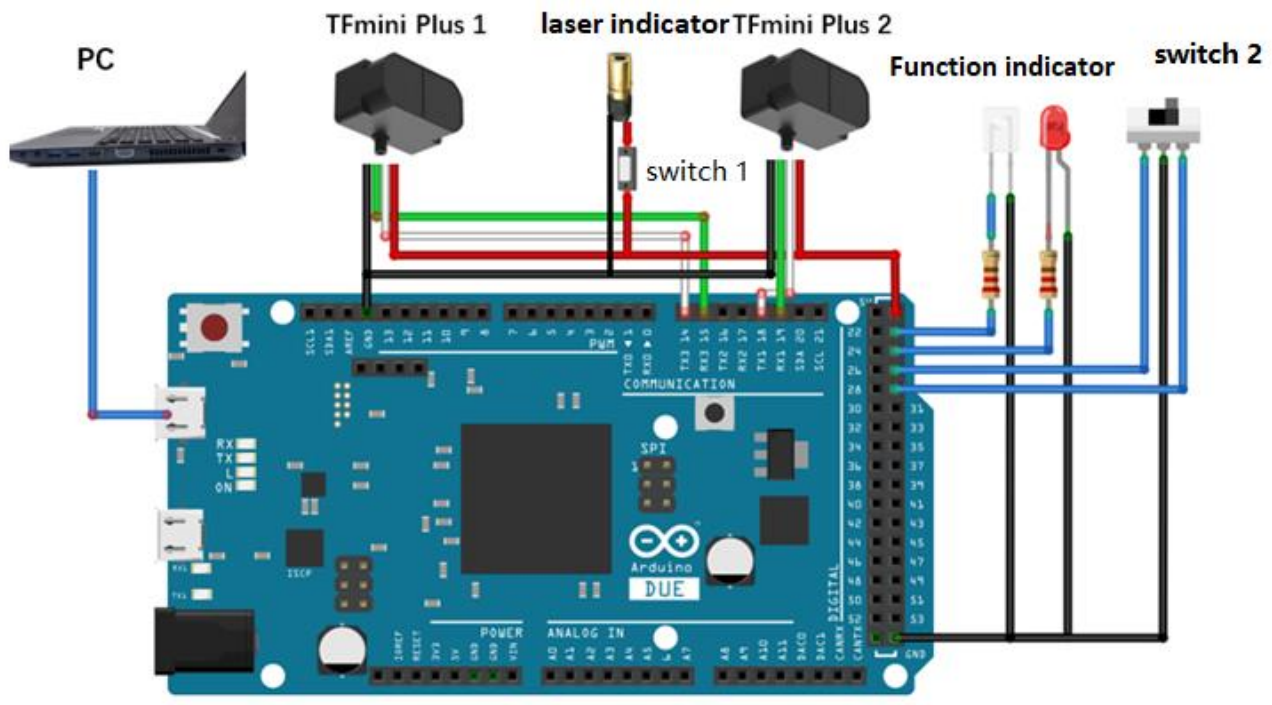

Fig.2.9 System wiring

Note:

（1） The TFmini Plus has a 5V supply voltage and is directly connected to the 5V and GND of the Arduino DUE board. Other LiDAR need to consult the product specifications to ensure that the power supply is normal.

（2） LiDAR 1 (RX, TX) is connected to the DUE board (TX3, RX3); LiDAR 2 (RX, TX) is connected to the DUE board (TX1, RX1);

（3） The function indicator is connected to the 23, 25 pins. By detecting the status of the switch, the corresponding function indicator is turned off and on. When the function is turned on, the green light is on; when the function is off, the red light is on;

3. System Work and Attentions

- **Attentions**

（1） The maximum frame rate of TFmini Plus is 1000, and the arm swing speed should not be too large.

（2） The arm sliding speed should not be too small, that is, the triggering time is in accordance with normal operation;

（3） It can be achieved that people walk at a general speed and do not trigger PPT switching.

- **Workflow**

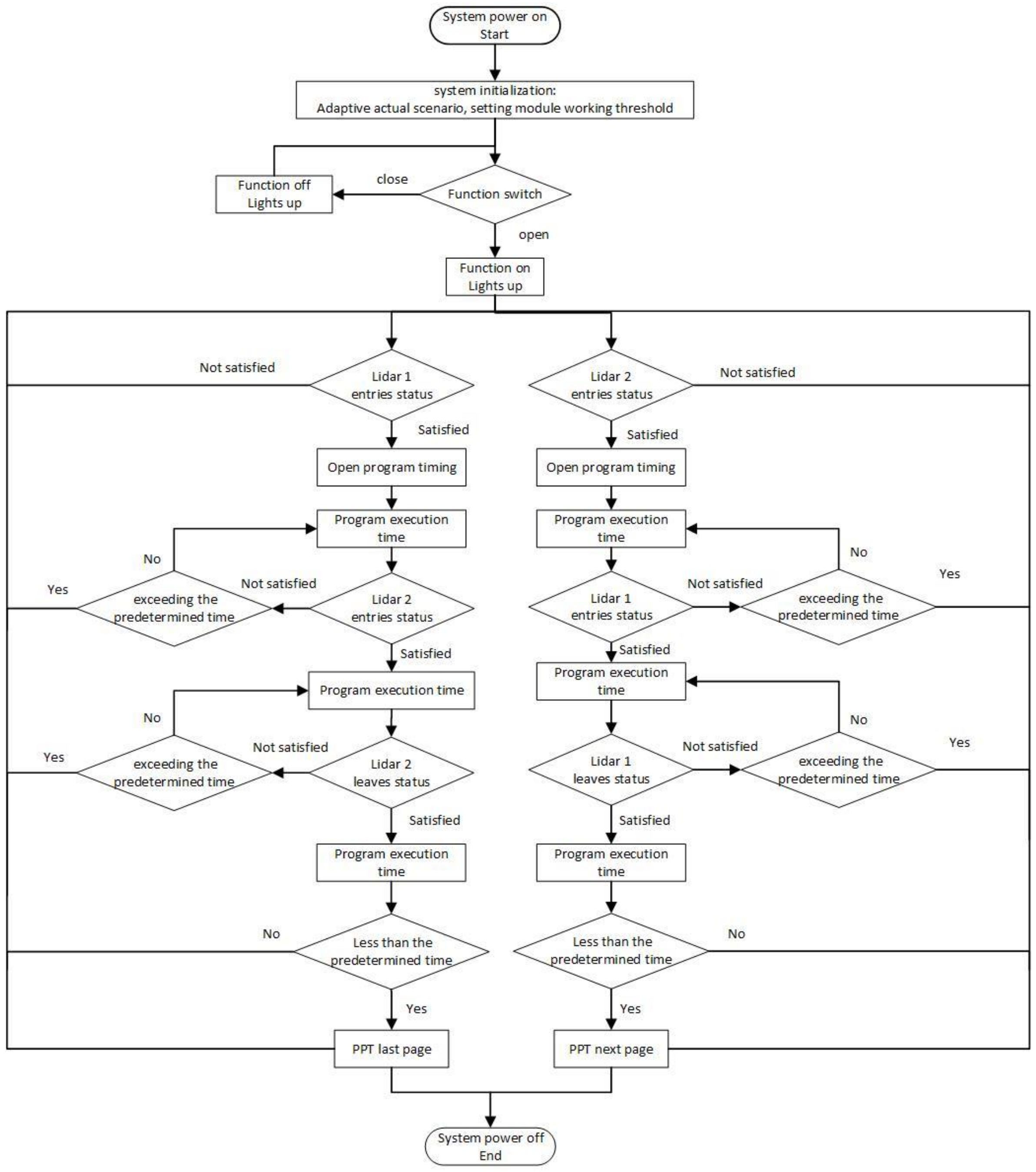

Fig.3.1 System workflow diagram

4. System Programming

```
#include <Keyboard.h>
/*******************************************************************
define pin27,29 as control and pin23,25 as display
```

```
*********************************************************************/
#define ON 23
#define OFF 25
#define LED_ON 27
#define LED_OFF 29
char left    = KEY_LEFT_ARROW;
char right    = KEY_RIGHT_ARROW;
const int HEADER=0x59;// frame header of data package
/*********************************************************************
Define module function initial state parameters
*********************************************************************/
bool receiveok1=false;
bool receiveok2=false;
bool status_complete1=false;
bool status_complete2=true;
/*********************************************************************
The variable DELAY can control the delay time of the module to filter out the misoperation caused by walking.
*********************************************************************/
int DELAY=180;
int distance1=0;
int distance2=0;
int DISTANCE1=1200;
int DISTANCE2=1200;
int threshold=0;
unsigned long tim[2]={0};
int period=0;
/*********************************************************************
Declarative function, LiDAR data acquisition function and acquisition environment distance threshold function
*********************************************************************/
int TFminione(void);
int TFminitwo(void);
int lidar_average(void);




/*********************************************************************
Pin function definition, initialization;
Serial port initialization;
Collect the surrounding environment and determine the module working threshold;
*********************************************************************/
void setup()
```

```
{
   pinMode(ON,INPUT);
   pinMode(OFF,INPUT);
   pinMode(LED_ON,OUTPUT);
   pinMode(LED_OFF,OUTPUT);
   digitalWrite(ON,HIGH);
   digitalWrite(OFF,HIGH);
   digitalWrite(LED_ON,LOW);
   digitalWrite(LED_OFF,LOW);
   Serial1.begin(115200);// set bit rate of serial port connecting LiDAR1 with Arduino
   Serial2.begin(115200);// set bit rate of serial port connecting LiDAR2 with Arduino
   Serial3.begin(115200);// set bit rate of serial port connecting Arduino with computer
   int threshold1=lidar_average();
   threshold=threshold1-100;
   Serial3.print("threshold=");
   Serial3.println(threshold);
}
/****************************************************************
First, determine the status of the function switch;
Then, determine the LiDAR ranging state;
If the judgment condition is satisfied, the corresponding operation is performed;
****************************************************************/
void loop()
{
          if(digitalRead(ON)==LOW)// Control switch open
          {
               digitalWrite(ON,HIGH);
               digitalWrite(LED_OFF,LOW);
               digitalWrite(LED_ON,HIGH);
               Keyboard.begin();
               receiveok1=false;
               receiveok2=false;
               distance1 =    TFminione();
               distance2 =    TFminitwo();
               if(receiveok1 & receiveok2)
               {
                   if(DISTANCE1>threshold    & distance1<=threshold    & distance2>threshold)
                   {
                          tim[0]=millis();
                          DISTANCE1=distance1;
                          DISTANCE2=distance2;
                          while(true)
                          {
```

```
receiveok1=false;
receiveok2=false;
distance1 =    TFminione();
distance2 =    TFminitwo();
if(receiveok1 & receiveok2)
{
        tim[1]=millis();
        period=tim[1]-tim[0];
        if(DISTANCE2>threshold    & distance2<=threshold )
        {
                DISTANCE1=distance1;
                DISTANCE2=distance2;
                while(true)
                {
                        receiveok1=false;
                        receiveok2=false;
                        distance1 =    TFminione();
                        distance2 =    TFminitwo();
                        if(receiveok1 & receiveok2)
                        {
                                tim[1]=millis();
                                period=tim[1]-tim[0];
                                if(distance2>threshold & distance1>threshold )
                                {
                                      tim[1]=millis();
                                      period=tim[1]-tim[0];
                                      if(period<DELAY)
                                      {
                                              Keyboard.write(right);
                                              status_complete1=true;
                                              break;
                                      }
                                      else
                                      {
                                            status_complete1=true;
                                            break;
                                      }
                                }
                                else if(period>DELAY)
                                {
                                      status_complete1=true;
                                      break;
                                }
```

```
                              else
                              {
                                   DISTANCE1=distance1;
                                   DISTANCE2=distance2;
                              }
                         }
                    }
               }
               else if(status_complete1)
               {
                    status_complete1=false;
                    break;
               }
               else if(period>DELAY)
               {
                    break;
               }
               else
               {
                    DISTANCE1=distance1;
                    DISTANCE2=distance2;
               }
          }
     }
}
else if(DISTANCE2>threshold   & distance2<=threshold   & distance1>threshold)
{
     tim[0]=millis();
     DISTANCE1=distance1;
     DISTANCE2=distance2;
     while(true)
     {
         receiveok1=false;
         receiveok2=false;
         distance1 =   TFminione();
         distance2 =   TFminitwo();
         if(receiveok1 & receiveok2)
         {
               tim[1]=millis();
               period=tim[1]-tim[0];
               if(DISTANCE1>threshold   & distance1<=threshold )
               {
                    DISTANCE1=distance1;
```

```
    DISTANCE2=distance2;
    while(true)
    {
        receiveok1=false;
        receiveok2=false;
        distance1 =   TFminione();
        distance2 =   TFminitwo();
        if(receiveok1 & receiveok2)
        {
            tim[1]=millis();
            period=tim[1]-tim[0];
            if(distance1>threshold & distance2>threshold )
            {
               tim[1]=millis();
               period=tim[1]-tim[0];
               if(period<DELAY)
               {
                   Keyboard.write(left);
                   status_complete2=true;
                   break;
               }
               else
               {
                  status_complete2=true;
                  break;
               }
            }
            else if(period>DELAY)
            {
               status_complete2=true;
               break;
            }
            else
            {
                DISTANCE1=distance1;
                DISTANCE2=distance2;
            }
        }
    }
}
else if(status_complete2)
{
    status_complete2=false;
```

```
                                        break;
                                    }
                                    else if(period>DELAY)
                                    {
                                        break;
                                    }
                                    else
                                    {
                                        DISTANCE1=distance1;
                                        DISTANCE2=distance2;
                                    }
                                }
                            }
                      }
                      else
                      {
                          DISTANCE1=distance1;
                          DISTANCE2=distance2;
                      }
                      Keyboard.end();
                  }
              }

              else if(digitalRead(OFF)==LOW)
              {
                    digitalWrite(OFF,HIGH);
                    digitalWrite(LED_OFF,HIGH);
                    digitalWrite(LED_ON,LOW);
              }
              else
              {
                  delay(1);
              }
}


/*****************************************************************
LiDAR 1 acquisition environment distance function
*****************************************************************/
int TFminione(void)
{
   int dist=0;//  actual distance measurements of LiDAR
```

```
    int check;//  save check value
    int uart[9]={0};//  save data measured by LiDAR
    int i=0;
    if (Serial1.available())//check if serial port has data input
    {
      if(Serial1.read()==HEADER)//assess data package frame header 0x59
      {
        uart[0]=HEADER;
        if(Serial1.read()==HEADER)//  assess data package frame header 0x59
        {
          uart[1]=HEADER;
          for(i=2;i<9;i++)// save data in array
          {
            uart[i]=Serial1.read();
          }
          check=uart[0]+uart[1]+uart[2]+uart[3]+uart[4]+uart[5]+uart[6]+uart[7];
          if(uart[8]==(check&0xff))//verify the received data as per protocol
          {
            dist=uart[2]+uart[3]*256;//calculate distance value
            receiveok1=true;
            return(dist);
          }
        }
      }
    }
    else
    {
        delay(1);
    }
}
/****************************************************************
LiDAR 2 acquisition environment distance function
****************************************************************/
int TFminitwo(void)
{
    int dist=0;//actual distance measurements of LiDAR
    int check;//  save check value
    int uart[9]={0};//save data measured by LiDAR
    int i=0;
    if (Serial2.available())//check if serial port has data input
    {
      if(Serial2.read()==HEADER)//assess data package frame header 0x59
      {
            uart[0]=HEADER;
            if(Serial2.read()==HEADER)//assess data package frame header 0x59
            {
                uart[1]=HEADER;
                for(i=2;i<9;i++)//save data in array
                {
                    uart[i]=Serial2.read();
                }
                check=uart[0]+uart[1]+uart[2]+uart[3]+uart[4]+uart[5]+uart[6]+uart[7];
                if(uart[8]==(check&0xff))// verify the received data as per protocol
                {
                    dist=uart[2]+uart[3]*256;//calculate distance value
                    receiveok2=true;
                    return(dist);
                }
            }
        }
    }
    else
    {
        delay(1);
    }
}
int lidar_average(void)
{
    int i=0;
    int sum=0;
    int average=0;
    int temp[100]={0};
    while(true)
    {
        distance1=TFminione();
        if(receiveok1)
        {
            temp[i]=distance1;
            Serial3.println(distance1);
            i++;
            if(i>=100)
            {
                for(i=0;i<100;i++)
                {
                    sum+=temp[i];
                }
                average=sum/100;
                Serial3.print("average=");
                Serial3.print(average);
                Serial3.print('\t');
                i=0;
                sum=0;
                return(average);
                delay(1);
                break;
            }
        }
        receiveok1=false;
    }
}
```

5. Instructions

- **Workflow:**

1. Power on the module;
2. The system adapts the actual scene, collects the distance information, and sets the module working threshold;
3. Turn on the PPT switch function
4. Turn on the laser indicator to roughly adjust the module's working area, then turn off the laser indicator;
5. The palm or arm sweeps the module from top to bottom or from bottom to top, and passing through two radars in turn. When leaving the work area, the PPT switches (last or next page);
6. Repeat step 4;
7. Power off the module.

- **Actual Structure**