无线视频采集程序

摄像头的工作原理其实就是将光学信号转换电信号的过程，通过图像传感器将物镜转换为电信号，然后再经过A/D转换成数字图像信号，最后将数字图像信号发送给摄像头内部DSP(数字信号处理)处理芯片进程处理。本系统设计将数字摄像头的驱动加载在无线路由器固件当中，当主控制器发出采集视频指令，摄像头开始工作，并将釆集回来的视频信息通过TCP协议利用WIFI无线路由器连接的信号发送到显示界面显示出来无线视频采集程序流程图如图所示：

红外程序

红外避障模块选用的红外避障传感器由发射管与接收管组成的，发射管发射红外线，当检测到障碍物时，接收管接受到红外线反射信号，经过比较器后，输出一个低电平信号送给单片机利用物体的反射性质，在一定范围内，如果没有障碍物，发射出去的红外线，因为传播距离越远而逐渐减弱，最后消失；如果有障碍物，红外线遇到障碍物，被反射到达传感器接收头，传感器检测到这一信号，就可以确认正前方有障碍物，并送给单片机，单片机进行一系列的处理分析，协调小车两轮工作，完成一个躲避障碍物动作。红外避障的控制程序的基本原理为首先检测是否有障碍物。当检测到有障碍物时，判断是做左边的传感器检测都到的还是右边的传感器检测到的，若是左边，则小车需先后退，然后右转。如过是右边的传感器检测到的，则小车应该先后退再左转。其算法流程图如图所示：

//GPIO

void MotorGPIO\_Configuration(void)

{

GPIO\_InitTypeDef GPIO\_InitStructure; //宏定义

GPIO\_InitStructure.GPIO\_Pin = FRONT\_LEFT\_F\_PIN; //为结构中的每一个元素定义

GPIO\_InitStructure.GPIO\_Speed = GPIO\_Speed\_2MHz; //同上

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_Out\_PP; //同上

GPIO\_Init(FRONT\_LEFT\_F\_GPIO, &GPIO\_InitStructure); //

GPIO\_InitStructure.GPIO\_Pin = FRONT\_LEFT\_B\_PIN; //收红外传感器连接的引脚传来的信号， backward

GPIO\_Init(FRONT\_LEFT\_B\_GPIO, &GPIO\_InitStructure); //进行处理

GPIO\_InitStructure.GPIO\_Pin = FRONT\_RIGHT\_F\_PIN; //收红外传感器连接的forwards信号

GPIO\_Init(FRONT\_RIGHT\_F\_GPIO, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = FRONT\_RIGHT\_B\_PIN;

GPIO\_Init(FRONT\_RIGHT\_B\_GPIO, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = BEHIND\_LEFT\_F\_PIN;

GPIO\_Init(BEHIND\_LEFT\_F\_GPIO, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = BEHIND\_LEFT\_B\_PIN;

GPIO\_Init(BEHIND\_LEFT\_B\_GPIO, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = BEHIND\_RIGHT\_F\_PIN;

GPIO\_Init(BEHIND\_RIGHT\_F\_GPIO, &GPIO\_InitStructure);

GPIO\_InitStructure.GPIO\_Pin = BEHIND\_RIGHT\_B\_PIN;

GPIO\_Init(BEHIND\_RIGHT\_B\_GPIO, &GPIO\_InitStructure);

}

//¸ù¾ÝÕ¼¿Õ±ÈÇý¶¯µç»ú×ª¶¯

void CarMove(void)

{

BEHIND\_RIGHT\_EN;

/\* //×óÇ°ÂÖ

if(front\_left\_speed\_duty > 0)//ÏòÇ°

{

if(speed\_count < front\_left\_speed\_duty)

{

FRONT\_LEFT\_GO;

}else

{

FRONT\_LEFT\_STOP;

}

}

else if(front\_left\_speed\_duty < 0)//Ïòºó

{

if(speed\_count < (-1)\*front\_left\_speed\_duty)

{

FRONT\_LEFT\_BACK;

}else

{

FRONT\_LEFT\_STOP;

}

}

else //Í£Ö¹

{

FRONT\_LEFT\_STOP;

}\*/

//ÓÒÇ°ÂÖ

if(front\_right\_speed\_duty > 0)//ÏòÇ°

{

if(speed\_count < front\_right\_speed\_duty)

{

FRONT\_RIGHT\_GO;

}else //Í£Ö¹

{

FRONT\_RIGHT\_STOP;

}

}

else if(front\_right\_speed\_duty < 0)//Ïòºó

{

if(speed\_count < (-1)\*front\_right\_speed\_duty)

{

FRONT\_RIGHT\_BACK;

}else //Í£Ö¹

{

FRONT\_RIGHT\_STOP;

}

}

else //Í£Ö¹

{

FRONT\_RIGHT\_STOP;

}

//×óºóÂÖ

if(behind\_left\_speed\_duty > 0)//ÏòÇ°

{

if(speed\_count < behind\_left\_speed\_duty)

{

BEHIND\_LEFT\_GO;

} else //Í£Ö¹

{

BEHIND\_LEFT\_STOP;

}

}

else if(behind\_left\_speed\_duty < 0)//Ïòºó

{

if(speed\_count < (-1)\*behind\_left\_speed\_duty)

{

BEHIND\_LEFT\_BACK;

} else //Í£Ö¹

{

BEHIND\_LEFT\_STOP;

}

}

else //Í£Ö¹

{

BEHIND\_LEFT\_STOP;

}

/\* //ÓÒºóÂÖ

if(behind\_right\_speed\_duty > 0)//ÏòÇ°

{

if(speed\_count < behind\_right\_speed\_duty)

{

BEHIND\_RIGHT\_GO;

} else //Í£Ö¹

{

BEHIND\_RIGHT\_STOP;

}

}

else if(behind\_right\_speed\_duty < 0)//Ïòºó

{

if(speed\_count < (-1)\*behind\_right\_speed\_duty)

{

BEHIND\_RIGHT\_BACK;

} else //Í£Ö¹

{

BEHIND\_RIGHT\_STOP;

}

}

else //Í£Ö¹

{

BEHIND\_RIGHT\_STOP;

}\*/

}

//ÏòÇ°

void CarGo(void)

{

front\_left\_speed\_duty=SPEED\_DUTY;

front\_right\_speed\_duty=SPEED\_DUTY;

behind\_left\_speed\_duty=SPEED\_DUTY;

behind\_right\_speed\_duty=SPEED\_DUTY;

}

//ºóÍË

void CarBack(void)

{

front\_left\_speed\_duty=-SPEED\_DUTY;

front\_right\_speed\_duty=-SPEED\_DUTY;

behind\_left\_speed\_duty=-SPEED\_DUTY;

behind\_right\_speed\_duty=-SPEED\_DUTY;

}

//Ïò×ó

void CarLeft(void)

{

front\_left\_speed\_duty=-20;

front\_right\_speed\_duty=SPEED\_DUTY;

behind\_left\_speed\_duty=-20;

behind\_right\_speed\_duty=SPEED\_DUTY+10;//Ôö¼ÓºóÂÖÇý¶¯Á¦

}

//ÏòÓÒ

void CarRight(void)

{

front\_left\_speed\_duty=SPEED\_DUTY;

front\_right\_speed\_duty=-20;

behind\_left\_speed\_duty=SPEED\_DUTY+10;//Ôö¼ÓºóÂÖÇý¶¯Á¦

behind\_right\_speed\_duty=-20;

}

//Í£Ö¹

void CarStop(void)

{

front\_left\_speed\_duty=0;

front\_right\_speed\_duty=0;

behind\_left\_speed\_duty=0;

behind\_right\_speed\_duty=0;

}

void MotorInit(void)

{

MotorGPIO\_Configuration();

CarStop();

}