**仿生飞鼠滑翔机器人文档说明**

****

北京航空航天大学

王林青 SY1607609

2018年12月10日星期一

## 滑翔控制策略

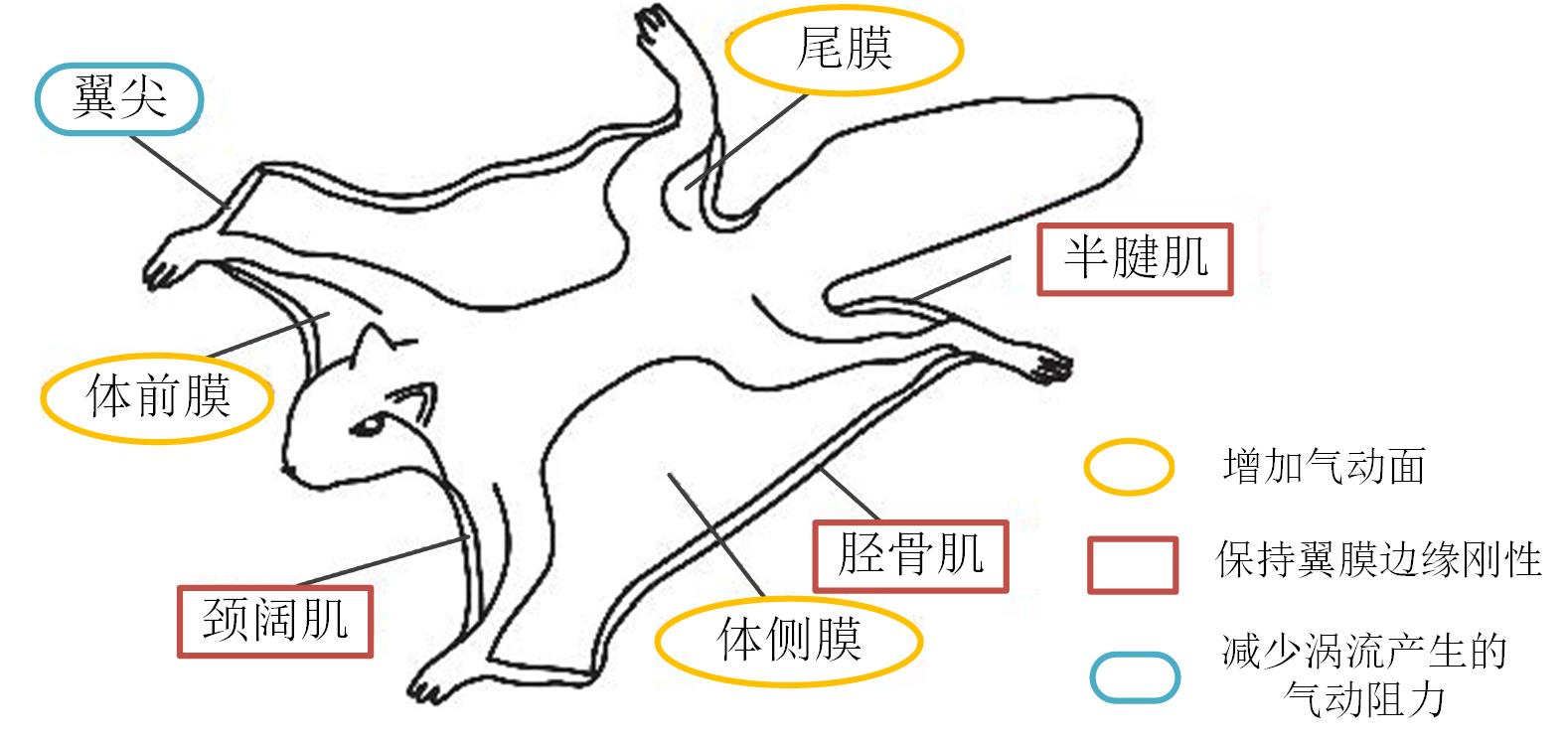


图1 飞鼠的总体构造

**1.1 经验控制**

飞鼠的俯仰控制是通过尾巴的摆动实现的，根据尾巴的摆动策略的不同，有经验控制策略、PID控制策略、模糊PID控制策略和全过程俯仰控制策略。其中全过程俯仰控制策略是上述几种俯仰控制策略的集大成者，理解全过程滑翔控制策略对理解以王老师为核心实验室提出的滑翔控制策略具有重要价值。

所有控制策略的本质用一句话概括为：当飞鼠的俯仰角变大时，向下摆动尾巴，否则向上摆动尾巴。



基于经验的控制策略

该控制策略很简单，将俯仰角与尾部摆角建立线性关系，一一对应。 其中 为尾部摆角， 为机器人的俯仰角。 为比例系数。基于经验的控制策略就是这样。

* 1. **PID控制策略**



PID不仅检测俯仰角，还检测俯仰角的变化，比例微分积分调节。详见论文。

* 1. **模糊PID**



模糊PID即是自动调参的PID，根据俯仰角的误差及误差变化率实时调整PID控制参数，详见论文。

* 1. **全过程俯仰控制策略**



说明看论文，这里仅仅讲一下算法实现。下位机的主要算法如下：

------------------------头文件-------------------------------

#include "stm32f10x.h"

#include "pwm\_output.h"

#include "Encoder.h"

#include "usart1.h"

#include "misc.h"

#include "SysTick.h"

#include "exti.h"

#include "led.h"

#include "Time\_test.h"

#include "stdlib.h"

// #include "math.h"

#include "nrf24l01.h"

#include "string.h"

#include "timer.h"

//#include "delay.h"

---------------------初始化变量-----------------------------------------

float a[3],w[3],angle[3],T;

float angle\_set=0.00;

extern unsigned char Re\_buf[11],temp\_buf[11],counter;

extern unsigned char sign;

uint32\_t En\_Counter1=0;

uint32\_t En\_Counter2=0;

extern volatile int16\_t pulse\_num;

u16 TIMCompare1=0;

#define pi 3.14;

static float ka=3.00;

static float kb=3.00;

static float kc=1.00;

static float kd=3.00;

static float head\_stable=0.00;

static float tail\_stab=0.00;

static float dif\_angle=0.00;

float head\_angle\_2=0.0;

//static float head\_stable=0.00;

extern u8 flag;

volatile u32 time; // ms 计时变量

volatile double velocity6 = 0;

volatile double velocity7 = 0;

volatile double velocity8 = 0;

volatile double velocity9 = 0;

int OUT5=9;

int OUT4=9;

int OUT3=9;

int OUT2=9;

float Motion\_angle1=0.0;

float Motion\_angle5=0.0;

float tail\_angle=0.0;

float head\_angle=0.0;

float head\_angle\_1=0.01;

float head\_angle\_bias=0.0;

float head\_angle\_add=0.0;

float speed=0.0;

int Output1 = 0;

int Output2 = 0;

int Output3 = 0;

int Output4 = 0;

int Output5 = 0;

s8 EncFullFlag = 0;

float A=116736/3;

float t=0;

float dett=0.02;

float f=1.0;

u8 array[32] = {3.0,30.0,0.0,30.0,4.0,50.0,50.0};

u8 ar=0;

u8 ta=0;

u8 aa=9;

int timeflag=0;

static float intialpian=0.0;

float pianhang\_angle=0.0;

float inityaw=0;

int yaw=0;

extern long int EXTI10\_VAL; //输入捕获值

extern long int EXTI10\_cycle; //输入捕获值

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*主函数初始化设置\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*

\* 函数名：main

\* 描述 ：主函数

\* 输入 ：无

\* 输出 ：无

\*/

int main(void)

{

unsigned char Temp[11];

SystemInit();

NVIC\_Configuration();//中断优先级设置

SysTick\_Init();

EXTIX\_Init(); //外部中断初始化

TIM\_PWM\_Init(); //设定定时器2 4为PWM

TIM3\_Configuration();//设定定时器3为PWM波发送20ms时段

NRF24L01\_Init();//无线设置

RCC\_Configuration(); //初始化时钟

GPIO\_Configuration(); //初始化IO端口

USART1\_Config(); //初始化 配置USART1

USART3\_Config(); //初始化 配置USART3 （用于接收MPU6050数据）

TIM3\_Int\_Init(499,7199);//对应200ms

Delay\_ms(200);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*可在while函数里面加无线传输函数\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*while循环核心程序\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

while(1){

if (sign){sign=0;

memcpy(Temp,Re\_buf,11);

if(Re\_buf[0]==0x55){

switch (Re\_buf[1])

{

case 0x51:

a[0] = ((short)(Temp[3]<<8 | Temp[2]))/32768.0\*16; //X轴加速度，16g

a[1] = ((short)(Temp[5]<<8 | Temp[4]))/32768.0\*16; //Y轴加速度

a[2] = ((short)(Temp[7]<<8 | Temp[6]))/32768.0\*16; //Z轴加速度

T = ((short)(Temp[9]<<8 | Temp[8]))/340.0+36.25; //温度

case 0x52: //标识这个包是角速度包

w[0] = ((short)(Temp[3]<<8| Temp[2]))/32768.0\*2000; //X轴角速度

w[1] = ((short)(Temp[5]<<8| Temp[4]))/32768.0\*2000; //Y轴角速度

w[2] = ((short)(Temp[7]<<8| Temp[6]))/32768.0\*2000; //Z轴角速度

T = ((short)(Temp[9]<<8| Temp[8]))/340.0+36.25; //温度

break;

case 0x53: //标识这个包是角度包

angle[0] = ((short)(Temp[3]<<8| Temp[2]))/32768.0\*180; //X轴滚转角（x 轴）

angle[1] = ((short)(Temp[5]<<8| Temp[4]))/32768.0\*180; //Y轴俯仰角（y 轴）

angle[2] = ((short)(Temp[7]<<8| Temp[6]))/32768.0\*180; //Z轴偏航角（z 轴）

T = ((short)(Temp[9]<<8| Temp[8]))/340.0+36.25; //温度

break;

default: break;

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*完成陀螺仪信号读取\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*算法设计+姿态调节\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

angle\_set=3.0\*(EXTI10\_cycle-2.0);// EXTI10\_cycle代表测速风扇返回读数，angle\_set表示设定俯仰角，根据公式当风速为零时，目标俯仰角设定为-6°，样机低头加速，当EXTI10\_cycle代表的风速达到2.0时，目标俯仰角为0，停止低头加速。

// if(EXTI10\_cycle<1)

// pianhang\_angle=0;

// else

\*\*\*\*\*\*\*\*\*\*可以将偏航角pianhang\_angle与风速通过以下关系关联\*\*\*\*\*\*\*\*\*\*\*

if(EXTI10\_cycle>=0 && EXTI10\_cycle<7)

pianhang\_angle=kd\*EXTI10\_cycle;//在一定的运动范围内向有偏转

//head\_angle=angle[1];

//head\_angle\_bias=angle[1]-angle\_set;//头部攻角与目标设定值之间的偏差

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*接收上位机的无线指令\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ta=NRF24L01\_RxPacket(array);

ka=array[1]/10.0;

kc=array[2]/10.0;

kb=array[3]/10.0;

kd=array[4]-10.0;

dif\_angle=-(array[5]-50.0);

intialpian=array[6]-60.0;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*上述ka kc kb kd 分别代表控制从上位机控制参数\*\*\*\*\*\*\*\*\*\*\*\*

//new added by Jack!;

head\_angle=angle[1]+head\_stable-angle\_set;

head\_angle\_add=head\_angle-(head\_angle\_1+head\_angle\_2)/2;

if((head\_angle>-5)&&(head\_angle<5)&&(head\_angle\_add<1)&&(head\_angle\_add>-1))

head\_angle\_add=0;

if((head\_angle>=-10)&&(head\_angle<10))

Motion\_angle1=ka\*head\_angle-kc\*head\_angle\_add;

if((head\_angle>-25)&&(head\_angle<-10))//样机在【6-15角度范围内抬头】

Motion\_angle1=kb\*(head\_angle+10)-10\*ka-kc\*head\_angle\_add;//改变尾巴俯仰，调节样机头部

if((head\_angle>=10)&&(head\_angle<25))//样机在【6-15角度范围内低头】

Motion\_angle1=kb\*(head\_angle-10)+10\*ka-kc\*head\_angle\_add;//改变尾巴俯仰，调节样机头部

Motion\_angle5=intialpian+pianhang\_angle;

Output1=9.1469\*(Motion\_angle1+dif\_angle-18)+1399.43;

Output5=-11.968\* (Motion\_angle5+8.0)+1351.32;

TIM\_SetCompare2(TIM4,Output1);//second pin the num1 motion

TIM\_SetCompare3(TIM2,Output5);//前肢运动控制

head\_angle\_2=head\_angle\_1;

head\_angle\_1=head\_angle;

}

}

}

}

//增加无线模块

上位机界面

