# 划船机算法文档

## 算法思想

基本思路：

划次计算：对加速度数据及进行滤波。按照过零的思想，找一个波谷，再匹配一个波峰，每完成一个配对，就记为一个划次

拉桨时间与回桨时间：找到连续的两个波峰时，计算一次周期，在此周期中，计算连续的正数序列的最大长度作为拉浆时间。

## 数据结构

划船结果结构体

typedef struct ResultStruct

{

int32\_t rowCounts; //划次

float avgRowDuration; //平均时间

int16\_t avgRowCountPerMin; //分均划频 :: 最后展示

int16\_t avgRowCountPerMinNow; //实时划频 :: 当前展示

float avgPaddleTime; //拉桨时间

float avgReturnPaddleTime; //回桨时间

}RowResultStruct;

typedef struct RowGroupInfo\_t

{

int32\_t mCount; //Row Number

int32\_t mTime; //Time

float mCalories;

int32\_t mAvgHeartRate;

float mAvgCadence;

}RowGroupInfo\_t;

实时数据结构体

//Real-Time Data:Used as the input of program

typedef struct RealTimeData

{

int16\_t acc[3];

int16\_t ang[3];

}RealTimeData;

原始数据栈结构体

//Data Stack :Save a certain number of points

typedef struct RawDataStack

{

RealTimeData\* Data;

int16\_t DataStackNum;

}RawDataStack;

数据栈结构体

typedef struct DataStack

{

int16\_t\* Data;

int16\_t DataNum;

}DataStack; //Information extracted from the original data after filtering

中间变量结构体

//Intermediate variables

typedef struct TempStack

{

int8\_t trend; //Point`s G

int32\_t IsolatedPointsNumber; //Number of isolated points

int32\_t Count1;

int32\_t CycleLimit;

float SimiLimit; //The lower limit of Waveform similarity

int8\_t hand;

int32\_t PeakLoc;

int32\_t LastPeakLoc;

int32\_t ValleyLoc;

int32\_t LastValleyLoc;

float CycleTime[3]; //Average cycle time

float PaddleTime[3]; //Used to calculate the average Paddle time

int16\_t PeakValueStack[3];

int16\_t PeaksLocStack[3];

int16\_t Wave[WAVELEN];

int16\_t windth;

}TempStack;

## 函数功能

1. void row\_initial(bool handside); //Initialization parameters

功能：初始化程序中用到的全局变量和临时变量

输入参数：

* handside：设置佩戴手表的左右手。
  + bool型
  + 当handside等于Ture（1）时，表示手表佩戴在左手上
  + 当handside不等于Ture（1）时，表示手表佩戴在右手上

1. void rowing\_receiveAccGyro(int16\_t\* acc\_buf, int16\_t\* gyro\_buf); //Real-time detection

功能：算法的主函数，根据当前的输入统计得到的结果

输入参数：

* acc\_buf：输入的陀螺仪数据中的加速度数据数组，数组中的三个数据表示x、y、z各轴的加速度
  + int16\_t型
* gyro\_buf：输入的陀螺仪数据中的加速度数据，数组中的三个数据表示x、y、z各轴的加速度
  + int16\_t型

算法过程：

//Update Data in every round.

1. void UpdateDataStack(RealTimeData OnPoint);

功能：根据输入的点，更新数据栈

输入参数：

算法过程：

1. void UpdateMainDataStack();

功能：更新MainData数据栈，这个过程主要是对原始的数据进行滤波

算法过程：对加速度数据栈中的AccData及进行均值滤波；对角速度数据栈中的AngData进行高斯滤波

1. void UpdateTimeStack(float\* CycleTime, float Value);

功能：更新时间栈

输入参数：

* CycleTime
* Value

算法过程：

1. void UpdatePeakStack(int16\_t\* Peak, int16\_t Value);

功能：更新波峰栈

输入参数：

算法过程：

1. void UpdateTrend(int32\_t Num); //update the trend of wave in every round

功能：更新Trend

输入参数：

* Num
  + Int32\_t型
  + 表示RawData栈中的第Num个点

算法过程：

* 当Trend等于-2时，表示当前还在第一个点，将其更新为0即可
* 当Trend等于-1时，表示数据处于下降阶段
* 当Trend等于1时，表示数据处于上升阶段

1. int8\_t checkValley(int32\_t Num, int32\_t sample\_num); //check if the current point is a valley\trough

功能：检查当前点的前一个点是不是一个真正的波谷

输入参数：当前点的序号，总的时间点序号

算法过程：

//Get & Print Result or RealTimeData

1. void getRowingResult(RowResultStruct\* result); //Get the data in real time

功能：获取当前结果

1. void printResult(RowResultStruct Result); //Print result data

功能：打印结果

输入参数：划船结果结构体 Result

1. void PrintRealTimeData(RealTimeData OnPoint); //Print the data in real time

功能：打印实时数据

输入参数：

//Operation On Array

1. int8\_t sign(int16\_t a);//Judging the sign of the value

功能： 判断数据符号

输入参数：

* a：输入数据
  + int16\_t类型

1. float ComputeVar(int16\_t\* Array, int16\_t length); //Calculate the variance of an array

功能： 计算数组方差（准确的来说，计算的是方差/（元素个数-1））

输入参数：

* Array：整形数组
* Length：整形数组的长度

1. int16\_t GetArrayMax(int16\_t\* Array, int16\_t Num); //Compute the Max Element of a Array

功能： 计算数组元素的最大值

输入参数：

* Array：整形数组
* Length：整形数组的长度

1. int16\_t GetArrayMean(int16\_t\* Array, int16\_t len); //Compute Mean of a Array

功能： 计算数组元素的平均值

输入参数：

* Array：整形数组
* Length：整形数组的长度

1. float GetFloatArrayMean(float\* Array, int16\_t len); //Compute the Mean of a Float Array

功能： 计算浮点数数组元素的平均值

输入参数：

* Array：整形数组
* Length：整形数组的长度

1. int16\_t GetArrayMaxPosNumLen(int16\_t\* Array, int16\_t start, int16\_t end); //Count the number of positive values between two peaks

功能： 计算整数序列中连续的最长的正数数目

输入参数：

* Array：整形数组
* Start：起始位置序号
* End：结束位置序号

//Compute Similarity

1. float ComputeSimilarity(float\* Array, int Num1, int16\_t\* Wave); //Compute the similarity of two signal

功能： 计算两段序列之间的相似性

输入参数：

* Array：浮点型数组
* Num1：数组个数
* Wave1：整型数组

1. void SeqResample(int16\_t\* data, int n, int des\_len, float\* rseq);

功能： 序列重采样

输入参数：

* Array：浮点型数组
* Num1：数组个数
* Wave1：整型数组

1. float inter\_linear(float x0, float x1, float y0, float y1, float x);

功能： 线性插值

输入参数：

* Array：浮点型数组
* Num1：数组个数
* Wave1：整型数组

1. float interp1(float x[], int16\_t y[], int n, float px);
2. void MultiPointInterp(float x[], int16\_t y[], int n, float px[], int m, float py[]);