## 概述

该库提供了用于航速航向跟踪的控制器，发送指定航速航向后，机器人跟踪指定航速航向指令，具有加减速约束。

此库扩展为能够实现多机器人的航速航向控制。

**用法**

推荐的方法是将 xx.h 添加到项目中，并使用适当的构建系统进行编译。

**约束**

该库生成及测试于Win系统

Qt版本Qt 5.12.10 MinGW 64-bit

编译器MinGW 7.3.0 64-bit for C++

## 例子

#include "cusvmodel.h"

#include "cmultiusvmodel.h"

//算法实现

std::vector<USVInitStructType> **algorithmImpl**(int algIterNum, std::vector<SFormationTaskInfo> taskPath, std::vector<referUSVInfo> referInfo, std::vector<USVOutPutStructType> multiUSVStatus);

int **main**()

{

cmultiUSVModel cmultilUSV;

std::vector<SFormationTaskInfo> taskPath; //任务航路

std::vector<referUSVInfo> referInfo; //构型信息

std::vector<USVOutPutStructType> multiUSVStatus; //导航信息

std::vector<InputOBSStruct> curOBSFusion; //态势信息

std::vector<USVInitStructType> multiUSVInitInfo; //各艇控制指令

int flagModel = 1; //控制模式，1-航速航向；2-航路，建议使用航速航向

int iterNum = 0; //整个任务迭代次数

int algIterNum = 0; //算法更新后清零

bool rcvFlag; //任务触发标志

cmultilUSV.multiInitialize(flagModel); //初始化

while(1)

{

//更新控制指令

cmultilUSV.updateMultiUSVControl(multiUSVInitInfo,iterNum,flagModel);

//获取导航信息和态势信息

cmultilUSV.getMultiNaviObsInfo(iterNum,*multiUSVStatus*,*curOBSFusion*);

//获取任务信息

bool isEnd;

rcvFlag = cmultilUSV.getTaskInfo(*isEnd*,*taskPath*,*referInfo*,iterNum);

//触发任务

if(true == rcvFlag)

{

if(true == isEnd)//到达最后一个任务点，不再发送任务

{

cmultilUSV.taskEnd(iterNum,flagModel);

break; //跳出算法循环

}

algIterNum = 0; //新任务，次数清零

}

//算法更新控制指令

multiUSVInitInfo = algorithmImpl(algIterNum,taskPath,referInfo,multiUSVStatus);

//迭代次数更新

++iterNum;

++algIterNum;

}

}

std::vector<USVInitStructType> **algorithmImpl**(int algIterNum,std::vector<SFormationTaskInfo> taskPath,std::vector<referUSVInfo> referInfo,std::vector<USVOutPutStructType> multiUSVStatus)

{

//根据实时位置，任务信息求下一时刻艇的期望航速航向

// std::vector<SFormationTaskInfo> taskPath; //编队航线

// std::vector<referUSVInfo> referInfo; //编队构型

// std::vector<USVOutPutStructType> multiUSVStatus; //编队内艇信息

std::vector<USVInitStructType> multiUSVInitInfo;

for(int i = 0;i<multiUSVStatus.size();++i)

{

USVInitStructType updateUSVControl;

updateUSVControl.iUSVId = multiUSVStatus[i].iUSVId;

updateUSVControl.dbInitX = multiUSVStatus[i].dbCurX;

updateUSVControl.dbInitY = multiUSVStatus[i].dbCurY;

updateUSVControl.fInitVel = 5;//multiUSVStatus[i].fCurVel; //速度(米/秒),

updateUSVControl.fInitYaw = 90;//multiUSVStatus[i].fCurYaw; //航向(度))

multiUSVInitInfo.push\_back(updateUSVControl);

}

return multiUSVInitInfo;

}

**期望在algorithmImpl函数中进行算法开发，也可根据函数意义自行组织。**

**说明**

控制器的更新步长为0.1s；

库中集成了Jason读取和txt数据存储功能；

该库的有效时间至2025年6月；

版权归发布单位所有。