上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

论文题目：三体船的循迹控制

彭正皓	515021910506
刘子平	515021910250
支鹏飞	515021910369
胡经朝	515021910523
刘彬杰	515021910509
宋深科	515021910648

指导教师：许劲松教授

2017 年 12 月 22 日，思源湖

目　录

第一章　简介

三体船是一种稳性很高的船舶，其具有三个浮体位于水线及水面以下。由于三个浮体在同样排水体积的情况下减少了水线面面积，因此能降低船舶阻力，提高机动性。三体船的两侧各设有一个螺旋桨用以产生推力。不像一般的船舶需要用舵来推力方向，三体船通过差速控制来控制航向。操作者通过控制左右螺旋桨的不同转速，来产生左右不同的推进力，从而实现对船只航向的控制。

在船舶的无人化领域，一般将实现船舶的无人控制分为三个重要问题，第一是规划问题，第二是控制问题，第三是感知问题。在本课程中，船舶运动的感知问题已经解决，我们会接收到船舶运行中的各项数据，因此前两个问题是我们关注的焦点。我们的任务就是设计一套完整的程序，其必须解决上述两个方面的问题。一方面是规划问题，程序必须能够产生合理的轨迹规划或理想量（如理想航向、理想速度等），使得船舶在偏离航线的基础上迅速恢复。另一方面是控制问题，程序必须根据硬件、环境的实际情况，将规划的轨迹或理想量转化成螺旋桨转速的指令。

这个学期，我们在许劲松老师的指导与秦操、邓乃铭学长的帮助下，成功地进行了三体船的循迹与控制实验，自主完成了所有规划和控制算法，使用 Python 语言，严格按照编程范式开发了一套除了通信模块以外集控制、循迹、仿真、监控于一体的软件平台。在规划问题下，我们尝试了简单的求等分点法，也尝试了复杂的 LOG 算法，并在仿真实验平台上进行了定量研究，确定了最优的循迹算法。而在控制问题下，我们经过仿真实验，得到了最优的航向角的 PID 控制参数。

本文展示了我们各自的贡献和感想，总结我们的工作和经验，以飨后人。首先我们介绍了两种循迹算法的思路，之后我们简略介绍无人船控制系统的各个组成部分。接着，我们介绍了将本程序部分一直到。然后是各个小组成员各自的贡献和感想。最后是总结和致谢。

第二章　循迹算法

循迹算法是无人船控制的核心，也是自动控制系统“控制、规划、感知”中的重要一环。为了简化问题，我们将循迹算法简化为如下模型：

给定任意一组首尾相连的线段以及对应的每段线段的出发点和终止点。在每一时刻，已知船体目前的位置，对应的目标线段以及其出发点和终止点，要求返回一个理想航向角，使得若船一直按给出的航向角行驶，其实际行驶轨迹与给出的所有线段最接近。

为了实现这个目标，具体到实际任务中，有以下几个子任务：

1. 要让船到对应线段的距离越近越好
2. 要让船在即将到达终止点时调整姿势准备开始下一段线段
3. 尽可能减少过冲
4. 保护硬件，避免高频改变推进指令
5. 做好安全防护，避免船体失控

在学期的前期，我们提出了等分点法，并通过仿真实验平台确定了具体的参数。后来，我们采用 LOG 法，获得了更好的结果。下面分别进行介绍。

2.1　等分点法

该方法的基本思想，是将理想航向角设定为从船体目前所在位置指向设定点的角度，所谓设定点指的是船体所在位置到目标线段的垂足与终点连成线段的 n 等分点。

n 是一个可变的参数，显然如果 n 越大，理想航向就会更加接近于与目标线段垂直的方向，而如果 n 较小，理想航向会更接近于指向终点。如果 n 取得过小，当船与目标线段有垂向偏移的时候，就需要更长的时间才能恢复，反之若 n 过大则会导致船的垂向运动过于频繁，而沿目标线段方向的运动速度有所下降。因此 n 的取值是一个需要探究的参数，

我们利用仿真模块（见3.2），做了系列实验来探寻 n 最优的取值。

仿真实验结果展示在表2–1中，表第二列的值为 Cost，是根据所有时刻船实际位置距目标航线的垂直距离的平方的平均值，即平均平方误差（Mean Squared Error）。可以看出，n 取 6 是最好的。

2.2　LOG 法

LOG 法的示意图见2–1。其具体工作原理如下：现有从起始点指向目标点的直线以及无人船当前的实际位置，以无人船当前位置坐标为圆心，以一定值为半径（一般取 1-2 倍船长，由于三体船船型特殊，须由实验测定）作圆，下分两种情况：

1. 当圆与直线无交点，作船的坐标到直线的垂线，设置航向指向垂足；2. 当圆与直线相交，设置航向指向交点中靠近目标点的那一个。

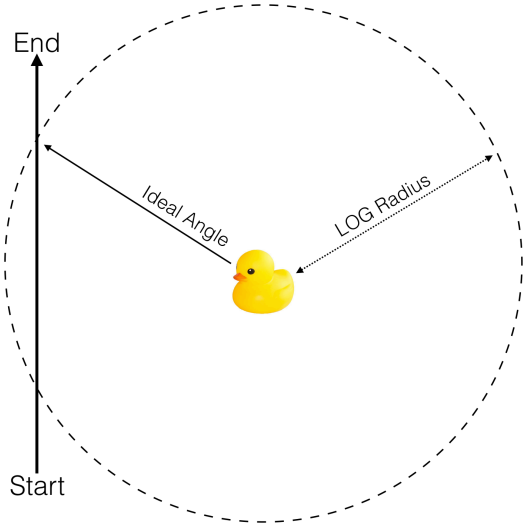


图 2–1 LOG 法示意图

该算法的特点在于能根据船与理想航迹的相对位置实时调整航向与理想航迹的夹角。当船与理想航际的距离太大，算法将直接使船以最快的速度靠近理想航际，在船接近理想航际以后，理想航向将视船与理想航际的距离而定：

距离越远，理想航向与直线所夹锐角将越大（无交点情况可视为夹角为 90 度），船靠近直线的速度分量将更大，即优先靠近；距离越近，理想航向将越接近理想航迹的方向，船的大部分速度分量用于向目标点移动，而有少部分垂直于直线的速度分量用于调整船与航线的相对位置，即优先到点。

表 2–1 等分点法实验结果

设定点	Cost
终点	0.54838413925
中点	0.48295447221
三等分	0.504954890885
四等分	0.427074577216
五等分	0.408871954602
六等分	0.381931136109
七等分	0.382402660838
八等分	0.466125888686

表 2–2 LOG 法实验结果

LOG 半径	Cost
9	0.414994311653
8	0.414674139793
7	0.385543802982
6.75	0.372176121327
6.625	0.403356029328
6.5	0.378798024698
6	0.404670573479
5	0.414300619607
4	0.396560181534
3	0.608423937903

2.3　对比

2017 年 12 月 1 日，我们进行了等分点法与 LOG 法的对比实验。由图2–2与图2–3的比较可以看到，LOG 法略胜一筹，这与我们的仿真结果相符。虽然实船实验的时候 Cost 值总体上大于对应的仿真实验（这是由于风、水阻力及船体硬件的性能与仿真模型并不严格相符），但是 LOG 法的 Cost 比等分点法的少。

因此我们最终选用的循迹算法是 LOG 法。

图 2–2 等分点法（$n = 6$）实船实验结果

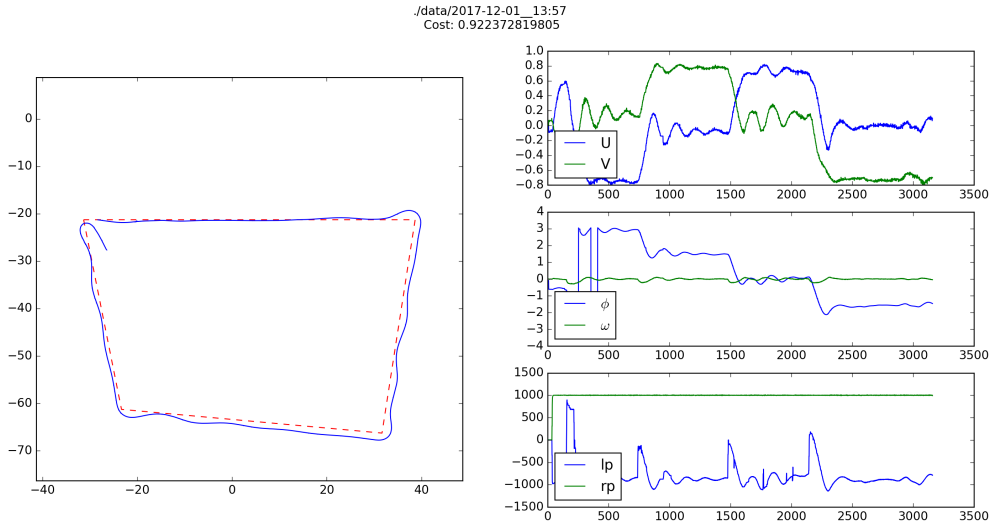
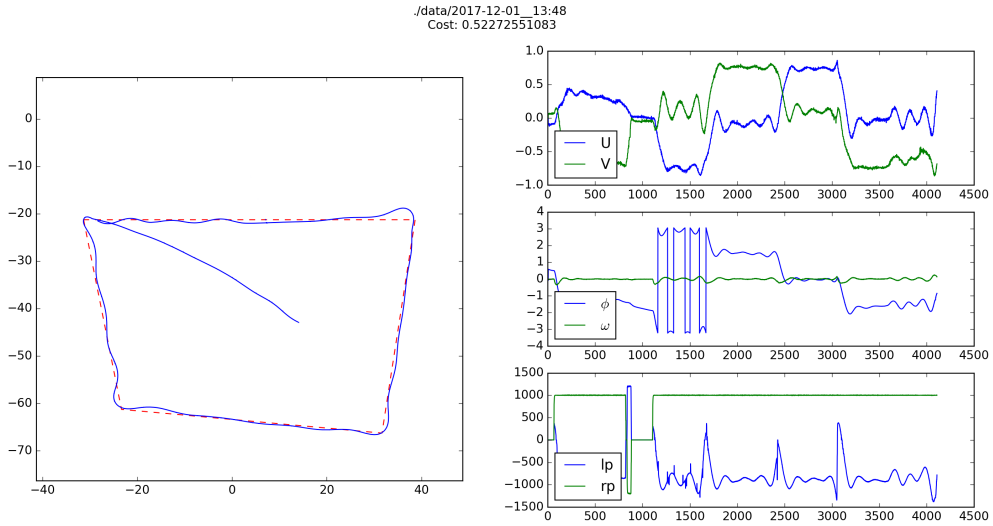


图 2–3 LOG 法（$r = 6$）实船实验结果



第三章　软件平台

为了获得更大的自由度，我们放弃了使用 Matlab 的 Simulink 模型与 Veristand 配合控制无人船的方法，改为从零开始（除了 MsgDev 库，即前人写好的通讯模块）开发了一套除了通信模块以外成熟的集控制、循迹、仿真、监控于一体的软件平台。本章就软件平台的各个模块进行详细的介绍。

3.1　决策模块（DecisionMaker）

本模块是对循迹算法的实现。其内部存储了目标航线上的点的坐标，也存储了目前船舶正处于哪段航线上。也就是说本模块能够根据船舶的状态做出除了理想航向角以外一些别的处理。具体而言，我们引入了转速衰减的功能，当船舶距离此段的终点一定范围内（即衰减半径内），船舶的基准转速会从 1000 转随距终点距离线性衰减，这样有利于缓解过冲。

此模块设计了几个简单的接口，方便他者调用。如 *getCost(state)* 函数返回当前点至当前目标航线的垂直距离的平方，此量对所有时刻平均，即得到上文提到的 Cost 值。又如 *wrapper_LOG* 函数会基于 LOG 法，接受当前状态、LOG 半径，返回理想航向角和衰减因子。*wrapper_divide* 函数会基于等分点法，根据当前状态返回理想航向角和衰减因子。理想航向角随后会被送入 PID 模块。衰减因子 *decay* 会在3.3中介绍。

3.2　仿真模块（Simulator）

仿真模块将秦操学长提供的三体船动力学 Simulink 模型移植到了 Python 上，也就是 *simulate(state, left, right, dt)* 函数，它可以输入上一时刻的船舶六个状态量：位置 x, y，绝对速度 x, y，首向角和角加速度 ϕ, ω，以及左右推力和时间间隔，返回下一时刻的六个状态。

本模块通过与主函数（Main）几乎一样的流程——设置目标航线，PID 参数，调用绘图模块，PID 模块，决策模块，来对实船的航行进行仿真，唯一不同之处在于实船航行下一时刻的数据需要通讯模块获得，而在本模块中是通过 *simulate* 函数获得。

得益于此模块，我们就可以通过几句简单的循环指令，不断尝试各种我们需要探究的参数，如循迹算法参数、PID 参数等。

3.3　PID 模块（PID）

在三体船的循迹控制中，PID 的输入是理想航向角和实际航向角，输出是左右桨的转速的增量。所谓增量指的是左右桨的转速在一个恒定值的基础上，加上（可以为负值）*output*，从而修正实际航向角和理想航向角之间的偏差。在实际应用中，有两种叠加 PID 输出量与固定转速的方法：

$$left \leftarrow decay \times 1000 + output,\ right \leftarrow decay \times 1000$$

$$left \leftarrow decay \times 1000 + \frac{output}{2},\ right \leftarrow decay \times 1000 - \frac{output}{2}$$

可以想见，前者的转弯半径更大，而后者反之。实验也证实后者的过冲更小。

鉴于 Python 下没有方便可行的 PID 库，我们移植了 Arduino 平台上的 PID 库，将原来的 C++ 代码改写成 Python 代码，提供了 PID 的服务。利用秦操学长提供的三体船动力学模型，我们先在仿真平台上获得了理想状态下的 PID 三参数。后来经过实船实验的修正，我们得到了如今的三个 PID 参数：

- $K_p = 1200$
- $K_i = 3$
- $K_d = 10$

3.4　通讯模块（Communicater）

此模块是对张磊学长提供的 MsgDev 库进行的二次开发，封装好了几个常用的函数如提供绝对坐标系数据的 *getNEData()*，上传左右桨转速的 *upload(left, right)*，存储此时刻数据的 *record()* 等。

关于 MsgDev 库的详细运用，可以参见彭正皓的 Github 页面《MsgDev 库使用说明》[1]。

3.5　台式机转发模块（Transmitter）

三体船船载计算机（下位机）的程序，只能响应一个特定 IP 发来的指令，即岸上台式机（上位机)。为了避免修改下位机程序，我们在上位机上运行一个转发程序，将来自笔记本电脑的控制信号转发到下位机。笔记本上通讯器模块的目标端口从原来的 55002 改为某一不会产生冲突的端口（如 55010)，再在上位机的转发程序中，接收来自笔记本 IP 下 55010 端口的数据，并立即转发到 55002 端口，就可以实现转发笔记本发出指令到下位机的功能。

3.6　绘图模块（Drawer）

为了方便可视化仿真实验和实船实验的结果，而且提供实船运行中实时显示运行各项参数的功能，我们设计了绘图模块。它绘制了本文出现的所有反映船舶运行状态的图（如图2–2、图2–3、图********）等。左侧大图中的红色虚线是目标航线，另一条线为实际航线。右侧三张图表反映了船舶的另外四个状态量及左右螺旋桨转速。

3.7　主程序（Main）

主程序是程序的入口，我们先依次定义 PID 参数、坐标原点位置（每周实验都要重新标定原点)、目标航迹上各点的坐标，然后开启 PID 模块、绘图模块、通讯模块、决策模块，之后进入一个死循环，每隔 0.1s 执行一次以下流程：下载船舶状态，决策得到理想航向角，PID 得到转速，上传转速。**在实验结束时，上传左右转速皆为零的指令，关闭螺旋桨，**之后保存数据，结束程序。

[1] https://github.com/PengZhenghao/Autonomous-Tracking-Trimaran/blob/master/MsgDev 库使用说明.md

第四章　Matlab 实现

为了更好的满足课程最初要求的使用 Matlab 中的 Simulink 进行控制模型建立及用 NI Veristand 进行船体的控制与监控，我们也进行了 Simulink 模型的搭建（部分，其中船体模型由学长直接提供），并在 Matlab 里进行了仿真。这章就 Matlab 模型里的各个部分进行简要介绍，并对仿真运行的结果做简要说明。

4.1　DRE 循迹算法模块

该函数输入为无人船当前坐标（X,Y），输出理想航向角 ang，即 ang = DRE（X,Y）。该函数首先人工定义了给定路线的关键点（如多边形的各角点），首先以多边形的其中一个角点为出发点，其相邻角点为目标点，两点构成的向量即当前航向，不停判断当前无人船位置与目标点的距离，当距离小于一给定值时，认为船已到达当前航向的目标点。改变该目标点为下一段航向的出发点，不断循环，完成完整航迹的遍历。在未到达当前目标点前，通过第二章所述的多种循迹算法，计算给出理想航向角 ang，以使船不断向理想航迹逼近。

4.2　Change 角度转换模块

该函数输入为当前航向角 $[0, 2\pi)$ 输出为当前航向角 $(-\pi, \pi]$，这样，输出的航向角就与函数 *atan2()* 的输出（$(-\pi, \pi]$）相同，在同一值域下无需再考虑其他角度的判断问题。而这样统一角度的作用还体现在使船模以锐角的形式转弯，仅用一个判断语句解决了转弯的问题，即判断理想航向角和实际航向角的夹角进行判断，具体请看我们的程序。

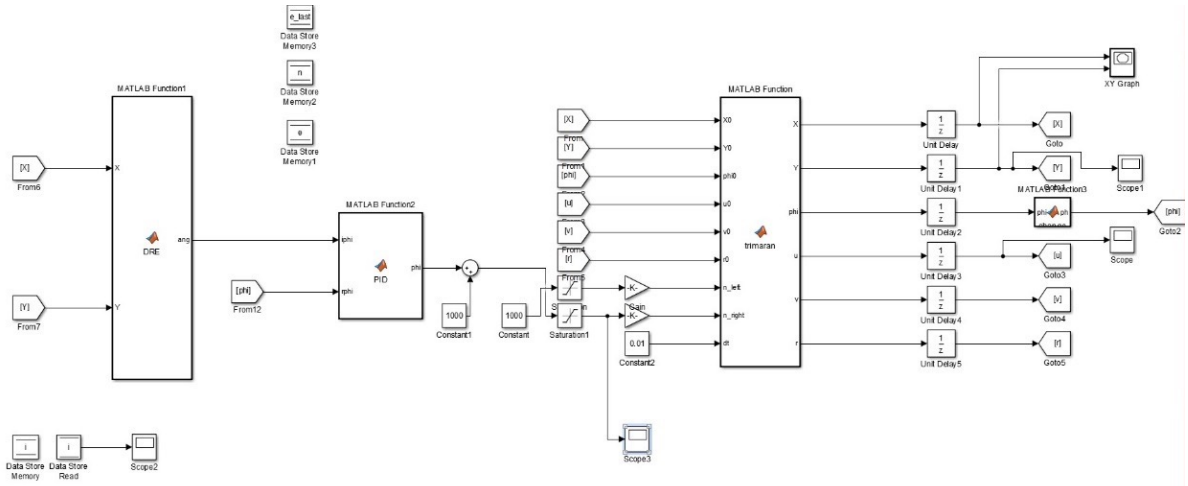
通过比例、积分、微分项对理想航向角和当前航向角进行误差处理。该函数输入为理想航向角 ang 和当前航向角（范围 $-\pi$ 到 π）输出为右桨的转速调整量。P、I、D 三参数依次选用此前得出的 1200、3、10。

4.3　Simulink 模型

图4–1为完整 Simulink 模型，DRE 和 PID 函数的输入均取自上一循环三体船模型给出的 X、Y、phi 值，通过固定左桨转速不变为 1000，右桨转速为 1000+phi，实现对船体航向的控制。

这里说明一下坐标系的问题，在 matlab 是模型船是按照实际的北东系坐标写的，因此，在最初时我们用笛卡尔坐标下写控制程序得到通过 PID 进行输出得到左桨的预想速度结果是错误的，因此我们改变思路，仍然在笛卡尔坐标系下写控制程序，但是，将输出的结果放在右桨上，因为我们认为，笛卡尔坐标系与北东坐标系唯一的区别就是一个是从“纸的正面看”，一个是从纸的背面看，所谓笛卡尔坐标系的左桨即为北东坐标系的右桨，这样 PID 输出到右桨巧妙地解决了笛卡尔坐标系和北东坐标系的转换，其中控制程序一样。结论即为，两种坐标系的转换就是同样控制程序下输出左右桨的转换

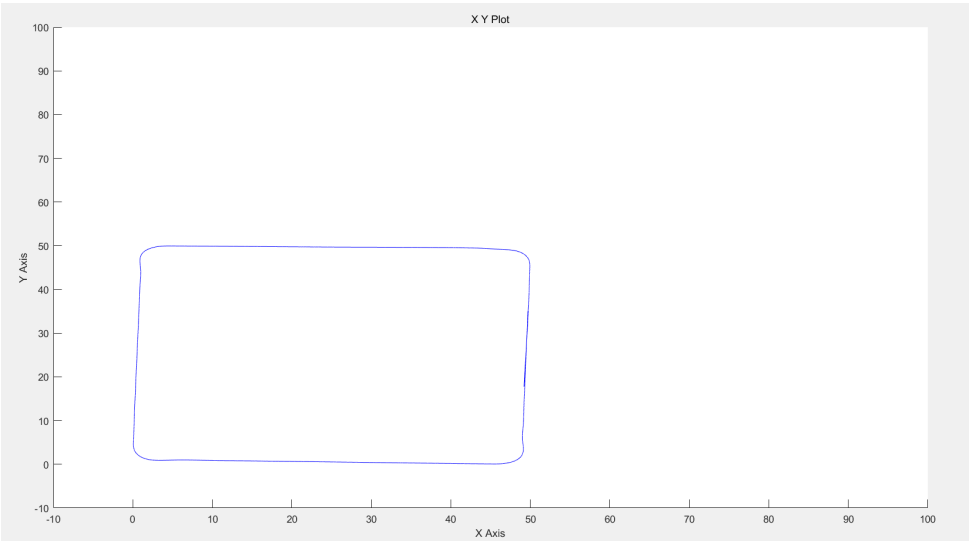
图 4–1



4.4　运行结果

图4–2为三等分法作为循迹算法运行后的仿真结果，可以看到除在转向时有少量过冲以外，模型基本按照预定轨迹航行。

图 4–2



第五章　个人总结——彭正皓

感谢许老师，感谢张学长、秦学长、邓学长的帮助。感谢学校给我们提供了这个宝贵的机会，在风和日丽的周五下午，在美丽的思源湖畔泛舟驾艇。

在这门课程中，与其说是开发循迹算法，不如说是实践软件工程。在许多个周五，我们都不得不在湖边干坐，等待其他同学解决三体船硬件软件上的种种问题。由此可见，搭建一个有效的，可靠的，鲁棒性强的软件系统是一个非常困难的挑战。我们一定要谨记在任何程序中，都要确保安全和可靠，基于此才能开发出更有趣味更有效率的程序。举个简单的例子，就是在任何程序中，都要确保在程序意外崩溃的情况下（又或者是由操作者手动结束程序时），程序干的最后一件事情是关闭电机，并恢复一切经更改的设置。举个例子，通讯信道经常在某些程序崩溃后仍然被占用着，导致船体无法接收后续的指令，这是极其危险的事情，我们也曾看到过几起这样的事故。所以说，大家一定要提高自己编程逻辑的周密度，一定要考虑各种各样的情况。

切记，无人船控制程序比普通的仅运行在电脑上的程序更加危险，因为你在控制的是一个真实的运动机械，它十分笨拙，且具有巨大的能量，极其危险。无论是三体船模，还是万吨巨轮。

Python 程序和这份报告主要由我操刀。

第六章　个人总结——刘子平

在无人船的实验课之中，我所做的工作主要分为以下方面：

在实验课的初期，我们初步打算采用 PID 控制方式，通过相应的循迹算法，给出无人船的理想航向角，将理想航向角和船的首向角作为 PID 控制的输入，输出的是舵角。然后对建立电机差速和舵角的数学关系式，根据舵角的值给出左右电机的实际转速。之所以采用这种方式是因为控制舵角的文献资料较多，有成熟的 kp、ki、kd 三个参数的值，对 PID 调参有所帮助。这个阶段我编写了 PID 的控制程序代码，采用的 PID 形式是增量型的 PID 形式。不过由于实验条件的原因，没有足够的数据支持我们建立较为精确的电机差速与舵角的数学关系式，我们最终更改为直接控制电机的转速。

之后过程之中，我的工作主要是寻迹算法代码的编写。其中我采用了两种不同的算法：第一是我自己想出来的中点法，即从船的当前位置想航线上做垂线，垂足与当前航迹段的终点的中点与船当前位置的连线即为船的理想航向。后来经彭正皓修改，我尝试了将中点更改三、四到十等分点，进行了算法优化。在彭正皓做的数值模拟平台上进行了该算法不同形式的数值模拟，最终确定使用六等分法。第二种算法是通过查阅文献得到的 LOG 算法，将中点法的中点改为在船当前位置做一半径为 R 的圆与航迹段的交点。同样的对 R 大小做了上述的数值模拟，最终确定了 R 等于 6。具体数值模拟结果见2–2

总体而言，这门课给予我接触无人船的机会，可以将自己课外接触到的与控制有关的理论知识应用到实践之中，希望有机会可以进一步的接触这方面的实验。同时也要感谢老师的指导以及集成组的同学搭建的平台。

第七章　个人总结——支鹏飞

这学期，我在许劲松老师以及各位学长的帮助下了解了无人船，用 matlab 实现了无人船的控制，在此非常感谢老师和学长的帮助，当然同时也非常感谢彭正皓和刘子平同学的建议。

学期最开始，我们组两位成员计划使用 Python 编写控制程序，而且两位同学已经接触过 PID 控制小车的内容，对于我而言，我从一无所知到如今的基本了解也非常感谢两位同学的帮助。经过与两位成员的交流，我逐渐建立起了对 PID 控制的感官上的了解，而两位成员使用的 Python 语言却从未接触过，虽然有较好的 C++ 语言的基础，但是受限于接口文件的难以编写，最终不得不放弃了用 C++ 编写控制程序的希望。

因此我最终按照老师最初的建议，用 matlab 写控制程序。接下来，我列出几个我认为比较有创新的点。

1. 在进行 matlab 编写时，遇到了很多问题，当然，循迹算法部分是通过学习小组的成员的等分点法的思想，在网上查找循迹算法的过程中，查到一篇《基于迭代学习理论的智能铲运机运动轨迹控制算法研究》（姜勇）的文章，这篇文章中讲到，进行确定 PID 的输入时，也就是循迹算法的前一部分，可以通过将位置偏差和转角偏差进行综合作为 PID 的误差的输入，因此，我进行了将两个误差相综合的误差作为 PID 的误差输入。
2. 由于整个程序以及 PID 的调整是通过重复运行程序，因此，我在 matlab 的 DRE 函数中引入全局变量，因此可以较简单的通过控制全局变量来控制是否到达指定点以及更换起始和目标点，这一点非常巧妙地解决了起始点和终止点的更新。
3. 我在 PID 函数中同样使用了全局变量，这样不只可以保存上一时刻的误差量，而且还可以实时更新误差总量，也就是积分项的实现。
4. 在实现 PID 控制前，我将船模实际的航向角转换为 $(-\pi, \pi]$ 的区间，这样做的好处在于可以和 atan2 函数的值域相吻合，极大的减小了程序复杂度。
5. 航向角转换的另一个优势就体现出来了，可以只通过一条语句来执行船以锐角的形式进行转向。我在这里贴出这条语句

```
 if rphi - iphi>pi || rphi - iphi <-pi
et=iphi-rphi;
else et=rphi-iphi;
end
```

它比较完美的实现了以锐角的形式进行转向。

以上就是我在 matlab 中进行实验的认为的主要的创新点和优点。

在进行 matlab 实验的过程中，感谢邓乃明学长的悉心帮助和指导，让我能够在错误中及时返航，最后写这篇报告时，模型还没有在实船上进行试验，但是在 matlab 中已经能够较好的完成所有的任务。

Matlab 的框架搭建和程序编写主要由我完成。

第八章　个人总结——胡经朝

本学期的船舶创新实验给了我们真正的操控船模的机会，这是个不小的挑战，不过在大家的努力下，我们最终的运行结果让人满意。非常感谢许老师给我们的指导与帮助、也感谢邓学长提供的大力支持和小组成员的给力协作。

在课程开始，由于我们刚接触无人船的自动控制，对硬件软件各方面都不是太了解，所以大家除了上课之余，通过课下不断与学长联系，上网查找资料来加快对整个无人船控制的了解，主要查询了包括现有无人船循迹算法、Veristand 使用手册、PID 控制等内容。后由于大家普遍对 python 较为熟悉、且为了实现更多的功能和获得更大的灵活性，我们选择以前人写好的 MsgDev 作为通信模块，从零开始开发了一套具有实时监控功能的自循迹无人船系统，同时基于 Matlab 的 Simulink 模型进行各种循迹算法的模拟仿真。这其中我们遇到了不少的困难，比如，在编写 Matlab 程序时，由于我们对 Matlab 的 Simulink 不太熟悉，我们无法对已赋初值的变量进行刷新，一开始我们使用了 Constant 模块，但效果不理想，最后用了全局变量模块才解决了问题；又比如由于对 PID 模块不熟悉，我们直接拖用了 Simulink 自带的 PID 模块，但始终无法解决问题，仿真的结果总显示船在不断地做圆周运动，或者直线运动，最后重写了 PID 函数才解决问题；诸如此类的 debug 问题不胜枚举。

当然在课时实船测试时，也遇到了很多软件硬件上的问题，比如，坐标系找点困难，不知道原点具体位置，于是我手动把船开到湖上的四点，记录下坐标，再输入到程序中，我们才第一次完整的完成了循迹。之后还有包括 Veristand 端口占用等软件问题，网线接触不良等硬件问题，通过协调也都一一解决。

在这门课的学习过程中，我们共同探究了循迹算法的诸多问题，一开始我们采用刘子平提出的中点法循迹，从感觉上来说这是一个完全可以实现循迹功能的算法，后来我们想到也可以通过三等分法、四等分法等来寻求一个最佳的逼近方法，通过求偏差累计来评估各个算法之间的优劣，后来有组员提出用以当前位置为圆心画圆与预定轨迹的交点作为前进方向，从原理上更为优秀，我们也参考了许老师的论文《内河无人船局部路径规划和循迹控制》中提到的 LOS（line of sight）视线法作为备选方案，因中点法和画圆法表现较为出色，我们后来没有尝试 LOS 法。

通过本课程，我最大的感受是做一个相对复杂的工程时，要把每个部分都做的尽善尽美、面面俱到是非常困难的，总会有各种各样的问题，需要用耐心和信心去面对；另外要选择好自己熟悉的工具，即所谓工欲善其事必先利其器，只有对软件、语言等有了熟练的掌握才能在需要的时候发挥出作用。

在本课程中，我主要做的工作有：循迹算法文献查询、参与 Matlab 程序编写，第四章 Matlab 模型报告起草。

第九章　个人总结——刘彬杰

本学期的船舶创新实验课为我们提供了一个实际上手操控无人船的机会，可以说相当难得。在此，要感谢许劲松老师，秦操学长，邓乃铭学长的悉心指导与支持，同时也要感谢集成组为我们搭建环境辛苦的付出，以及与另外一个三体船组的交流让我们发现问题，共同探索。

由于大家对于无人船控制的了解几乎都是从零开始，课程前期，我们查阅了不少文献作为准备工作，了解到许劲松老师、瞿洋、徐海祥等所作论文中较为主流的 LOS 循迹算法思路；同时，对于 Matlab、Simulink、Veristand 的基本使用方法也在自学跟进和请教学长；由于对 Python 熟悉度更高，最后决定主要通过 python 进行开发。

循迹算法模块，我们基于 LOS 循迹算法思想，先采用了刘子平提出的中点逼近法，并在 Python 取得了不错的模拟结果；其后在此基础上，我们共同探究了三等分法、四等分法等进行系列调整，以获取最优算法，进一步完善；等分法取得较为满意的结果后，又提出 LOG 算法，两者进行了对比并分别在实船进行了验证，效果均较理想。

在对 Matlab 的上手过程中，因为缺乏经验，我们遇到了不少问题，对于 Simulink 的模块搭建也知之甚少。具体地，我们用 Simulink 中自带的 PID 时出现问题，在自行编写 PID 后得以解决；航向角范围问题也困扰我们许久，因此对于 phi 的输出值分情况进行了讨论，诸如此类的小问题还有不少。但总的来说，通过慢慢摸索，我们用 Matlab 初步实现了循迹控制。

实船实验中我们也不可避免地遇到了问题，包括坐标系问题（北东系与正常坐标系的转换问题，角度范围问题，通过分情况讨论改进）、程序模型与实船的参数不统一（实船左桨转速在程序中并非正数，这在观察到实船原地打转现象后发现，并进行调整）、实船操作的过冲量明显大于模拟时过冲（通过对循迹算法的调整进行优化）；另外，在实验过程中我们也多次遇到实验环境出现问题。

由此我也得到了一些启发：1. 对实验现象的仔细观察是很重要的，因为它可能揭示出存在的问题，并让我们根据实验现象进行改进；2. 使用软件和工具的熟练度会很大程度上决定我们的进程，因此在面对新的软件时，我们需要多下功夫，所谓“磨刀不误砍柴工”确有道理；3. 硬件设备、网络环境等支持对于工程开发的效率而言非常重要，因为虽然集成组很辛苦，但也有很多周我们不得不因集成组出现的故障而放弃跑船，这使得进行实验验证的时间缩减了不少，开发效率大打折扣。

本次实验课程中，我做的工作主要有：参与 Matlab 程序的共同编写、前期 Simulink 仿真初步探究实验变量关系、进行以轨迹偏差为输入的 PID 算法探究（但未能成功，实际采用的是支鹏飞编写的以转角偏差为输入的 PID）以及文献查阅。

第十章　个人总结——宋深科

在本学期的无人船实验课中，在老师的指导下、助教以及集成组同学的帮助下，虽然过程不易，但我们小组最终还是实现了三体船的自动循迹控制。当然，这很大程度上要多谢我那些优秀的队友，而经过这一学期对无人船相关软件、硬件的接触，我也受益良多。

最先，在我们了解了三体船的控制机制之后，准备利用 PID 算法控制舵角，再由舵角的值来确定螺旋桨差速，不过遗憾的是，即使刘子平和彭正皓迅速写好了 PID 算法，但我和刘彬杰没能得出舵角和转速的关系。我们本设想利用学长给出的 MATLAB 船模型通过模拟来确定这一关系，操作后发现过于复杂，且实际中船舶会受到各种干扰，和其他组员沟通后，我们放弃了这一想法。

而后，我们决定直接让 PID 利用航向角去控制转速，并且最终较好的实现了这一想法。而由于对 Python 语言不甚了解，在这一过程中，我主要做的工作就是协助组员的实验，手动控制三体船，帮忙确定各点位置等。同时，我们还分头各自去查阅文献，希望能集思广益，找到更好的循迹算法，最终，我们选用了 Log 算法和等分点算法。

最后，在小组实船实验得到较好结果后，由于 Python 算法参与较少，我和支鹏飞、刘彬杰、胡经朝便尝试在 MATLAB 平台上实现我们小组已确认可行度较高的循迹算法。于是我们以学长给出的 MATLAB 船模型为基础，遵循由船位置和理想航线得出航向角、让 PID 算法利用角度控制转速这一思想，经过我们的通力合作，最终在 Simulink 里实现了这一过程的模拟，并得到了较好的结果。在最后，我曾试图将 MATLAB 和 Veristand 连接，却因为软件版本问题而作罢。希望我们最终能借助张宇翔组的电脑，实现对实船的控制，那样的话我们的 MATLAB 工作便宣告圆满了。

经过这一学期在无人船课程上的学习，我对整个自动控制的的过程有了很好的接触，而老师的很多对控制、对学习的观点，也对我有很大启发。另外，得益于助教和组员的帮助，我对 Python 和 MATLAB 都有了初步了解，也亲身经历了控制系统软硬件结合的过程，对我个人日后的成长很有益处。感谢老师一学期的辛勤指导，感谢助教、组员、集成组同学们一学期对我的帮助，这一学期思源湖边的聚会，让我获益匪浅，希望日后还有机会能和老师、同学一起学习、一起进步。

图 10–1 矩形

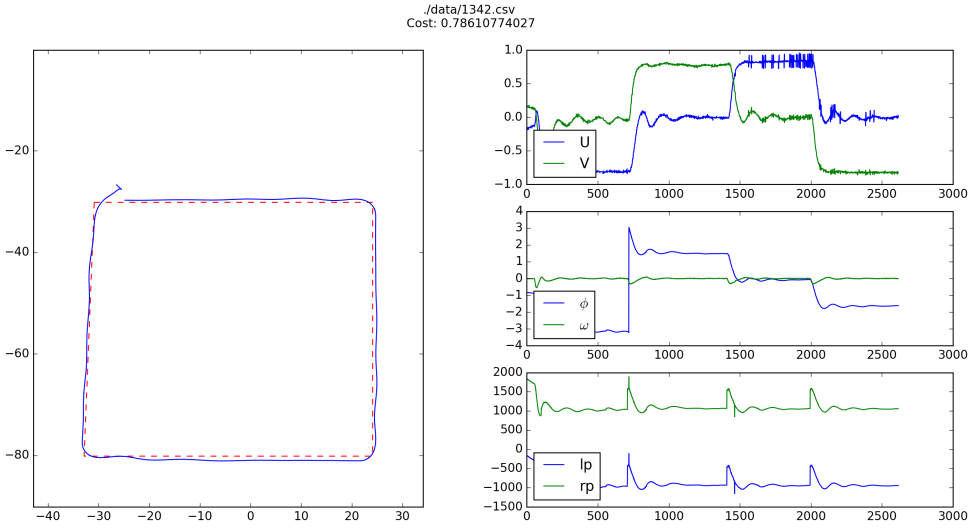
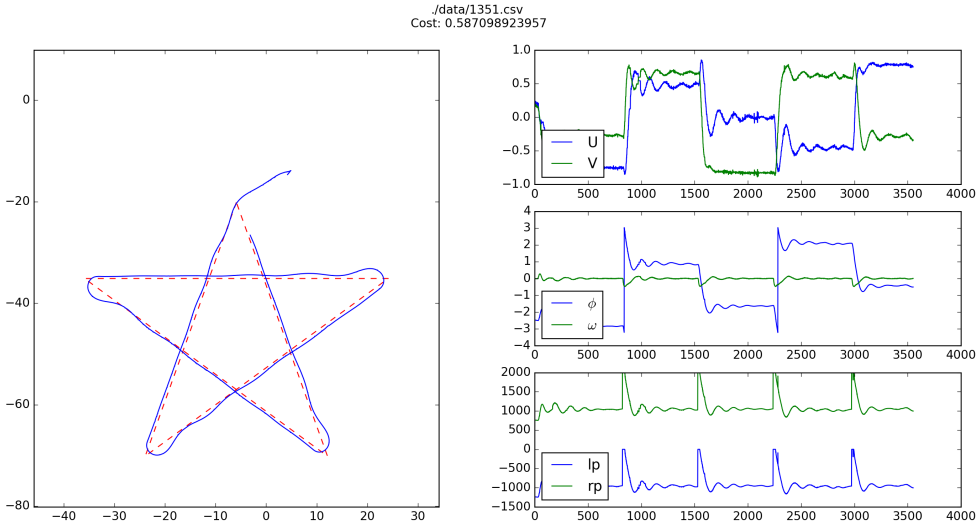


图 10–2 五角星形



第十一章　最终结果与致谢

这个学期，我们成功地完成了三体船的循迹与控制实验，自主完成了所有规划和控制算法，开发了一套三体船仿真实验平台，尝试了各类循迹算法，从无到有不借助任何现成软件实现了三体船航行状态的实时监控和控制。

2017 年 12 月 22 日，我们完成了最后的测试。首先，我们尝试了令三体船走一个矩形。之后，为了测试无人船锐角转弯的性能，我们尝试令起走了一个五角星形。实验的最终结果被展示在了图10–1与图10–2中。可以看到，船舶基本按照规定的航线行驶，在转弯的时候，也取得了非常小的过冲，左右螺旋桨的转速基本平稳，没有明显的高频波动，船速稳定在 $1m/s$ 以内，且没有明显的波动。因此，我们取得了非常完美的实验结果。

在本学期的无人船自动控制之旅中，我们要大力感谢许劲松教授对我们的指导，张磊学长提供的 MsgDev 库及许多帮助，秦操学长提供的三体船动力学模型，邓乃铭学长提供的帮助。