文章编号:1671-4598(2018)05-0136-04 DOI:10.16526/j.cnki.11—4762/tp.2018.05.034 中图分类号:V279 文献标识码:A

基于探测目标的自适应航路规划算法产品化实现方法

罗喜霜,宋 亮,郑 亮,雷 玮
(北京经纬恒润科技有限公司,北京 100191)

摘要:为将基于探测目标的自适应航路规划算法进行产品化,采用基于模型的设计方法以及自动代码生成技术,使算法模型快速、健壮地转化为软件产品;软件产品由核心算法和外设驱动组成,核心算法采用基于模型设计思想进行设计,仿真验证后自动生成目标语言代码,考虑到IO访问方式的复杂性,外设驱动模块采用代码自动生成与手动编码相结合的方式实现,核心代码与外设驱动模块组成的完整工程下载到硬件板卡,通过试验验证软件产品与初始设计模型的一致性,实现算法的产品化;经过实践,所提出的算法产品化实现方法能够提高设算法设计效率和各阶段中间产品继承性,降低各阶段调试复杂性,实现产品开发的敏捷性和健壮性,可作为复杂算法工程实现的参考。

关键词:代码生成;构建工程;编译下载;算法调试;算法产品化

Target—based Adaptive Guidance Algorithm Production Method

Luo Xishuang, Song Liang, Zheng Liang, Lei Wei
(Hirain Technology, Beijing 100191, China)

Abstract: In order to make production of the target—based adaptive guidance algorithm for extensive use, model—based design method and code auto—generation method are used to transform the algorithm into software production. The software production is made up of core algorithm and peripheral driver. The core algorithm is designed by model—based design method and is transformed into production code by code auto—generation method, and the peripheral driver is realized by hand—writing code because of its complexity. The core algorithm and peripheral driver are downloaded together to the hardware circuit board to realize production. Ground tests are taken to verify the consistency of the algorithm model and production code. And the test results show that this algorithm production method can improve the algorithm design efficiency and succession, reduce the design complexity, make the algorithm production process quickly and flexibly and it can be used for reference for other complex algorithm production process.

Keywords: code generation; construct project; building and download; algorithm debug; algorithm production

0 引言

航路规划算法和飞行控制算法,是飞行器能够正确执行特定任务的关键,特别的,对于无人飞行器,高度自主的航路规划能力始终是领域研究的热点,且受到主要军事组织的重视[1]。在城市低空空域应急管控系统课题研究中,涉及到目标发现、目标锁定、目标跟踪等相关算法的设计和实现。基于探测目标的自适应航路规划算法研究是城市低空空域应急管控系统课题研究中的关键子课题之一,需要对算法进行设计,仿真和验证,并形成产品用于应急管控系统,同时将其推广应用到其他相似领域中。

近年来,由于Matlab/Simulink等仿真环境和设计工具的不断完善,部分研究机构和型号单位开始使用其进行飞控系统核心算法软件的设计开发。然而,使用方式仍然没有跳出传统的基于文档的设计开发范式,算法模型经过Simulink仿真验证后,仍采用手动编码的方式进行目标代码编写,效率较低且易引入人为错误,进而导致较长的研发周期[2]。显然,减少甚至完全取消手动编码将提高研发效率,而Simulink的代码自动生成功能使之成为可能。目前,高校所进行的研究中,利用Simulink实现模型自动生成目标代码的案例较多[3-4],应用领域广,但限于客观条件大多停留在课题研究层面。

本文以基于探测目标的自适应航路规划算法为目标,结合基于模型设计的开发范式,充分利用Simulink自动代码生成特性,最大程度压缩合并开发过程中的重复工作,提出一种针对算法产品化的方法。本文对产品化过程和方法进行描述,主要包括:基于模型的算法设计及仿真验证,算法的自动代码生成,自动生成代码与特殊手写代码的融合,算法代码调试工程的构建和调试环境部署,代码的编译下载和调试,代码烧写,以及试验验证等。

1 基于模型的算法设计与仿真验证

算法的设计结果与仿真验证是产品化的基础,能够体现产品功能性能范围。算法设计和算法实现,是两个重要的环节。算法设计是为了确保算法的功能要求得到满足,算法实现是为了确保算法的工程应用。在算法设计阶段,需要考虑后续算法实现的方便。采用基于模型的算法设计和仿真验证方法,可以满足上述要求。

基于探测目标的自适应航路规划算法为飞行器载体提供目标探测和跟踪指令,因此采用基于模型的算法设计和仿真验证方法进行设计时,需要构建仿真辅助模型,包括:目标和飞行器本体模型、飞行器地面站模型、飞行器控制模型以及相关的

收稿日期:2017-08-21; **修回日期**:2017-10-25。
基金项目:北京市科技计划项目基金究(D161100005816002)。
作者简介:罗喜霜(1976-),女,博士,广西马山县人,高级工程师,主要从事飞行制导导航控制方向的研究。

算法模型，此外还需要构建仿真环境模型，如飞行器飞行所处的位置环境模型和障碍物模型等。

采用 Matlab/Simulink 仿真建模软件，利用其丰富的设计工具，可以方便实现上述模型的构建和仿真，提高设计与验证的效率。同时 Matlab/Simulink 具备成熟健壮的自动代码生成工具，使得所提产品化方法得以实现。

图 1 为基于探测目标的自适应航路规划算法设计仿真模型的组成，包括：自适应航路规划模型 Programmer，实现航路规划计算；自动飞控模型 AutoControl，实现主飞控逻辑及控制律解算；通信延迟模拟模块，模拟传感器和控制器间数据传输的延迟和抖动特性；地面站模型 GroundStationSend，模拟地面站指令和状态显示；飞行器本体模型 PlaneModel，模拟飞行器动力学、机载传感器和作动器。

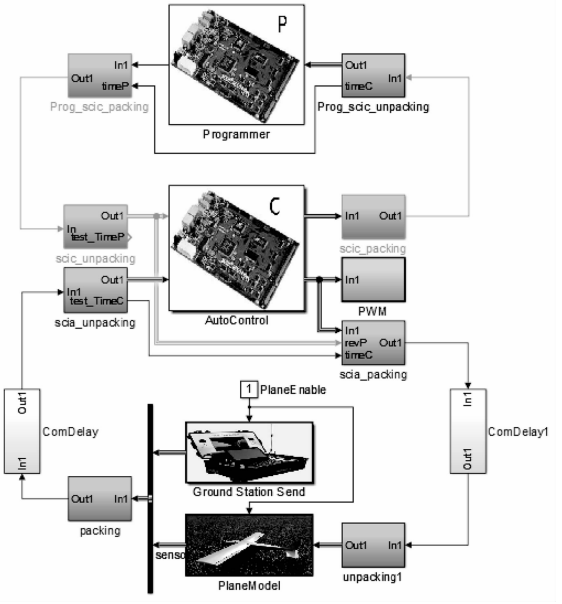

图 1　基于探测目标的自适应航路规划算法设计仿真模型

构建基于探测目标的自适应航路规划算法设计仿真模型时，为满足后续代码实现的要求，需遵循如下建模要点：

1）模型模块划分要充分考虑后续实际系统应用时的接口划分，同一接口发送的数据使用一条数据总线表示，进行通信接收/发送端模块的连接；

2）根据计算精度和硬件板卡处理能力确定算法模型中所有信号的类型，Matlab/Simulink 默认数据类型为 double，建模时根据预先分配的类型对信号进行更改，不同硬件板卡中数据类型的定义可能不同，需根据具体情况进行信号类型的设置，如对于 DSP C28346，由于最小数据长度为 2 个字节，因此 char、uint8 和 int8 均为 2 两个字节，与 Matlab/Simulink 中的定义不同，因此为了保证算法结果的一致性，建模时不能使用 char 等数据类型；

3）模型中要合理考虑实际系统的误差影响因素，例如考虑系统接口之间的信号传输延迟、信号波动等，在上述模型中，对于需要 RS232 等 IO 接口传输的信号，在相应信号的接收端和发送端之间加入通信延迟模拟模块，使信号延迟节拍按周期性变化；

4）构建模型时，需要区分离散状态和连续状态，下载到 DSP、ARM 等目标机中的核心功能代码不能包含连续变量，因此需要将代码中的连续变量模块用相应的离散模块替换，如 PID 控制器中的微分模块和积分模块、状态空间模块等，辅助仿真模型不需要下载到硬件板卡中，可以充分利用标准工具箱中的连续模块，简化建模过程，如航空宇航工具箱中的动力学方程模块等；

5）根据功能需求和实现特性，综合使用 Matlab/Simulink 模块和代码进行模型搭建，例如，自适应航路规划算法模型采用 Embedded Matlab 模块实现，既满足了手写代码的自然语言方便性，又结合了 Simulink 模块建模的可读性，对于控制逻辑部分则采用 Stateflow 和 Simulink 事件模块实现，提高系统调用逻辑的层次性，便于升级维护。

在 Matlab/Simulink 下完成基于探测目标的自适应航路规划算法设计仿真模型后，可以开展相关数字仿真试验，根据实际的飞行模式要求进行对应仿真场景设置，记录仿真数据，对仿真结果进行分析，在确认仿真结果满足要求后，即可对航路规划算法模块和飞行控制算法模块进行代码生成。

2　算法的代码生成

2.1　代码生成模块的构建

基于探测目标的自适应航路规划算法设计仿真模型，是一个完整的闭环仿真模型，用于进行数字仿真验证。需下载到硬件板卡中的是设计仿真模型中的核心算法模型，需将该部分模型提取出来形成独立的新模型，并增加外设驱动等 IO 通信相关模块，形成核心算法代码生成模型，具体包括以下几方面内容。

1）增加通信 IO 驱动模块。根据不同硬件板卡选择对应的驱动模块库，模块库通过 Matlab supportPackageInstaller 函数进行在线安装，目前 DSP、ARM、Arduino 等主流嵌入式板卡均在支持范围内。根据通信接口形式选择对应的 IO 驱动模块，如，航路规划模块使用一路 RS232 进行信号收发，则将串口接收、发送、中断触发模块加入模型中与算法模块连接，并正确设置中断号和接收字节数等信息，如图 2 所示。

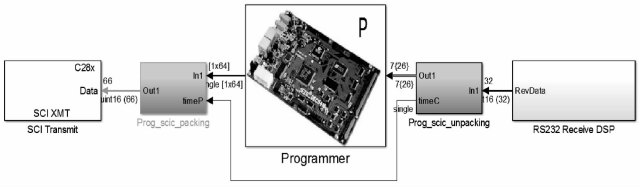

图 2　航路规划算法对应的代码生成模块

2）按照通信协议进行数据打包解包模块的搭建。在 Matlab/Simulink 仿真环境下，提供了与最终代码运行的硬件平台如 DSP 相对应的数据打包模块和数据解包模块，可以直接使用，在存在特殊协议情况下，通用的数据打包解包模块不能满足要求时，可以通过手写 C 代码，以 S－function 模块实现，与通用的数据打包解包模块联合使用，即可解决特殊应用问题。

2.2　代码生成环境配置及代码生成

在完成核心算法代码生成模型构建后，需进行代码生成环境的配置。代码生成模型最终生成的代码是可以运行在硬件平台上的目标代码，例如，运行在 DSP 上的 C 代码等。由图 1 中可见，算法仿真模型在设计仿真环境下是以图形化方式表

示，转换成目标代码后，算法模型将变成为 C 代码文件。算法模型中存在需调整的参数，这些参数在生成的 C 代码文件中是否方便查找和修改，取决于进行代码生成前的准备工作，包括以下几个方面。

1）将可能需要调试的模块，如航路点计算模块，设置为原子子系统，提高可读性，进而提高调试效率。如图 3 所示，若未将该模块设置为原子子系统，则该模块生成的代码将与模型中其他模块生成的代码按顺序在主函数中排列，调试不便；设为原子子系统后，根据所需模式，可以使该模块生成独立函数文件或生成为主函数中的子函数，便于调试阶段进行故障定位与变量监控。

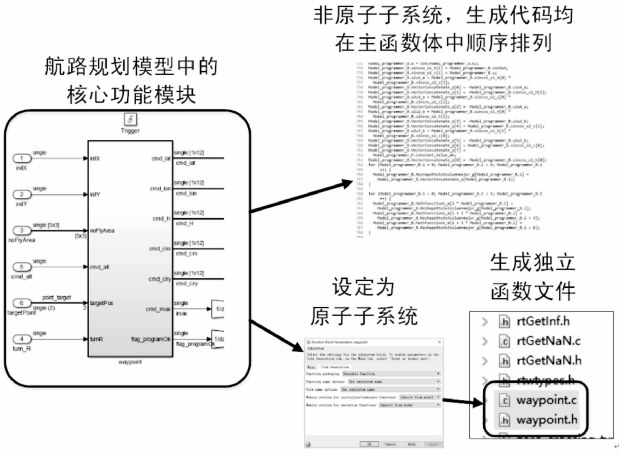

图 3 将需调整参数的模块设置为原子子系统

2）根据 Matlab/Embedded Coder 说明文档，安装代码自动生成所需的工具，以 DSP C28346 为例，工具包括：编译器环境 CCSvx、C2000 代码生成工具 CGT、DSP/BIOS、仿真器驱动 XDC Tools、目标机头文件。工具环境安装完成后，在待下载的 Simulink 模型中进行模型层面的配置，如图 4 所示，设置前一步安装的编译环境，硬件板卡型号，特别的，需在外设选项卡中对所使用的 IO 接口参数进行设置，如本模型使用 RS232 通信，因此需对 IO 通道、奇偶位、数据位、波特率、IO 管脚等关键参数进行设置。

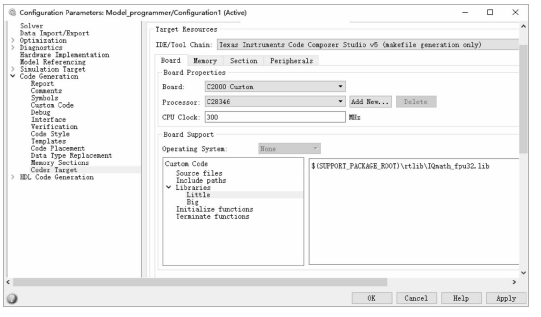

图 4 代码生成参数配置界面

在完成代码生成参数配置后，点击模型的 Build 按钮，即可生成所需的目标代码，包括 .c、.h、引用库文件和汇编文件等。这些文件将在编译器环境 CCS 中使用。

2.3 特殊情况手写代码的集成

从图 2 中可以看出，航路规划算法对应的代码生成模型，主要包括 RS232 串口收发驱动模块、数据打包解包模块和核心算法模块，在自动生成的算法代码中实现对应功能。在实际工程应用中，同时需要对算法解算中间过程进行数据记录，因此还需要实现数据存储功能。

当所需存储数据量较大时，宜使用大容量 SD 卡。但一般而言，第三方硬件板卡驱动库并不提供 SD 卡读写模块，因此在构建航路规划算法对应的代码生成模块时，需要手动编码，且涉及硬件操作的驱动模块一般仅能用 C 等低级语言编写而不能通过 Simulink 建模实现，目前有两种可行的方法：

1）在 Simulink 仿真建模环境下，通过手写 C 代码 SD 卡读写函数，封装成 S－function 模块，集成到算法的代码生成模型中共同进行代码生成，该方法所需的前期工作量较大，但后期使用方便，不需要对生成的 C 代码工程进行修改，后期使用工作量小且可避免引入人为错误；

2）在所生成的目标代码文件中，找到代码执行主入口函数，在合适位置手动编码 SD 卡的初始化函数和实际读写函数，此方法便于利用 SD 卡读写示例代码，可以快速实现 SD 操作功能，但后期使用过程中，一旦更新模型其他功能，需重复地手动增加 SD 卡读写相关代码，较为繁琐且可能引入人为错误。

3 构建算法代码调试工程

通过自动代码生成与手写代码相结合的方式，对航路规划算法模型以及飞行控制算法模型进行代码实现，并将代码烧写到硬件板卡中形成最终的嵌入式产品。有两种方式进行代码烧写，以 DSP 为例进行说明：

1）完全使用 Matlab/Simulink 环境而不直接使用 CCS 编译环境，可以将模型烧写到硬件板卡中，这种方法操作简单，但由于脱离 CCS，因此不能在 CCS 编译环境下进行手写代码的嵌入操作，使用受限；

2）在 Matlab/Simulink 中生成算法模型的目标代码，然后拷贝到 CCS 编译环境中并结合 CCS 操作实现代码烧写，该方法过程较前述方法略为繁琐，但由于可以方便地在 CCS 编译环境中进行代码修改、增加、删除，使用范围更广。

本节以第二种方式为例进行介绍。代码烧写完成后，需要对代码实现的正确性进行验证，因此需要借助硬件板卡的仿真器环境，进行代码的调试和检验。

3.1 代码调试工程的构建及调试环境配置

目前，若在 Matlab/Simulink 配置环境中选择 CCS 4.0 以上版本作为编译环境，则不支持生成可由 CCS 编译环境直接使用的目标代码工程。需要进行手动配置。利用 CCS 构建代码调试环境过程如下：新建工程，添加所需的头文件、源文件、库函数和相关路径，尤其需要注意的是要添加 SD 卡读写所需的 FATFS 文件系统，之后将 Matlab 自动代码生产的工程文件复制到该新建工程中，设置自动代码生产的 main 函数为主函数，最后将手写的 SD 卡读写函数嵌入到自动生产的代码中。

在 CCS 编译环境中进行代码调试如图 5 所示。

代码调试工程操作界面中，工具栏上提供了相关的操作图标，点击对应的图标即可进行相应的操作，包括编译、调试等。

代码调试工程操作界面中的左侧树状导航图分类显示整个调试工程所包含的所有文件，选择其中一个文件，则对应该文

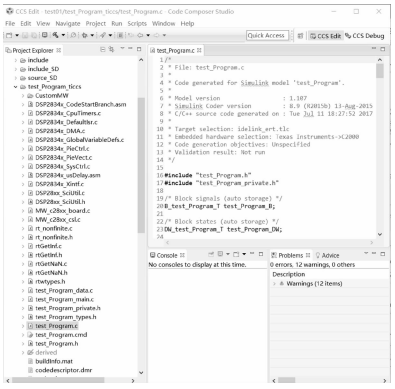

图 5　代码调试工程操作界面

件的内容在操作界面右侧区域中显示。

3.2　代码编译下载及调试

在完成调试工程构建后，选择 CCS 工具提供的代码编译功能，进行代码编译下载。

首先进行 Build 操作，编译工程，排除工程的环境配置问题后生成 .out 文件，之后对工程进行 Debug 操作，在 Debug 界面调试程序和验证算法是否正确。在调试过程中可以综合使用断点、变量数值监控和变量曲线监控等调试手段。在调试过程中，由于遵循 2.2.1 节所提出的原子子系统设置规范，可以清晰地确定原设计模型和所生成代码间的对应关系，提高问题定位效率。

3.3　代码产品固化

代码调试检验后，需将其固化到硬件板卡的 Flash 中，固化为嵌入式产品。固化之后，硬件板卡重新启动后可以自动执行写入的算法代码。

为方便产品代码的烧写，本文新建烧写工程，将代码调试工程和代码烧写工程分开操作。代码调试完成后编译生成 .out 文件，将该 .out 文件使用自带的代码转换程序生成十六进制文件，在烧写工程中转为 c 代码和 DSP 可执行文件。烧写工程中不需要专门配置代码下载至 Flash 的位置，只需将调试使用的 .cmd 文件复制到烧写工程中，其中已经加入了固定的程序偏移地址，自动将程序烧写进 Flash 中的指定位置。

4　试验验证

为验证基于探测目标的自适应航路规划算法产品化实现过程的正确性，需要对比产品在算法数字仿真环境下的仿真结果和产品化后硬件在环仿真结果的一致性。其必要性在于：虽然由模型自动生成的目标代码不会引入人为编码错误，但生成代码的使用环境和模型仿真的环境是不同的，例如，模型仿真时仿真数据不经过真实 IO 接口，因此所得结果不能反映 IO 接口接入后的影响。此外，一些不可预知因素也可能对结果造成影响，故而必须进行试验验证。

4.1　验证准则

验证准则因产品而异，一般根据产品的性能指标进行验证即可。如果没有明确性能指标要求，则可通过定性的方法检验产品是否符合设计，进而确认是符合需求。

基于探测目标的自适应航路规划算法产品，对高度保持精度、三轴姿态控制精度有明确的指标要求，同时，定性地要求自适应航路规划结果能够规避所有障碍区域。因此使用如下验证准则：

1）高度、三轴姿态校验：根据数字仿真阶段设定的仿真场景，分别进行模型仿真和产品硬件在环仿真，两者给出的高度、三轴姿态误差小于 20%，且满足产品规定的控制精度要求，则验证通过；

2）自适应航路规划校验：根据数字仿真阶段设定的仿真场景，分别进行模型仿真和产品硬件在环仿真，所得航路点趋势一致，且成功规避所有障碍区域，则验证通过。

4.2　算法产品化后的仿真结果

将图 1 所示设计仿真模型中的航路规划算法模型和飞行控制算法模型提取出来，结合外设驱动等模块形成代码生成模型，进行代码生成，烧写到硬件板卡中，形成产品，其他仿真辅助模型如飞行器本体模型、地面站模型等下载到通用实时仿真系统 HiGale[5] 中，按照图 1 的信号连接关系，使用串口线缆进行仿真机 HiGale 以及硬件产品的通信连接，开展硬件在环仿真，系统组成如图 6 所示。

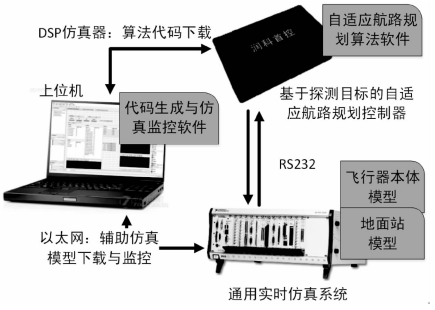

图 6　硬件在环系统框图

硬件在环仿真过程中产生的数据通过 Higale 通用实时仿真系统在线实时存储，所存数据通过以太网直接保存到上位机中，以便于上位机中的模型仿真数据进行直接对比。

图 7 给出模型仿真和硬件在环仿真所得的自适应航路规划结果，在规划航线上两者误差在 0.5 m 范围内，且所规划出的航线均成功规避障碍区域，验证通过。由于串口通信延迟抖动的影响，高度保持误差在 6 m 以内，趋势滞后约 0.3 s，三轴姿态控制结果误差在 0.2°以内，趋势滞后 0.05 s，均满足验证准则，限于篇幅不再给出图示。

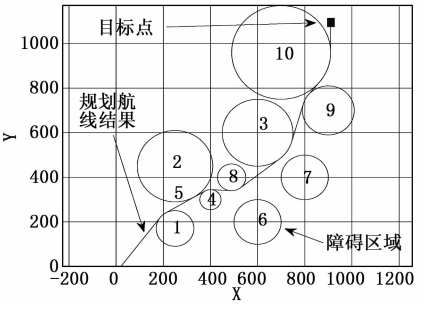

图 7　自适应航路规划结果对比

5　结论

基于探测目标的自适应航路规划算法产品化实现过程，采用了自动代码生成和手写代码相结合的方式，快速实现了算法的工程化。由试验结果可见，算法实现结果正确，过程高效。

（下转第 143 页）