Rapid Prototyping of Aero-engine Complex Control Method

LU Jun，GUO Ying-qing，WANG Bin-zheng
School of Engine and Energy，Northwestern Polytechnical University
No.188 POBox, Northwestern Polytechnical University, Xi'an, ShaanXi Province 710072, China

Keywords: rapid prototyping, complex control system, MATLAB/RTW, hardware-in-the-loop

Abstract

This paper presents an approach of complex control method (CCM) real-time simulation and rapid prototyping for aero-engine control system and describes its principle and realization in detail. This approach is mainly based on MATLAB/RTW for rapid prototyping from system modeling to embedded implementation. According to the simulation results between automatic code and manual code for an aero-engine multi-variable control method, it shows that this approach is feasible and effective, and not only decreases development cycle but also improves the reliability and universality. So a series of problems can be resolved during the simulation stage and rapid application to prototype testing.

1. Introduction

In recent years, with the ceaseless development of micro-eletronics technology, full authority digital engine control(FADEC) method has become an undoubted trend in the field of aero-engine control system all over the world. Because of the high complexity of aero-engine's structure and the high requirement of its performance, it makes related control method more complex than ones before, especially in multi-variable control, fault diagnose, etc. For the design and simulation of CCM, advanced simulation software Matlab/Simulink is mainly used in the control field. After simulated and verified via the digitial off-line state, CCM is transformed into application program running in Electronic Control Unit(ECU) and is simulated via hardware-in-the-loop, which is one of the key processes in the aero-engine control system development.

However, there are many problems as follow in the process of traditional simulation methods extensively applied: (1) During the modeling and simulation of the highly complex aero-engine, how to make use of the validated model realized in other ways; (2) During the transition from digital off-line simulation to HIL real-time simulation, the way of writing code is inefficient and the error-prone of code is notable; (3) During designing CCM from the beginning to the end, it maybe result to unnecessary faults and increase the development cycle because of independent processes (e.g. it is possible to happen that the code must be rewritten in the transition from simulation to prototype testing). Apparently, the development of aero-engine CCM is restricted by these problems.

Although modeling and simulation of complex dynamic system can be completed in Simulink in many conditions, this package can't usually achieve the request for highly complex dynamic system (e.g. the aircraft's engine). But this package provides us with an interface method called S-Function which allows you to add your own blocks in other languages (C, C++, Fortran, Ada, etc) to Simulink models. More important thing is that MATLAB possesses an automatic code generator--real-time workshop (RTW)[1] which can generate optimized, transplantable, and customize code from Simulink models on a variety of systems, thus freeing the engineers from the tedious and error-prone task of writing code for a given CCM. In this way, the aforementioned problems can be perfectly resolved.

Furthermore, according to the request of real-time simulation and prototype testing, an outstanding RTOS (e.g. Vxworks, QNX, LynxOS, pSOSystem, etc) is necessary, especially digitial eletronic engine control(DEEC) system running in the embedded device. Contrasting with other RTOSs in the aspects of reliability, real-time, performance, and integrated develop environment (IDE), Vxworks is more advantaged[2]. It is extensively applied in the field of communication, military system, aviation, space, etc.

Therefore, this paper presents an approach of CCM real-time simulation and rapid prototyping for aero-engine control system and describes its principle and realization in detail. This approach is mainly based on MATLAB/RTW for rapid prototyping from system modeling to embedded implementation. According to the simulation results between automatic code and manual code for an aero-engine multi-variable control method, it shows that this approach is feasible and effective, and not only decreases development cycle but also improves the reliability and universality. So a series of problems can be resolved during the simulation stage and rapid application to prototype testing.

2. Nomenclature

H = height
M_a = mach number of aircraft
W_{FM} = fuel flow rate to main burner
W_{FA} = fuel flow rate to afterburner
A_8 = main nozzle throat area

3. Rapid Prototyping

A typical DEEC development involves the following stages:

(1) Aero-engine modeling and validation;

(2) Control method design;

(3) Control system simulation and tuning;
(4) Embedded implementation;
(5) Prototype testing.

One of the main objectives in designing DEEC is to reduce the development cycle in going from step (1) to step (5). Engineers are thus looking for a prototyping scheme [4~8] which results in the shortest possible time in going from the initial concept to the embedded realization of DEEC. So the rest of works include confirming performance standard, designing CCM, choosing actual hardware, setting sampling period, mastering development tools, etc. And these works should be focused on for engineers, especially the part of designing CCM. Furthermore,the approach proposed in this paper can shorten the development cycle in rapid prototyping.

The rapid prototyping approach proposed in this paper based on MATLAB/RTW involves the following simulation flows: (1) creating the model of aero-engine in Simulink or obtaining it via S-Function interface technology; (2) designing CCM in some tool packages, such as system identification toolbox, robust control toolbox, optimization toolbox, stateflow, etc; (3) simulating the whole system in off-line; (4) building up a HIL simulation platform and separating control system from the whole system; (5) generating the embedded code for ECU via RTW; (6) simulating it in the HIL real-time simulation platform. Finally, aero-engine CCM can be verified and validated through ceaselessly debugging in a short time.

4. Realization

In this paper, the realization processes and key works of the proposed approach are described through the LQG/LTR multi-variable robust control method development's example for a gas-turbine engine and its software-hardware framework of HIL platform.

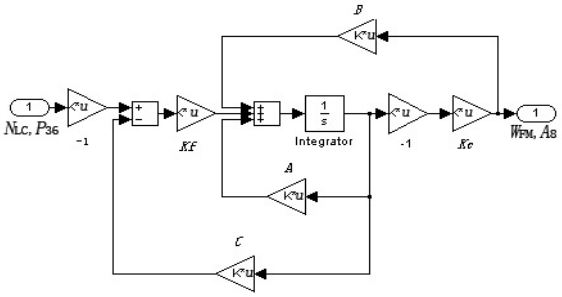

Fig. 1 LQG/LTR controller model in Simulink

4.1 Platform Framework

Figure 2 shows the software-hardware framework of HIL platform for a gas-turbine engine. The whole platform is composed of an industrial control computer for simulated engine, a personal computer for workstation, a PC104 for ECU, and other devices for communication and data collection. According to the request of real-time simulation and prototype testing, RTOS is Vxworks in ECU and xPC in simulated engine. MATLAB runs in the workstation, completing main tasks such as CDS modeling, CCS designing, digital off-line simulation, code generating, real-time control, parameters tuning, data logging, etc. At the same time, Tonardo IDE also runs in this workstation, completing some tasks such as the kernel of Vxworks generating, communication between server and client connecting, the embedded program downloading, etc. A gas-turbine engine model program which has been validated and packaged by S-Function runs in the simulated engine, sending signals like engine states, environment conditions via D/A. Furthermore, it communicates with the workstation via TCP/IP. Similarly, CCS program runs in the ECU, also sending control signals via D/A and communicating with the workstation via TCP/IP. In the whole framework, the real-time simulation loop is mainly composed of A/D and D/A whereas the monitoring loop is TCP/IP.

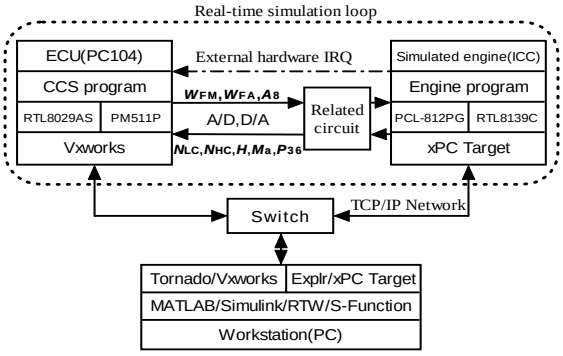

Fig. 2 Software-hardware framework

4.2 Constructing & Key works

After obtaining the non-line aero-engine model by packaging in Matlab/Simulink, it is simplified to convert into line model and design the related control method. Different from normal way in the digital off-line simulation is that the controlled objective includes line and non-line aero-engine model, which is benefit to recorrecting the control method in deeply. Next step is to divide the whole system into two unabridged systems through throwing away those models replaced by actual devices, and adding related device drivers and self-defined hardware IRQ block. In other words, the virtual signals between control system and aero-engine are changed to real signals, as shown in Figure 3.

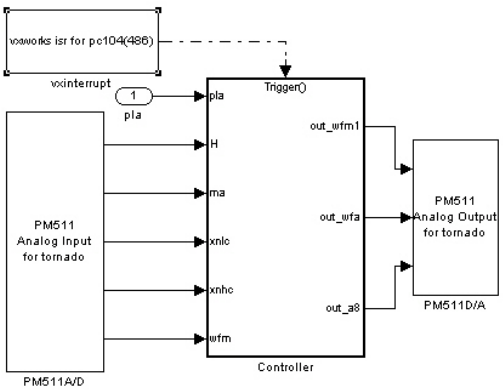

Fig. 3 Control system model in Simulink

It involves the following key works:

(1) Creating model of aero-engine in Simulink or obtaining it created in other languages via S-Function interface technology;
(2) Writing or transplanting device drivers for Vxworks[9];
(3) Writing self-defined hardware IRQ block for Vxworks;
(4) Setting template makefiler--tornado.tmf for tornado target;
(5) Choosing integrator solver and compiler;

Among these, step (2) and step (3) need to be packaged into Simulink models by S-Function when completed.

After that, the model can be automatically translated into embedded code running in Vxworks or xPC RTOS via RTW, which is rapid transition from digital off-line simulation to HIL real-time simulation and prototype testing.

5. Simulation & Verification

After building up the HIL platform shown in Figure 2 and DEEC system shown in Figure 3, the following simulation processes is:

(1) Running Tornado IDE in the workstation, clipping a kernel of Vxworks, creating a boot for ECU, finally starting it and downloading the kernel via TCP/IP.
(2) At the same way, running xPC management GUI, creating the kernel of xPC for virtual engine, and starting.
(3) Setting the parameters in external mode, included the way of communicating with target, specific signals, the way of trigger in the beginning of simulation. In this way, it is convenient to access to signal control, real-time monitor, on-line parameters tuning, data logging, etc.

In this paper, we make some experiments about real-time simulation and compare between automatic code and manual code for an aero-engine multi-variable control method--LQG/LTR, including simulation precision and execute effeciency.

On the ground condition (H=0km, M_a=0), the original value of main parameters is: W_{FM}=4500kg/h, W_{FA}=0kg/h, A_8=0.2602m^2.

The simulation processes includes acceleration state, deceleration state, afterburner light-on, afterburner light-off, middle-steady state, as shown in Table 1.

Table 1 Simulation processes

Time /s	P_{LA}	Afterburner light-on	Aero-engine state
0	70°	No	Max. state
3	130°	Yes	Max. with afterburner light-on
6	80°	Yes	Little afterburner light-on
9	70°	No	Max. state
12	60°	No	middle-steady state

Figure 5 and Figure 6 provide coherent simulation results about precision since one cannot distinguish between curves of the real-time simulations in two ways, except that there is about 10^{-4} level relative error at 3s. According to analyse their code for this minim error, we know that the reason is the different from initialization. However, this situition can't effect on actual requirement of aero-engine control system.

The results of execute efficiency in two ways is shown in Figure 7 and Figure 8. It shows that the average execute time in the way of automatic code is about 2.0020×10^{-6}s and the ones in manual code is about 1.4619×10^{-6}s. Apperantly, the front is 36.95% longer than the latter because the law of automatic code is incline to universal and compatible. So compared with the manual code, the automatic code is met with the actual requirement in despite of lossing a little efficiency.

In short, the simulation results verify that the approach proposed in this paper is feasible and effective.

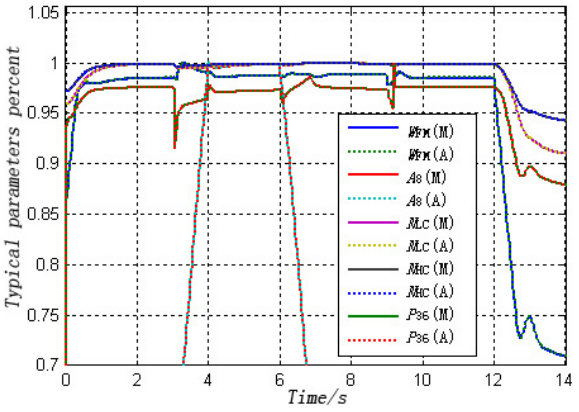

Fig.4 Parameters variety in two ways

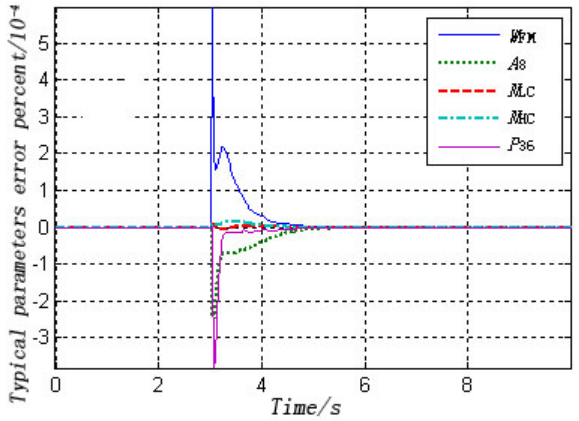

Fig.5 Compare with parameters variety in two ways

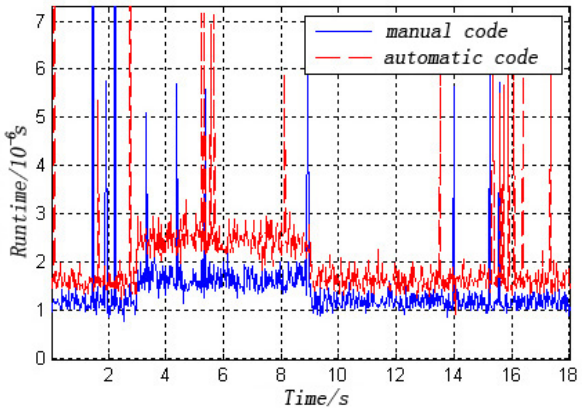

Fig.6 Runtime in two ways

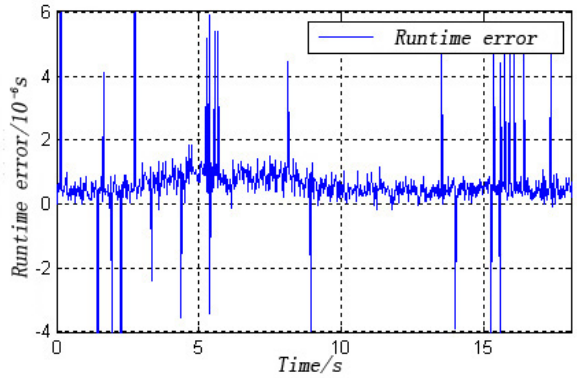

Fig.7 Compare with runtime in two ways

5. Conclusion

This paper presents an approach of aero-engine CCM real-time simualtion and rapid prototyping and builds up the matched HIL simualtion platform. Finally, according to the simulation and verification between automatic code and manual code for an aero-engine multi-variable control method—LQG/LTR, the results is as follow: (1) there is almost no error in two ways in simulaiton precision and met with the actual application; (2) the front efficiency also met with the actual application although it is less than latter ones.

Furthermore, according to our experiments, it is proved that the front is obviously shorter than the latter in the development cycle.

In short, this approach is feasible and effective, and not only decreases development cycle but also improves the reliability and universality. Since that, a series of problems can be resolved during the simulation stage and rapid application to prototype testing.

Acknowledgements

The research described in this paper was supported by graduate starting seed fund of Northwestern Polytechnical University.

References

[1] Tom Erkkinen： Model style guidelines for flight code generation, *AIAA*, 2005-6216.

[2] CHENG Zhiyu, WEN Yanjun, CHEN Qi.： *Vxworks Development and Practice*, People's Post & Telecommunication Press, Beijing, 2004.

[3] YANG Di, LI Litao, YANG Xu, etal: *MATLAB/RTW and It's Application*, Tsinghua University Press, Beijing, 2002.

[4] C. A. Rabbath, E. Bensoudane: Real-time modeling and simulation of a gas-turbine engine control system, *AIAA*, 2001-4246.

[5] C.A. Rabbath, H. Desira, K. Butts: Effective modeling and simulation of internal combustion engine control systems, *Proceedings of American Control Conference*, 2001, pp. 1321-1326.

[6] Akihiro Kimura, Iwao Maeda: Development of engine control system using real time simulator, *IEEE*, 1996, pp. 157-163.

[7] Marko Bacic, Ron Daniel: Towards a low-cost hardware-in-the-loop simulator for free flight simulation of unmanned air vehicles, *AIAA*, 2005-6102.

[8] Paul A. Barnard: Software development principles applied to graphical model development[R]. *AIAA*, 2005-5888.

[9] Dobrokhodov V.: Developing serial communication interfaces for rapid prototyping of navigation and control tasks[R]. *AIAA*, 2005-6099.