Proceedings of The Canadian Society for Mechanical Engineering International Congress 2012
CSME International Congress 2012
June 4-6, 2012, Winnipeg, Manitoba, Canada

A PROGRAMMABLE LOGIC CONTROLLER EMULATOR WITH AUTOMATIC ALGORITHM-SWITCHING FOR COMPUTATIONAL FLUID DYNAMICS USING OPENFOAM

David L. F. Gaden and Eric L. Bibeau
Department of Mechanical Engineering, University of Manitoba
Winnipeg, Canada
david_gaden@umanitoba.ca, eric_bibeau@umanitoba.ca

Abstract—A programmable logic controller (PLC) emulator is implemented into OpenFOAM, a computational fluid dynamics (CFD) software suite. The controller is generic and allows users to study the fluid dynamics and the control system of engineering processes simultaneously. The user specifies their own inputs, outputs and logic, and the software will subsequently simulate the entire time dependent process. The controller also allows the simulation to dynamically change the algorithm it uses, making it suitable for a wider range of processes, including those with distinct stages characterised by different physics. This is the most versatile PLC emulator ever implemented and successfully demonstrated for CFD.

Keywords-PLC;control;CFD;algorithm-switching;OpenFOAM

I. INTRODUCTION

Computational fluid dynamics (CFD) is used to simulate engineering processes involving fluid flow. Many such processes involve control systems, and these complicate the use of CFD. For instance, the heating, ventilating, and air-conditioning (HVAC) system of a building has various components, such as fans and heaters, that may be on or off depending on the state of the system. This can be accommodated by performing CFD simulations for all possible states, or by incorporating the control system into the simulation itself.

Control systems sometimes use a previously created library of simulation data to inform their operation [1-6]. This is achieved by performing CFD simulations for all possible states, and for a variety of expected conditions. However, this strategy can become impractical with system complexity, as the number of states increases exponentially with the number of switchable outputs. Furthermore, each simulation begins with a predetermined initial state, which will not accurately capture transitions between states, nor will the set of simulations be adequate in evaluating the control strategy. A better solution is to incorporate the control system directly into the CFD simulation itself, handling system changes internally based on logic provided by the user.

There is a large body of literature involving CFD simulations with built-in control systems [3,7-9], and these provide insight into their individual control strategy. However, these systems are custom-written, application-specific implementations. A CFD simulation with a general framework for process control would be more versatile. A programmable logic controller (PLC) emulator for would achieve this goal.

Once a general process control framework is implemented, a CFD solver becomes applicable to a wider set of conditions, due to the changes in the system state. As a direct consequence of this, the system may enter a state where the original solver algorithm is inadequate. This includes engineering processes with distinct stages characterized by differing physics, or simulations that require an initialization algorithm. For instance, a chemical batch process may begin with a transient heating and mixing stage, followed by a steady-state reacting stage. A solver for such a process must be capable of *algorithm-switching*.

Algorithm-switching is the ability of a CFD solver to dynamically change the algorithm during a simulation. Although not formally defined, some algorithm-switching has previously been reported. Gains in efficiency have been realized by dynamically changing the discretization schemes mid-run [10,11]. Kobolov et al. [12,13] developed a solver that switches from a rarefied medium algorithm to a continuous medium algorithm for spacecraft re-entry studies. However, a general algorithm-switching framework for CFD has not been reported.

The purpose of this paper is to develop a proof-of-concept PLC emulator and an algorithm-switching framework for CFD simulations, verify the emulator, and release the resulting source code to the CFD community as free and open source software.

II. THEORY

A PLC has a set of inputs, outputs, and logic.

- **Inputs** are any measurable quantity within the simulation, including time. These represent actual sensors, such as flow meters, thermocouples, or timers in the physical system being simulated.
- **Outputs** are changeable entities within the model that can alter the course of the simulation. This can include boundary conditions, modelling constants, source terms, and even the algorithm to use, dynamically changing the sets of non-linear equations to solve. The outputs represent physical components such as valves, heaters, and fans.
- **Logic** is a user-supplied set of conditions that describes the state of the outputs for all given sets of inputs; or, when provided with an initial state, describes only the conditions that cause a change of state. Ladder logic is the most common form of PLC logic descriptor.

III. CODE DEVELOPMENT

OpenFOAM, a free and open source CFD suite, is used as a platform for this project.

A. *Overview*

The sequence of operations for the proposed emulator is shown in Figure 1.

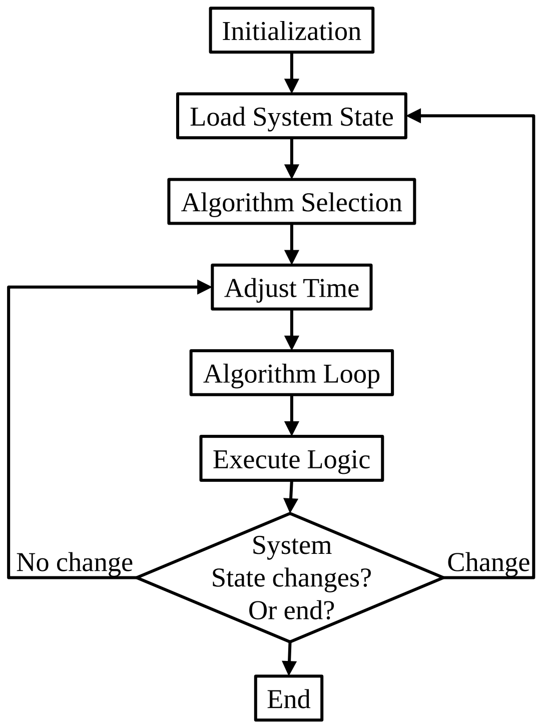

Figure 1. PLC emulator process diagram

Since the PLC emulator uses timers as inputs, it sometimes must modify the end time and timestep of the current simulation in order to properly meet the end of the timer. That is the purpose of the “adjust time” step in the emulator.

B. *Algorithm-switching*

OpenFOAM does not have algorithm-switching capabilities, therefore an extension was written, called *multiSolver*. multiSolver was publicly released as free and open source software in July of 2010 [14], and has subsequently been used by the OpenFOAM community [15].

Since each algorithm requires its own set of objects in memory, the only safe and practical way to switch between algorithms is to let the first algorithm terminate normally and subsequently initialize the next algorithm using the latest values for each variable. Therefore, there is considerable overhead in switching between algorithms. However, this gives an opportunity to change the case settings, including boundary conditions, modelling constants, and source terms. Thus multiSolver can be used in a general way to control the system state.

C. *Outputs*

To control the simulation, the PLC emulator must be capable of changing:

- the boundary conditions;
- the source terms;
- the modelling constants; and
- the algorithm.

multiSolver has the ability to address these requirements, and therefore can be used universally; however, each time it makes a change to the system state, multiSolver deletes and reinitializes all objects in memory. This overhead needs to be considered if the control scheme requires frequent changes to the outputs and for larger simulations; however, this cost of using a PLC emulator must also be balanced with its benefits, which include:

- increased simulation quality assurance by minimizing user errors when manually manipulating input and output variables between control states,
- reduced time for the CFD user when running a complete simulation case, and
- making simulations more representative of the physical world.

D. *Inputs*

The inputs available to the PLC are:

- **variable limits** – when a given variable is greater than or less than a given limit, evaluated either at a specific position, or averaged over the entire geometry;
- **equation** – when a user-supplied equation is greater than or less than a given limit;
- **timers** – these can repeat, and can be triggered by other inputs;
- **system state groups** – these inputs are true or false depending on the state of the system;

- **solver signals** – signals sent from the solver to the PLC; and
- **conditional switches** – these are true or false depending on the conditions of other inputs; and

The PLC emulator uses OpenFOAM's *runtime selection* idiom, wherein it uses a set of generic base classes as interfaces between model components. This gives it a modular, extensible design. Therefore, if a user wishes to create additional custom inputs, this can be achieved without modifying any of the existing code, thus providing a complete environment in which to easily define the control logic.

E. Logic

The logic is user-specified in a text-based input file, the format for which is documented with the software release. The logic input file is currently verbose: a case with four controllable switches requires close to 400 lines of user-input as a text parser has not yet been implemented.

IV. CASE SETUP

A proof of concept case study and verification is conducted. The case study simulates a semi-continuous chemical process in the reactor vessel, as shown in Figure 2.

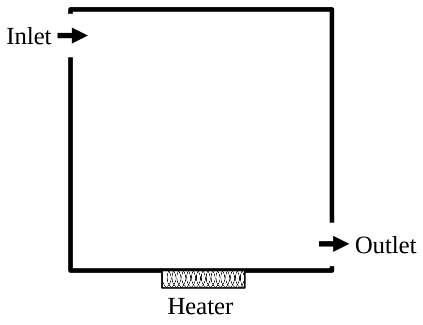

Figure 2. Reactor vessel

The process is:

- a relatively cold fluid (323 K) flows into a reactor vessel for sixty seconds, at which point the vessel is sealed and heated;
- once the heat reaches a set temperature (473 K), the reactions begin;
- the temperature is controlled by a thermostat that switches on below 513 K, and off above 523 K;
- after 1,000 seconds (close to sixteen minutes), the inlet and outlet are opened again; and
- the products flow out, cold fluid flows in, and the process repeats.

The case study does not have actual reactions; rather it uses the concept of reactions to demonstrate algorithm-switching. There are two algorithms: one with reactions, and one without. They are both identical, except the one with reactions also defines a secondary variable, *S*, which follows a sinusoidal trajectory.

There are three outputs managed by the PLC in this model. These are:

- fluid flow, on or off;
- heater, on or off; and
- reactions, on or off.

The inputs are:

- temperature; and
- time.

For simplicity, the case study is a two-dimensional, incompressible simulation. Boussinesq buoyancy is used, thus providing a mixing effect when the flow is switched off. In order to demonstrate thermostat function, the heat loss through the vessel walls is exaggerated.

V. RESULTS AND DISCUSSION

The temperature and reaction variable profiles are a good proxy for visualising the operation of the controller. A plot of these variables is shown in Figure 3. The simulation begins with the flow and heater on. The sharp change in the temperature rise at 60 seconds (feature A) occurs when the process flow stops. At approximately 390 seconds, as the temperature passes 200 °C, the first reactions begin and the reaction variable begins registering oscillations (feature B). This demonstrates algorithm-switching is occurring. The saw-toothed profile of the temperature during these reactions (feature C) is the thermostat controlling the heater. The sharp drops at 1,000 and 2,000 seconds (feature D) occur when the flow is switched on, allowing cold fluid to enter the reactor vessel.

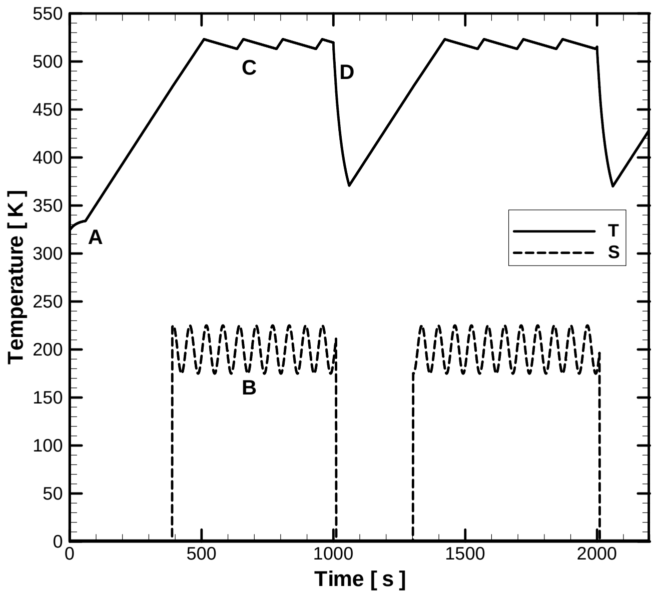

Figure 3. The average temperature and reaction variable, as a function of process time.

The flow profiles also show proper controller function. When the flow is switched off, buoyancy forces dominate, giving the profile shown in Figure 4.

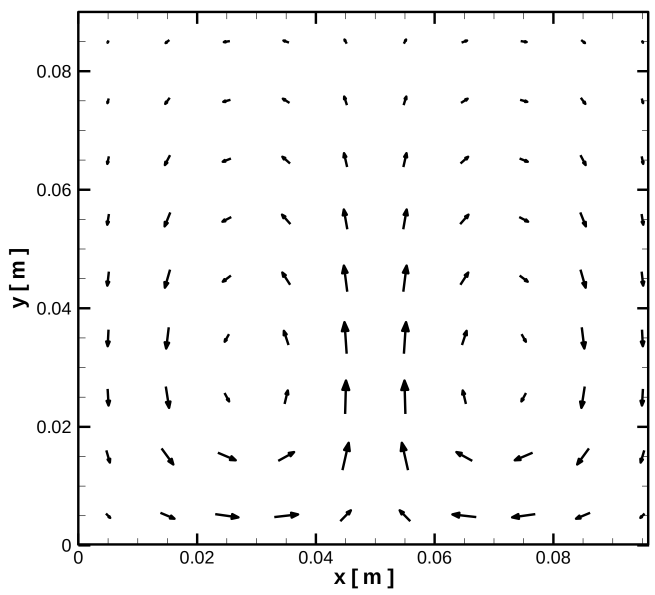

Figure 4. Velocity vectors with flow switched off.

Similarly, when the flow is switched on, a cross-flow is seen traveling from inlet to outlet, as shown in Figure 5.

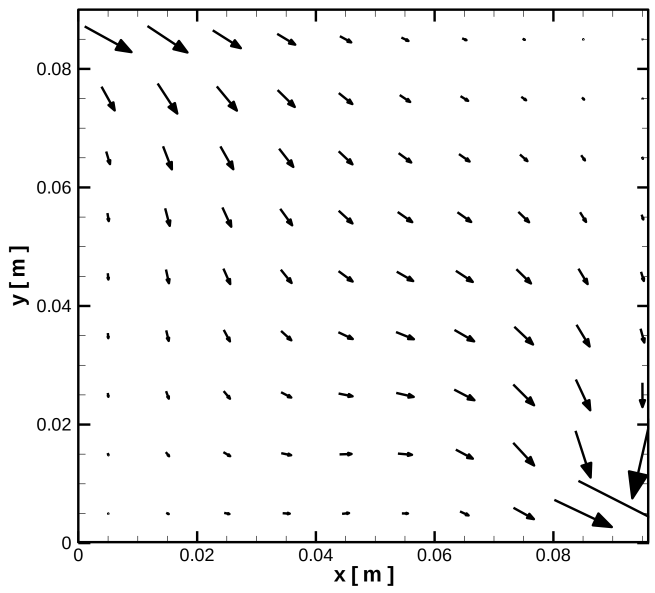

Figure 5. Velocity vectors with flow switched on.

The case study shows proper function of the PLC emulator, and the expected response within the CFD solution. Algorithm-switching for a simple case study has been verified.

VI. Conclusion

A general PLC emulator with algorithm-switching was implemented within the framework of a CFD simulation. A proof-of-concept case study verified its proper function and demonstrated its usefulness. With this PLC emulator, a user can evaluate complex 3-D processes using CFD, from the perspectives of fluid flow and process control simultaneously. Algorithm-switching makes this tool more versatile as it can apply to a wider range of physics and boundary conditions.

VII. Future Work

The PLC emulator does not currently support parallel processing. Once parallel processing has been implemented, the PLC emulator will be made publicly available as free and open source software.

Other suggested improvements include:

- a logic parser would make the entering the PLC logic easier from a user's perspective, as the current format requires approximately 100 lines of user-input per output switch;
- mesh motion needs to be incorporated into the multiSolver component in order to allow control systems that manipulate geometry; and
- a mechanism through which the PLC emulator can directly change boundary conditions would improve model performance.

Contributions from the open source community are welcomed.

Acknowledgment

Funding for this project was provided by Manitoba Hydro, and the Natural Sciences and Engineering Research Council of Canada (NSERC). Technical assistance from Hrvoje Jasak was gratefully appreciated.

References

[1] R. Zhang, C. Zhang, J. Jiang, "A new approach to design a control system for a FGR furnace using the combination of the CFD and linear system identification techniques," Combustion Theory and Modelling, Vol. 15, No. 2, pp. 183-204, 2011.

[2] Y. Liu, Hanchang Shi, Huiming Shi, Z. Wang, "Study on a discrete-time dynamic control model to enhance nitroten removal with fluctuation of influent in oxidation ditches," Water Research, Vol. 44, No. 18, pp. 5150-5157, 2010.

[3] M. Milovanovic, O. M. Aamo, "Attenuation of vortex shedding in CFD simulations by model-based output feedback control," 49th IEEE Conference on Decision and Control, Atlanta, USA, December 15-17, 2010.

[4] T. Plikas, L. Gunnewiek, T. Gerritsen, M. Brothers, A. Karges, "The predictive control of furnace tapblock operation using CFD and PCA modeling," JOM, October 2005.

[5] Y. Yang, M. A. Reuter, D. T. M. Hartman, "CFD modelling for control of hazardous waste incinerator," Control Engineering Practice, Vol. 11, No. 1, pp. 93-101, 2003.

[6] Stropky D., Bibeau E.L., Yuan J., Salcudean M., "Advanced process modelling saves money," 2001 Joint Engineering/F&C Conference, San Antonio, Texas, December, 2001.

[7] Z. Sun, S. Wang, "A CFD-based test method for control of indoor environment and space ventilation," Building and Environment, Vol. 45, No. 6, pp. 1441-1447, 2010.

[8] S. Wong, W. Zhou, J. Hua, "Designing process controller for a continuous bread baking process based on CFD modelling," Journal of Food Engineering, Vol. 81, No. 3, pp. 523-534, 2007.

[9] S. B. Choi, H. Xu, M. D. Mirmirani, "LQG control of a CFD-based aeroelastic wing model," Proceedings of the 42nd IEEE Conference on Decision and Control, Maui, USA, December 2003.

[10] R. Winkelmann, J. Häuser, R. D. Williams, "Strategies for parallel and numerical scalability of CFD codes," Computer Methods in Applied Mechanics and Engineering, Vol. 174, No. 3-4, pp. 433-456, 1999.

[11] R. Löhner, H. Luo, J. D. Baum, D. Rice, "Improvements in speed for explicit, transient compressible flow solvers," International Journal for Numerical Methods in Fluids, Vol. 56, No. 12, pp. 2229-2244, 2008.

[12] V. I. Kolobov, H. Q. Yang, S. A. Bayyuk, V. V. Aristov, A. Frolova, S. Zabelok, "Unified methods for continuum and rarefied flows," 42nd AIAA Aerospace Sciences Meeting and Exhibit, Reno, USA, January 5-8, 2004.

[13] V. Kobolov, V. Aristov, R. Arslanbekov, S. Bayyuk, A. Frolova, S. Zabelok, "Construction of a unified continuum/kinetic solver for aerodynamic problems," Journal of Spacecraft and Rockets, Vol. 42, No. 4, pp. 598-606, 2005.

[14] D. L. F. Gaden, “multiSolver released,” CFD-Online Forums, message posted to www.cfd-online.com/Forums/openfoam-news-announcements-other/78502-multisolver-released.html, July 23, 2010.

[15] K. Gott, A. Kulkarni, “Construction and application of a unified flow solver for use with physical vapour deposition (PVD) modeling,” poster presented at the 6th OpenFOAM Workshop, State College, USA, June 13-16, 2011.