Proceeding of the 2004 American Control Conference
Boston, Massachusetts June 30 - July 2, 2004

Gain Hands-On Experience in Process Control Using Control Station

Doug Cooper[1], Robert Rice and Jeff Arbogast
Chemical Engineering Department
University of Connecticut
Storrs, CT 06269-3222

Phone (860) 486-4092
Fax (860) 486-2959
cooper@engr.uconn.edu

Abstract

Training simulators enhance learning by integrating the theoretical abstraction of textbooks with the tactile nature of the lab. The primary objective of training simulators is education. They motivate, help with visualization, and provide hands-on practice and experience. This paper explores the use of training simulators for process control education. Discussed are the methods used and experiences gained in developing Control Station, a training simulator used in the curriculum of hundreds of academic institutions worldwide.

1. Introduction

Hands-on challenges which demonstrate and reinforce abstract process control concepts benefit the learning process. Such challenges can be motivating, promote critical thinking, facilitate understanding in the use and limitations of the theory, and help prepare students for the challenges of the professional world.

A process control training simulator offers an alluring method for providing students with the significant hands-on practice critical to learning the subject. The proper tool can provide virtual experience much the way airplane and power plant simulators do in those fields. It can give students a broad range of focused engineering applications of theory in an efficient, safe and economical fashion. And it can work as an instructional companion as it provides interactive challenges that track along with classroom lectures.

Process control is a subject area well suited to exploit the benefits of a training simulator [1]. Modern control installations are computer based, so a video display is the natural window through which the subject is practiced. With color graphic animation and interactive challenges, a training simulator can offer experiences that literally rival those of the real world [2]. These experiences can be obtained risk free and at minimal cost, enabling students to feel comfortable exploring nonstandard solutions at their desk. If properly designed as a pedagogical tool with case studies organized to present incremental challenges, we believe learning can be enormously enhanced for process control with such a training simulator.

2. A Chemical Perspective

Each discipline views process control from a different perspective. To help orient the reader, consider these "typical" examples drawn from chemical process control:

Process Variables: temperature, pressure, pressure drop, level, flow, density, concentration
Final Control Elements: solenoid, variable position valve, variable speed pump or compressor, heater or cooler
Control Algorithms: on/off, PID, cascade, ratio, feed forward, multivariable decouplers, model predictive (Smith predictor up through dynamic matrix control)
Process Applications: reactors, separators, distillation columns, heat exchangers, furnaces

The chemical engineering perspective considers that each process is one of a kind. Consequently, every control system can appear unique in design, implementation and man-machine interface.

Additionally, chemical processes are nonlinear and nonstationary, have long time constants, significant dead time, and noisy measurement signals. Disturbances occur from numerous sources including loop interaction from other controllers in the plant.

3. About Control Station

To this end, Control Station was developed as an instructional tool for process dynamics and control. Control Station is a point-and-click environment compatible with Microsoft Windows® and most computer networks. The development goal was a simulator that is visually appealing, easy-to use and accepted by students and practitioners. The software is designed so students will:

- learn how to collect and analyze process data to determine the essential dynamic behavior of a process,
- learn what "good" or "best" control performance means for a particular process,
- understand the computational methods behind the different control algorithms and learn when and how to use each one to achieve best performance,
- learn how the different adjustable or tuning parameters required for control algorithm implementation impact

[1] Author to whom correspondence should be addressed

0-7803-8335-4/04/$17.00 ©2004 AACC

control performance and how to determine values for these parameters,

- become aware of the limitations and pitfalls of each control algorithm and learn how to turn this knowledge to their advantage.

Control Station is comprised of three modules: *Case Studies*, *Custom Process* and *Design Tools*. The *Case Studies* module provides real-world experience in modern methods and practices of process control through a collection of realistic processes to practice upon. The *Case Studies* processes available for study in Control Station include: level control of a tank, temperature control of a heat exchanger, concentration control of a reactor, and purity control of a distillation column. The controllers available for exploration and study on these processes include: P-Only, PI, PD and PID controllers; cascade; feed forward; multivariable decoupling; model predictive (Smith predictor); and discrete sampled data control.

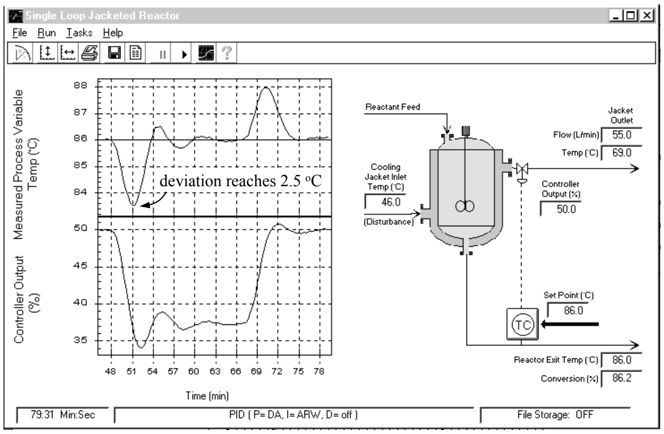
The *Custom Process* module is a block oriented environment that lets students construct a process and controller architecture to their own specifications for a wide range of custom control analyses. Students can investigate the benefits and drawbacks of different control architectures, tunings sensitivities, loop performance capabilities, and a host of other issues important to the practitioner.

The *Design Tools* module is used to fit linear dynamic models to process data and to compute PID controller tuning values. The models from *Design Tools* can also be used to construct advanced control strategies that use process models internal to the controller architecture such as feed forward and model predictive control. Because data can be imported from real operating processes, *Design Tools* can solve a wide range of challenging problems for controller design, analysis and tuning.

4. Example: Rejecting Process Disturbances

To illustrate how Control Station is used for learning, consider the implementation of a control strategy designed to reject a disturbance that upsets steady operation. The most popular architectures for disturbance rejection are cascade control and feed forward with feedback trim. Both architectures trade off added instrumentation and engineering time in return for a controller better able to reject disturbances to the measured process variable. Neither architecture benefits set point tracking performance.

We consider the jacketed reactor process for this study. We begin by establishing a base case of disturbance rejection using a conventional single feedback control loop. We then detail the implementation of cascade and feed forward architectures and compare disturbance rejection performance and capability.



Flow (L/min) 55.0
Temp (°C) 69.0
Cooling Jacket Inlet Temp (°C) 46.0
Controller Output (%) 50.0
Set Point (°C) 86.0
Reactor Exit Temp (°C) 86.0
Conversion (%) 86.2
79:31 Min:Sec
PID (P= DA, I= ARW, D= off)
File Storage: OFF

Figure 1 – Disturbance rejection in jacketed reactor with single loop feedback control architecture

4.1 Base Case - Conventional Feedback Control

Figure 1 shows the screen display for the jacketed reactor process under single loop feedback control. The jacketed reactor simulation is of a continuously stirred vessel in which an exothermic reaction occurs. Residence time is constant in this well mixed reactor, so the conversion of reactant feed to desired product can be inferred from the temperature of the reactor exit stream.

To control the reactor's exit stream temperature (the measured process variable), the vessel is enclosed with a cooling jacket. The controller manipulates a valve to adjust the cooling jacket liquid flow rate. If the exit stream temperature (and thus conversion) is too high, the controller opens the valve, thus increasing the cooling liquid flow rate. This cools the reactor and slows the reaction, which causes the measured temperature of the stream exiting the reactor to drop. As shown in Fig. 1, the disturbance variable for this process is the temperature of cooling liquid entering the jacket.

The control objective is to maintain the reactor exit stream temperature at set point by rejecting disturbances from changes in the cooling jacket inlet temperature. Though the user is free to change operating conditions, here the design level of operation is a measured reactor exit stream temperature of 86 ^{o}C. The cooling jacket inlet temperature (disturbance) is normally at 46 ^{o}C, but can unexpectedly spike as low as 40 ^{o}C. A controller output of 50% causes the reactor to steady at the design exit stream temperature of 86 ^{o}C when the cooling jacket inlet temperature is at its design value of 46 ^{o}C.

Tuning the single loop PI controller for this study follows a standard procedure applicable to real processes in the plant or lab as well as to the Control Station simulations:

a) from design level of operation, step of pulse the controller output signal so that the measured process variable response clearly dominates the noise in the measurement signal. Record the process response data.

b) fit a first order plus dead time (FOPDT) model to this controller output to measured process variable data. *Design Tools* is well-suited for this modeling task.
c) use the resulting FOPDT model parameters in a correlation to compute initial controller tuning values. *Design Tools* also performs this task if desired.
d) implement your controller and fine tune online until your control objectives are satisfied.

As shown in Fig. 2, the controller output is stepped from the design value of 50% up to 53%, down to 47% and back to 50%. The measured reactor exit stream temperature exhibits a clear response after each controller output step that dominates any measurement noise.

The resulting dynamic process data is read into Control Station's *Design Tools* module, where a few clicks of the mouse results in a fit of a FOPDT model to the process data. The model fit is shown in Fig. 2 with the models parameters listed at the bottom of the figure.

Design Tools uses the model parameters in tuning correlations and recommends the internal model control (IMC) (3,4) tuning values for a PI controller:

Controller Gain, $K_C = -2.7$ %/°C
Reset Time, $\tau_I = 1.6$ min

We return to the jacketed reactor process and enter these values into the controller design menu. The disturbance rejection performance of the single loop PI controller is tested as shown in Fig. 1.

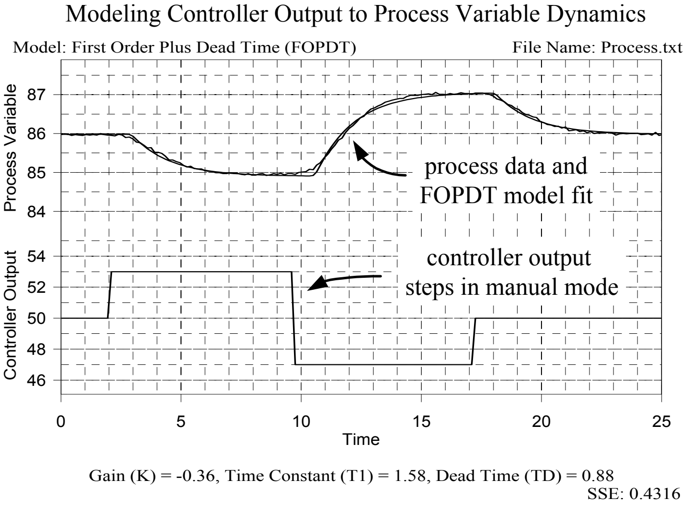


Figure 2 − Design Tools *fit of a FOPDT model of controller output to measured process variable data*

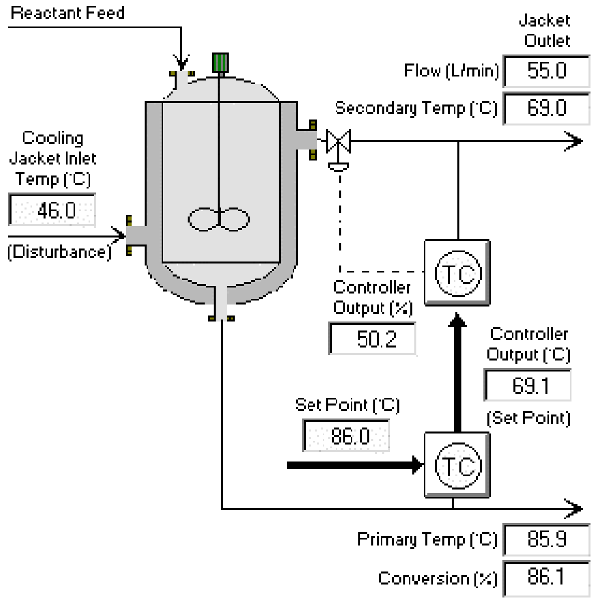
As shown in Fig. 1, the measured reactor exit stream temperature is initially steady at the design set point value of 86 °C. To test the controller, the cooling jacket inlet temperature is stepped from its design value of 46 °C down to 40 °C and back again (this disturbance trace is not shown in Fig. 1 but could be seen using software plotting options). The single loop PI controller is able to maintain reactor exit stream temperature near the constant set point of 86 °C, with deviations ranging as high as 2.5 °C during the event.

4.2 Cascade Control Disturbance Rejection

The open loop behavior of the cascade jacketed reactor is identical to that of the single loop base case. The Control Station process graphic for the cascade controller architecture is shown in Fig. 3. A cascade design requires a *secondary* process variable. The main process variable associated with original control objective is called the *primary* variable. This secondary process variable has specific requirements:

- it must be measurable with a sensor,
- the same final control element (e.g. valve) used to manipulate the primary variable must also manipulate the secondary variable,
- the same disturbances of concern for the primary variable must also disrupt the secondary variable, and
- the secondary variable must be *inside* the primary process variable, which means it responds well before the primary variable to disturbances and final control element manipulations.

With a secondary process variable identified, a cascade structure is constructed with the secondary loop literally nested inside the primary loop.



Flow (L/min) 55.0
Secondary Temp (°C) 69.0
Cooling Jacket Inlet Temp (°C) 46.0
(Disturbance)
Controller Output (%) 50.2
Controller Output (°C) 69.1
(Set Point)
Set Point (°C) 86.0
Primary Temp (°C) 85.9
Conversion (%) 86.1

Figure 3 − Jacketed reactor process with cascade control architecture

A cascade architecture requires two sensors and two controllers but only one final control element because the output of the primary controller, rather than going to a valve, becomes the set point of the secondary controller as shown in Fig. 3. Cascade loop tuning is a familiar task as the architecture is comprised of two controllers from the

PID family. The approach is to tune the inner secondary loop first and then leave it in automatic. It literally becomes part of the primary process. Then tune the outer primary controller. With both controllers in automatic, tuning of the cascade is complete.

Because the control objective for the jacketed reactor is disturbance rejection, it is appropriate to consider a cascade architecture. The primary process variable remains the reactor exit stream temperature. To construct a cascade, we need to identify a secondary process variable and as shown in Fig. 3, we use the cooling jacket outlet temperature in this study. The manipulated variable of the secondary loop is the cooling jacket liquid flow rate.

To implement a cascade, the secondary controller is tuned while the primary controller is in manual mode. Like the single loop PI controller study, the design level of operation is a reactor exit stream temperature of 86 ^{o}C , which is achieved with a controller output at 50% when the cooling jacket inlet temperature is 46 ^{o}C.

The details of the P-Only secondary controller design are omitted but the set point tracking performance of the secondary loop is shown in Fig. 4. The primary loop is still in manual mode at this point. As expected for a P-Only controller, offset exists when the set point is not at the design value. The secondary process variable responds quickly and settles rapidly to set point changes so we consider the design to be complete. The secondary loop is left in automatic and we now tune the primary controller.

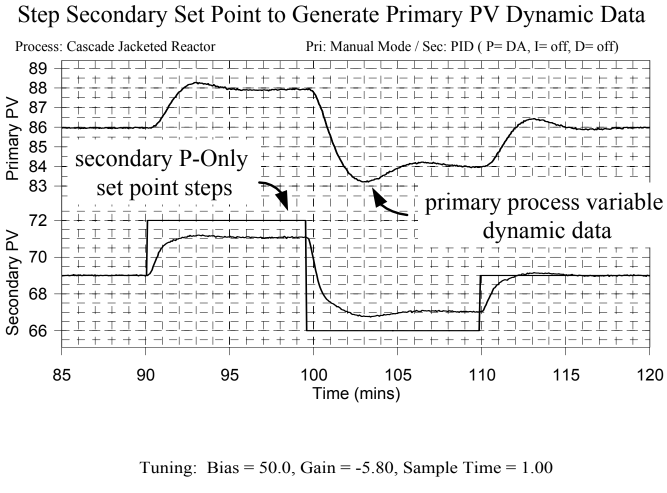


Figure 4 − Set point tracking performance of the secondary loop under P-Only control

In the cascade architecture, the controller output of the primary loop is the set point of the secondary controller. So to design the primary controller, we must perturb the secondary set point and then fit a model to the corresponding response in the primary measured process variable. Figure 4 contains such data and this is used for the design of the primary PI controller.

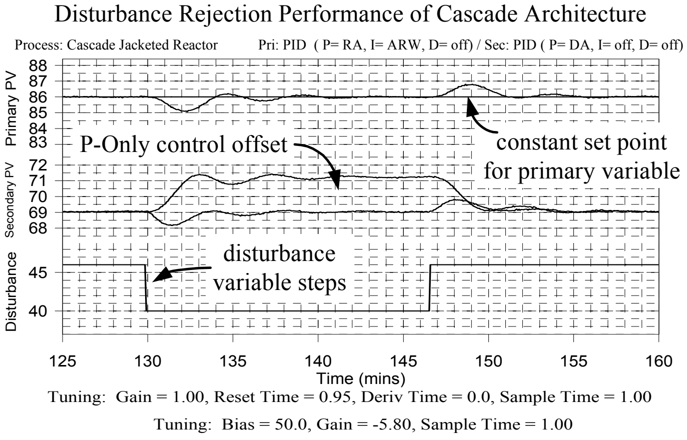


Figure 5 − Disturbance rejection in the jacketed reactor with a P-Only to PI cascade

The set point of the secondary P-Only controller as shown in Fig. 4 is stepped from its design value of 69 ^{o}C, up to 72 ^{o}C, down to 66 ^{o}C, then back to 69 ^{o}C. The measured reactor exit stream temperature displays a clear response after each change that dominates the measurement noise. A FOPDT fit of the dynamic data by *Design Tools* yields model parameters that when used in the IMC correlation, yield the PI controller tuning values:

$$\text{Controller Gain, } K_C = 1.0 \; \frac{^{o}\text{C cooling jacket out}}{^{o}\text{C reactor exit}}$$

$$\text{Reset Time, } \tau_I = 0.95 \text{ min}$$

Figure 5 shows the disturbance rejection performance of the cascade reactor using these tuning parameters for the primary PI controller while the secondary loop remains under P-Only control as previously described. Just as in Fig. 1, the primary measured process variable (reactor exit stream temperature) is initialized at the design set point value of 86 ^{o}C. To test the controller, the cooling jacket inlet temperature is again stepped from its design value of 46 ^{o}C down to 40 ^{o}C and back again.

As shown in Fig. 5, the cascade architecture performs better than the single loop case in maintaining the reactor product temperature near the constant set point of 86 ^{o}C. Specifically, while the measured reactor exit temperature deviations for the single loop PI controller range as high 2.5 ^{o}C during the event, here the cascade limits the maximum deviation to about 1.0 ^{o}C. This improvement did not come free, however, as the cascade architecture requires an additional sensor, controller and tuning effort.

4.3 Feed Forward Disturbance Rejection

A feed forward controller is an alternative architecture for improved disturbance rejection. Feed forward implementation requires the purchase and installation of a sensor to directly measure the offending disturbance, and the development and programming of a feed forward control element. This element is comprised of a disturbance model and a process model. Both models are linear in form. The computation performed by the feed forward element may be thought of as a two step procedure:

- the *disturbance* model receives the disturbance measurement and predicts an "impact profile," or when and by how much the measured process variable will be impacted.
- given this predicted sequence of disruption, the *process* model then back calculates a series of feed forward control actions that will exactly counteract the disturbance as it arrives so the measured process variable remains constant at its set point.

Implementation requires that the two models be programmed into the control computer. Linear models will never exactly describe the behavior of real nonlinear process, so a feed forward element can dramatically reduce the impact of a disturbance but will never provide perfect disturbance rejection.

To account for model inaccuracies, the feed forward signal is combined with traditional feedback control action to create a total controller output signal. The feedback controller provides *trim*. That is, it rejects those unmodeled portions of the measured disturbance that make it past the feed forward element and reach the measured process variable. The feedback controller also works to reject all other unmeasured disturbances affecting plant operation and provides set point tracking capabilities as needed.

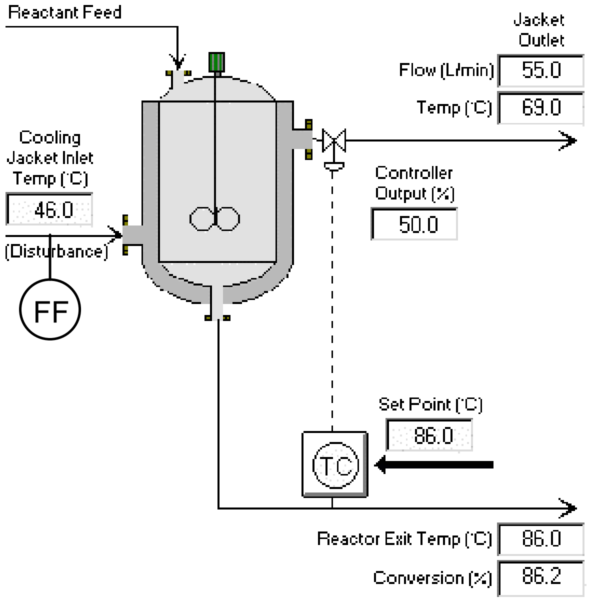


Figure 6 – Jacketed reactor with disturbance sensor for feed forward control

The control objective for this study is the same as that used in the previous two investigations. To construct the feed forward controller, the cooling jacket inlet temperature (the disturbance) is measured with a sensor as shown in Fig. 6. The signal from this disturbance temperature sensor is sent to a feed forward element comprised of a process model and disturbance model.

To account for model inaccuracies, the feed forward signal is combined with traditional feedback control action to create a total controller output signal. The feedback controller provides *trim*. That is, it rejects those unmodeled portions of the measured disturbance that make it past the feed forward element and reach the measured process variable. The feedback controller also works to reject all other unmeasured disturbances affecting plant operation and provides set point tracking capabilities as needed.

The control objective for this study is the same as that used in the previous two investigations. To construct the feed forward controller, the cooling jacket inlet temperature (the disturbance) is measured with a sensor as shown in Fig. 6. The signal from this disturbance temperature sensor is sent to a feed forward element comprised of a process model and disturbance model.

The process model describes the controller output to measured process variable behavior. Figure 2 shows the data used. The FOPDT model fit shown in Fig. 2 reasonably describes the dynamics. Hence, the parameter values from that fit are used to construct the process model of the feed forward element.

Generating disturbance driven data can be problematic for real processes if the disturbance variable cannot be manipulated at will. Control Station permits such disturbance manipulations so students can explore and learn. As shown in Fig. 7, the cooling jacket inlet temperature is stepped around the design value of 46 °C. The measured reactor exit stream temperature exhibits a clear response after each step that dominates measurement noise.

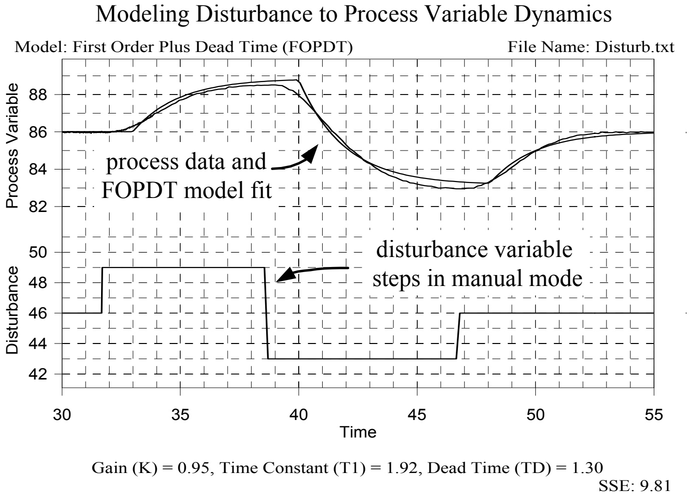


Figure 7 – FOPDT model fit of disturbance to measured process variable data

The FOPDT model fit by *Design Tools* in Fig. 7 reasonably describes the disturbance to measured process variable dynamics. Thus, the model parameters shown at the bottom of the figure are used to construct the disturbance model of the feed forward element. With a process and disturbance model identified, the feed forward element can be implemented. This is achieved using convenient parameter entry forms in the controller design menu of the jacketed reactor simulation.

The feed forward element is combined with the same PI feedback controller used in the previous base case to yield a feed forward with feedback trim architecture. Figure 8 shows the disturbance rejection performance of this controller. As in the previous demonstrations, the measured process variable (reactor exit stream temperature) is initialized at the design set point value of 86 ^{o}C. To test the controller, the cooling jacket inlet temperature is stepped from its design value of 46 ^{o}C down to 40 ^{o}C and back again.

The feed controller with feedback trim architecture performs better than the single loop case in maintaining the reactor product temperature near the constant set point of 86 ^{o}C. Specifically, while the measured reactor exit temperature deviations for the single loop PI controller range as high as high as 2.5 ^{o}C during the event, this advanced architecture limits the maximum deviation to less than 1.0 ^{o}C.

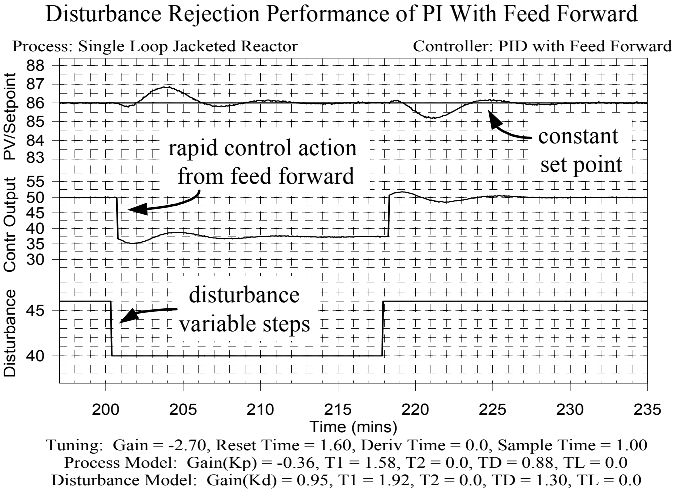


Figure 8 – Disturbance rejection in jacketed reactor under feed forward with feedback trim

As with the cascade architecture, the improved performance required an additional sensor, controller and tuning effort. As shown in Fig. 8, the feed forward controller initiated rapid compensating controller action just after the disturbance event to minimize its impact on the measured process variable. Perfect disturbance rejection was not achieved because the FOPDT models only approximate the higher order and nonlinear behavior of the jacketed reactor process.

5. Conclusion

Presented here are some examples of the lessons and challenges that a training simulator can provide. Space prohibits presentation of other studies available in Control Station, including the comparison of a P-Only, PI and PID controller in a single loop application, the study of the control of nonlinear processes, the use of the Smith predictor controller which is a simple form of a model predictive controller, the decoupling of multivariable controller interaction, and a host of process identification methods and procedures.

We stress that a training simulator is not better than nor a replacement for real lab experiences. In fact, we believe that hands-on studies with actual equipment are fundamental to the learning process.

We believe, however, that a training simulator like Control Station can provide students with a broad range of meaningful experiences in a safe and efficient fashion. These can be obtained risk free and at minimal cost, enabling students to feel comfortable exploring nonstandard solutions at their desk. We believe if a training simulator is properly designed, it can bridge the gap between textbook and laboratory, enabling enhanced learning for process control theory and practice. If the reader would like to learn more, they are encouraged to contact Doug Cooper at cooper@engr.uconn.edu, or visit: www.controlstation.com.

6. References

1. Marlin, T.E., "Software Laboratory for Undergraduate Process Control Education," *Computers & Chemical Eng.*, **20**, S1371 (1996)
2. VanDoren, V.J., "Simulation Simplifies 'What-If' Analysis," *Control Eng.*, **45**, 68 (1998)
3. Seborg, D.E., T.Edgar and D.Mellichamp, *Process Dynamics and Control*, Wiley, New York, NY (1989)
4. Ogunnaike, B. and W. Ray, *Process Dynamics, Modeling, and Control*, Oxford, New York, NY (1994)