Temperature Control of a Gas Flow

A Major Qualifying Project Submitted to the Faculty of

Worcester Polytechnic Institute

In Partial Fulfillment of the Requirements for the

Degree of Bachelor of Science

Thaddeus Adams

Ben Baranowski

Rodrigo Ma

Date: April 26th 2012

Advised by:

William M. Clark

TABLE OF CONTENTS

Table of Figures

Table of Tables

ABSTRACT

This MQP addresses the lack of laboratory process control at WPI by expanding the Gas Flow Apparatus experiment in the Unit Operations course. A LabVIEW PID controller and a vortex tube were installed in the experiment. The PID controller successfully controls the heater on the gas flow. The vortex tube separates compressed air into hot and cold streams. An energy balance was derived for the vortex tube. Classroom procedures were developed to use these devices for the Unit Operations course.

EXECUTIVE SUMMARY

Process control is important in today's world and needs to be taught to chemical engineering students. Currently, several institutions other than WPI teach process control in a laboratory setting. Since ABET has recommended that the WPI department of chemical engineering offer students more process control instruction in laboratory settings, it is in the best interest of the chemical engineering faculty to comply. PID control is one of the most common forms of process control, and it is already used for a Unit Operation experiment. This Gas Flow Apparatus experiment has been identified as being brief, and plans were in place to expand on it. By adding a PID control module and a vortex tube, it is possible to expand the current experiment to a more thorough and interesting laboratory investigation.

A PID from Omega Engineering is currently installed on the Gas Flow Apparatus, but is difficult to use as an instructional tool. The LabVIEW software is relatively easy to program without extensive prior experience. By programming a PID controller into LabVIEW it can be used to control the current gas flow heater. This Virtual Instrument will also allow students to directly modify the PID parameters. The Omega PID will not be discarded. It will be properly tuned and kept on a switching device in case the LabVIEW PID cannot be used.

In addition to PID control, a vortex tube was installed to the Gas Flow Apparatus. A vortex tube is a T-shaped tube that separates fluids by density. If the fluid is pure, the vortex tube will separate the fluid by temperature. A vortex tube requires no moving parts or power supply. This is an interesting thermodynamic study.

To gain an initial understanding of PID control, the Omega PID was tuned to study the effect of changing PID gains on temperature. Proportional band, integral time, and derivative time were all modified independently of the other variables and data was recorded. This study also attempted to program the auto-tuning function, but it was found that errors occurred too often to properly use the auto-tuning function.

Once an understanding of PID tuning was developed, a LabVIEW simulation was constructed. First, a virtual PID was built. The virtual PID was a simulation entirely contained in LabVIEW. This virtual PID was constructed to assess the feasibility of using LabVIEW as a PID in the Gas Flow Apparatus. The virtual PID also offers a medium for students to gain an understanding LabVIEW and to

experiment with PID tuning without running the risk of damaging equipment attached to the Gas Flow Apparatus. The construction of the virtual PID was made to emulate the Gas Flow Apparatus as much as possible. To achieve this, a passive cooling structure was programmed and a constant heat transfer was defined. The virtual PID also contained a graph that displayed the process variable, set point, and signal strength as a function of time. This graph allows students to see how different tuning combinations affect the approach, settling time, and steady state of the PID process, as well as determine if their tuned gains would overtax the equipment.

The virtual LabVIEW PID was a success, so a physical LabVIEW PID was constructed. Appropriate equipment was identified, purchased, and installed. This equipment allowed LabVIEW to communicate with the Gas Flow Apparatus. Two Data Acquisition Devices (DAQs) were selected for this task. The first was a NI USB-6501. The NI USB-6501 is a digital input and output device (Digital I/O) that was used to communicate to the heater. The DAQ received a signal from the PID sub-VI and transmits the signal to a solid state relay. Dependent on the signal, the solid state relay either activates and turns the heater on or deactivates and turns the heater off. The second DAQ purchased was a NI USB-9213. The NI USB-9213 is a thermocouple input device. This DAQ was used to relay the readings of all ten thermocouples on the Gas Flow Apparatus to LabVIEW. The reading from the appropriate thermocouple was fed to the PID sub-VI as the current process variable, and all temperature readings were displayed in the simulation.

Once the DAQs were appropriately connected, the VI construction began. The back panel required the NI USB-9213 DAQ assistant to be wired to the PID sub-VI. In order to attain proper PID control with the digital output device, PWM signaling had to be used. A PWM structure was built using a for loop and a case structure. The PID signal was then sent to the NI USB-6501 DAQ assistant after passing through the PWM structure. A schematic of the Gas Flow Apparatus was drawn on the front panel so that all temperature indicators could be placed where their respective thermocouples were located on the real Gas Flow Apparatus. The front panel also contains a real-time graph of the PID appropriate temperature for analysis of PID control.

During the construction of the LabVIEW PID controllers, testing on the vortex tube was performed as well. The testing was to determine if the vortex tube was an appropriate method to further expand on the thermodynamic lessons in the Gas Flow Apparatus experiment. Trials were performed, and energy balances were performed for each trial. For each trial, an energy balance was used to determine the temperature of the hot stream. This temperature was then compared to the thermocouple reading of the hot stream. The same process was used to compare the calculated cold mass flow with the mass flow value obtained from a rotometer. If the compared values were found to be close in value, then the vortex tube would be determined to be an appropriate tool to teach the laws of thermodynamics.

Both the virtual LabVIEW PID and physical LabVIEW PID can attain good control if tuned properly. The correct tuning for each differs, and both also differ from the Omega PID's optimum tuning specifications. Despite the values of the appropriate PID gains being different, the LabVIEW controllers offer an easy-to-use interface that can be used to teach students. Also, the energy balances around the vortex tube produced minimal differences between calculated and recorded variables. This justifies the vortex tube as a teaching agent because the experiment used is easily reproducible and students should be able to record good data. Classroom procedures were developed for both the PID controllers and the vortex tube.

The additions to the Gas Flow Apparatus are ready to be implemented as soon as possible. The classroom procedures developed offer a way for the WPI Chemical Engineering Department to address the lack of process control and expand on an existing experiment.

1 INTRODUCTION

Chemical engineering is a field, not only responsible for plant and process design, but also is the discipline for the optimization and proper operation of a chemical process. It is crucial that a chemical plant is designed with the proper safeguards to ensure the desired system function. This subsection of the manufacturing process is called control engineering or process control, and represents an important concept for current and future chemical engineers. Process control is a systematic means of improving the function or efficiency of operations.

Process control is defined as the control of system parameters in a specified process. This discipline is applied widely in academia and industry; examples range from home heating systems to controlling large chemical plants. Currently, the Department of Chemical Engineering (ChE) at Worcester Polytechnic Institute (WPI) offers a Process Control class (CHE 4405 – Chemical Process Dynamics and Control Laboratory), but does not offer a comprehensive laboratory component on this subject. This deficiency keeps students from experiencing how this essential aspect of chemical engineering works first hand. In order to round out the aspiring chemical engineering student's education, it is imperative to expand the curriculum to include hands on laboratory experience. Furthermore, the need to pursue the upgrade became apparent when the Accreditation Board for Engineering and Technology (ABET) visited WPI and suggested that more exposure to process control experiences be offered to the undergraduates.

The purpose of this Major Qualifying Project (MQP) was to develop a laboratory learning experience on process control. WPI currently has a Gas Flow Apparatus located in Goddard Hall, which is used as the Unit Operations experiment to teach students about compressible fluid flow. This experiment has been identified by WPI faculty as being unusually brief, and does not currently meet its full potential as an instructional laboratory tool. It is possible to add a process control component into the existing gas flow experiment, which will hopefully solve both the brevity of the experiment and also address the ABET requirement of increased process control experiences.

In order to accomplish this task, the existing gas flow experiment in the unit operations laboratory was modified. A PID controller regulates a heater in the gas flow apparatus, which increases the air temperature in the experiment. The increase in temperature causes the density of the air to change, which affects readings on various instruments downstream from the heater. Although this

PID acceptably controlled the heater, it was cumbersome to operate and was not useful for instruction. This MQP proposed that the current PID be replaced by a digital PID controller that is easier to use and a better process control learning device.

In addition, while the airflow experiment was being updated there was an opportunity to add a small and interesting thermodynamics component to the experiment. This was accomplished by installing a vortex tube, further described in the background. The main objectives of this project were:

- Digitizing the current PID controller using the LabVIEW software program
- Including a vortex tube in the circuit in order to expand the thermodynamics-based laboratory.

The completion of these objectives was finalized by designing classroom procedures and guidelines, which will enable future students to hone their process engineering and thermodynamics knowledge, skills, and abilities by having hands-on exposure on these topics.

2 Background

In recent years, process control has become an increasingly important topic for chemical engineers. This trend can be attributed to many changes in the industry, namely tougher competition, tighter environmental controls, and changes in the economy (Seborg et al., 2010, 1). It is therefore important that chemical engineering students receive at least a basic background in process control. Worcester Polytechnic Institute has historically provided an elective class (CHE-4405: Chemical Process Dynamics and Control Lab) to cover this requirement. However, ABET has suggested that students be exposed to some process control in the laboratory setting for the required Unit Operations course.

2.1 Existing Classroom Experiments

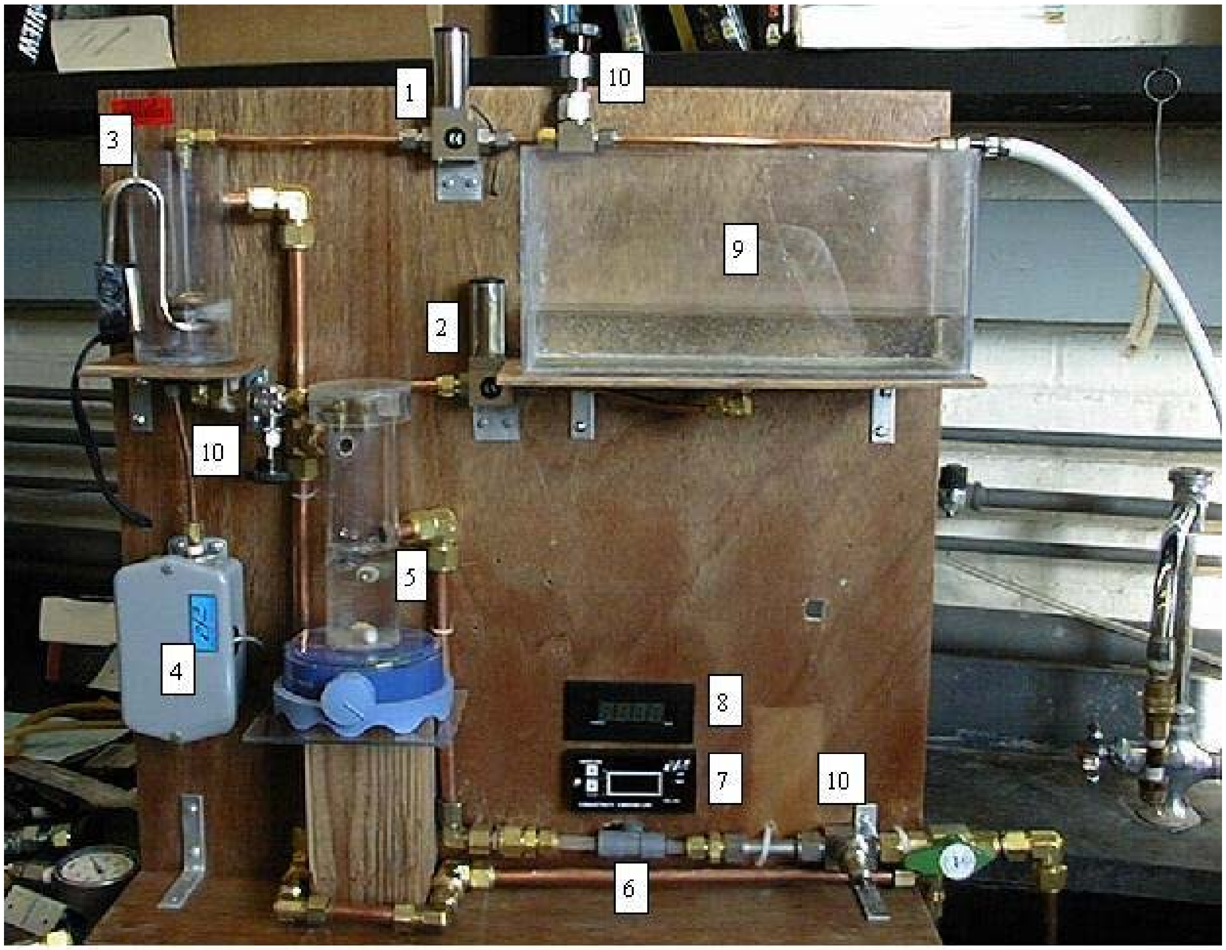
In looking for potential experiments to replicate or modify, four objectives have to be met. The experiment has to be 1) easy to replicate, 2) easy to use by students, 3) must be reliable, and 4) economical to perform. The proposed changes to the gas flow apparatus experiment have the opportunity to fulfill each of these requirements. Although a process control experiment is not currently used in WPI Unit Operations courses, other universities teach process control through various experiments. Some of these experiments are worth mentioning in this paper, as they highlight the teaching methods used by other institutions, as well as the importance placed on a basic knowledge of process control. Some of these experiments are described below.

2.1.1 RPI Studio Control Lab with PID Controller

Rensselaer Polytechnic Institute (RPI) currently uses PID controllers in a salt-water concentration experiment for chemical engineering students to study in a studio laboratory class (Bequette et. al, 2000). In this experiment, fresh water is provided on tap while concentrated salt water is stored in a tank. The fresh water flow rate is manipulated by a manual valve, while the salt water flow rate is inferred by the pressure in the closed tank. Fresh feed water is heated by an electric heater before mixing with the salt water in a continuous stirred tank reactor (CSTR). The CSTR contains a temperature probe which measures the temperature of the mixture before discharging the mixture. The effluent flows past a conductivity sensor before being emptied into a sink. The goal of the experiment is to regulate the fluid level, temperature, and conductivity of the salt water in a tank. This is accomplished by regulating the power to the heater as well as the inlet flow rates of the fresh and salt water streams.

After attempting fully manual control, the students are taught the theoretical basis of modeling, control loops, and feedback control. Students then attempt to control the process using PID controllers and also try to optimize them by trial and error. Shortly after this tedious approach, the students are instructed in tuning techniques, which are introduced in order to optimize the process control loops as the final stage of the experiment. These experiments are performed during several different class periods during the semester, and each one is scheduled between series of lectures on the appropriate topics.

The three controlled variables in this experiment (inlet flow, temperature, and conductivity) are difficult to control manually as any changes to one input radically changes the other variables. Through hands-on experience with a PID controller, the students learn the effectiveness of process control systems and their superiority over manual control. Proper tuning of the PID controller provides students with a deeper understanding of how easy it is to introduce controllers in order to radically improve the control of a process. RPI's experiment uses LabVIEW for real-time data acquisition. This program uses a simple interface and allows the student the opportunity to directly observe the experiment. The small-scale RPI experiment is shown in Figure 1.



1. Control Valve Fresh Water	6. Inline Conductivity Probe
2. Control Valve Salt Water	7. Conductivity Measurement Display
3. Heater for Fresh Water	8. Temperature Measurement Display
4. Differential Pressure To Infer Tank Level	9. Salt Water Tank
5. Temperature Probe In CSTR	10. Manual Valves For Trimming ΔP

FIGURE 1: PICTURE OF THE RPI CONCENTRATION EXPERIMENT WITH EQUIPMENT LABELS (BEQUETTE ET. AL, 2000).

While replication and testing of the RPI laboratory process is not a goal of this MQP, the RPI lab is a good example of a process control experiment that is both scientifically sound and relatively simple to operate.

2.1.2 *University of Loughborough "Learning Object" Experiment*

The University of Loughborough currently instructs their engineering students in process control via learning objects. A learning object (LO) is defined by Loughborough's faculty as a digital or web-based resource that can be used and re-used to instruct students in certain learning activities (Abdulwahed et. al, 2009). The particular learning objects used by the University of Loughborough are online laboratory learning objects (OLLO), which are used to remotely control an experiment. One of the strengths behind Loughborough's OLLO is that it cannot only control an experiment in the laboratory remotely, but can also virtually mimic an experiment based on past runs. This allows students to get an accurate first estimate of how their process will respond to different control settings before running the experiment.

The OLLO used by the University of Loughborough controls the level of water in a tank. In this experiment, both liquid flow rate and liquid levels are variables manipulated by students. Students first attempt to control the experimental variables manually, and are later taught how to use a PID controller. In this way students are first exposed to the difficulties of manual control, and then are taught the usefulness of PID control. The University of Loughborough uses their process control OLLO to teach students different types of process control depending on which academic year the students are in. Freshmen students are taught process dynamics, sophomores are taught process control and instrumentation, and juniors are taught PID control and tuning. Additional elective classes use the experiment to teach students in-depth PID theory. The entire experiment is controlled through a software interface built using LabVIEW, which gathers process data using a USB-DAQ.

By using the OLLO-based experiment, the students from the University of Loughborough can learn how to operate an experiment remotely, which prepares them later for the actual hands-on laboratory experiments. It is noteworthy that the Loughborough faculty felt it was important to include the actual experimental rig for instruction, as there is a large consensus that hands-on laboratory experiments are vital to science and engineering education (Johnstone & Al-Shuaili, 2001). The Loughborough OLLO is a good example of using two complimentary methods laboratory experiments to teach students process control.

2.1.3 "Process Control Laboratory Experience" at Villanova University

Villanova University presently conducts a "Process Control Laboratory Experiment" in conjunction with its Process Simulation and Control course. In this experiment students use a 50-gallon gravity-drained tank, which has two inlet valves (a manual and a control valve) and one outlet manual valve. A diagram of the system is shown below in Figure 2:

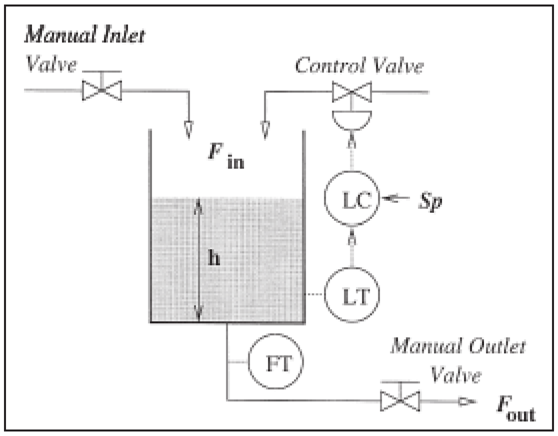


FIGURE 2: PROCESS SCHEMATIC OF THE VILLANOVA UNIVERSITY EXPERIMENT

The experiment consists of two three-hour lab sessions. During the first session, students become familiar with the equipment and perform experimental work. For this session, the tank is operated at steady-state with proportional-control only, and the liquid level set point of the tank is 50%. The students are asked to eliminate a steady-state offset in the tank level, and after some discussion on how to resolve this issue, the students are instructed to tune the proportional gain.

The second session consists of a dynamic simulation of the tank-system with a specific disturbance (chosen by students) introduced into the scenario. Although the available disturbances will not be listed, their aim is to upset the otherwise steady system. Designated roles, such as time keeping, data logging, and disturbance initiator, are distributed among the students in each group. Having recorded the necessary data, the students are engaged in a process simulation. In this phase students can predict the behavior of the tank-level over time by making appropriate assumptions and using mathematical correlations.

Through this laboratory, students are able to experience the actual response time of a physical system. Furthermore, by using the proportional-only controller, adjusting the valves manually and using the system in both open and closed-loop settings, students are exposed to a hands-on process control experience. Chemical engineering students at Villanova have claimed this laboratory is a "valuable addition to their process simulation and control education" (Muske, 2003). This feedback emphasizes the importance of laboratory experiments.

2.1.4 *University of Kentucky's Chemical Process Control Course*

The University of Kentucky currently offers a senior-level process control course with both a lecture and a laboratory component. This class is focused on offering students an inductive learning experience, where specific observations are used to lead the student to more general conclusions. Additionally, the experimental portion offers hands-on experience that reinforces the theory learned in class.

The lab component of this course uses two experiments to demonstrate the theory in practice. The first is a pressure regulation apparatus. This apparatus is similar to the gas flow experiment currently operated at WPI. The experimental setup consisted of a "pneumatic control valve, various pressure gauges, an orifice meter a square-route extractor I/P transducers, and a storage tank" (*Osei-Prempeh & Silverstein, 2010*). This setup can be connected to a control panel that includes an industrial-type PID controller.

The second device is a "Process Plant Trainer," which includes three plate heat exchangers, two feed tanks, various solenoid valves, control sensors, and thermocouples. This setup allows for the simulation of simple and complex operations that chemical engineers might encounter in industry. This setup is also connected to a control panel with a PID, which allows for stable control of the system.

There are five experiments performed on the two devices in this semester-long class. All of the experiments touched on process control. One of the experiments is centered on PID controllers and feedback in a closed-loop system; this experiment can be performed on either of the devices. The experiment itself consists of students tuning the controller to make the system either stable or unstable. Students are then able to observe the difference between a proportional and a proportional-integral control of the system by tuning the PID appropriately.

2.2 Process Control

Process Control refers to the regulation of one or more system parameters in an experiment, such as temperature, pressure or flow rate (Parallax Inc, 2006). Generally speaking, these parameters are variables that can be measured by sensors and instrumentation. The systems may range from very simple processes to much more complex systems. An example of a simple process is a household thermostat, while a more complex system may be a centralized control room operating an entire chemical plant.

For the control of a particular process a specific attribute is often monitored using a sensor. This could be a pressure gauge, thermometer, manometer or other measurement device. Measurement devices are often called "transmitters," as they take the physical phenomenon as measured by the sensor and change it into a useful signal (Seal, 1998, xvii). The signal from this sensor is relayed via the transmitter, which uses pneumatics, electrical impulse, or other signals, to a controller that reads the information and compares it to a *set point* or target value. If the reading deviates more than a prescribed range, the controller will command an action to reach the set point, such as opening or closing a valve. The device that responds to the controller is termed the "control device."

Based on the set point, the controller will drive a device to as close to the desired value as possible. More complex controllers offer a greater degree of modulation with the aim of maintaining more precise control on the system. Controllers that can vary the strength of the control device response were classically termed "throttling" controllers (Ziegler et. al., 1942). These devices will sample at a higher frequency and react more rapidly than other commonly found devices.

As a representative example, consider the cooling system for a room. A person inside the room sets the temperature for the Air Conditioning Unit (AC). Typically this is done by adjusting the setting on a thermostat sensor. On a hot summer day the occupant may lower the set point temperature to cool down the room. This system set point is then compared with the temperature measured by the thermostat's sensor. If there is an appreciable difference between these two values, the AC will work to drive the temperature to the set point.

In this example, the main process variable is temperature, and the control system will require a sensor that monitors temperature and provides feedback to the controller. If the temperature sensor reads a temperature different from the set point, it will drive the cooling equipment to get it to the target. The cooling system will accomplish this by delivering cold air to the room until the thermostat reaches the prescribed temperature set by the occupant. If the temperature of the room is equal to or lower than the set point, the controller will stop the flow of cold air. This control method is referred to as *Bang-Bang* control (also called On/Off control and cyclic control), where a device can only be fully on or fully off at any given time, and is one of many possible control scenarios. Figure 3 is a simple block diagram that illustrates this example of process control:

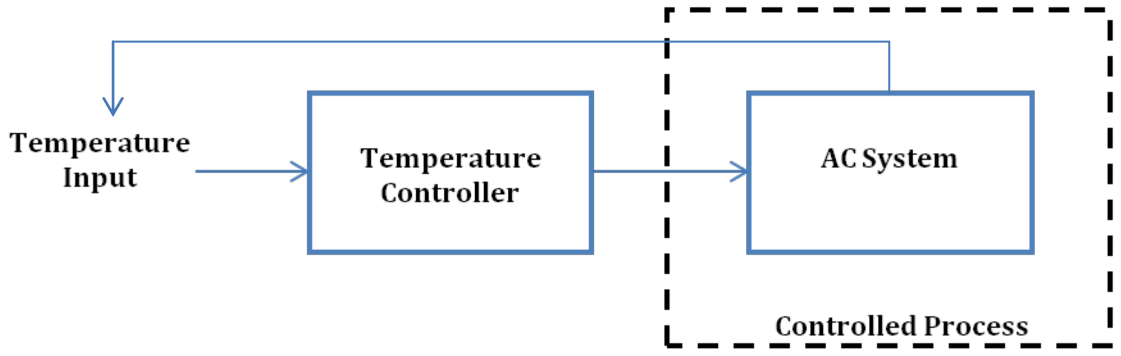


FIGURE 3 BLOCK DIAGRAM DEPICTING THE COOLING SYSTEM FOR A HOUSE

Figure 3 shows that once the set point is defined, the sensor will force some response from the AC controller. Every control system is made with the same general components; a sensor, a controller, and some type controlled equipment.

2.2.1 Types of Control

In process control, the type of control is often defined by its *control loop*. The basics of a control loop have been developed in the previous example. A control loop is a concept involving a process, a controller, a control element, and a sensor. The sensor transmits data to the controller, which forces the control element to change its operation, which creates an effect measured by the sensor. If this cycle is continuously repeated, each element can be symbolically represented as a circular loop. The simple example of a bang-bang control loop has been described in the example given with Figure 3. Many different types of control exist, with several explained in the following paragraphs.

A *feedback loop* is a loop where the sensor takes a measurement of the process variable (sometimes specified as the "control variable") after it has been affected by the process (Seborg et al., 2010). The measurement is then sent to the controller, which creates a corrective action if necessary.

Feedback loops have many distinct advantages in process control, including the inherent ability to reduce sensitivity and the ability to provide a corrective action regardless of the source of disturbance. The control described for Figure 3 is a feedback loop.

Feedforward loops, by contrast, are loops where the sensor takes a measurement of the process variable (sometimes specified as the "disturbance variable" or the "input variable") before it has been affected by the process. For example, in a vessel with inlet feed, a feedforward loop would attempt to correct any problem with the inlet feed before filling the vessel. Feedforward control allows for corrective action before large deviations occur in a process. One limitation of feedforward control is that it can at times result in *offsets*. An offset is when a process variable is held steady at a value other than the set point. This is because feedforward control monitors the process variable before it is affected by the control device. In doing so, the controller cannot always properly estimate the difference between the current value of the process variable and the set point.

It is possible for feedback and feedforward control loops to be combined into more complex control loops. However, this is not common. A more complex controller can become expensive and can occasionally require more oversight from a technician. Often times an inexpensive feedback or feedforward loop will accomplish the same task to satisfaction.

A widely used type of control is *Proportional, Integral, and Derivative control* (Wescott, 2000). Often abbreviated to PID control, this control system is used because of its simplicity, stability, and adaptability. The majority of loops in industry are feedback loops, and a large percentage of these feedback loops are PID loops (Astrom & Hagglund, 2001). It has been estimated that over 90% of all control loops are PID loops. In some fields, the estimated figure is even greater (Desborough & Miller, 2002). It is therefore necessary to instruct chemical engineering students in the basics of PID control to better prepare them for industry. It is this method of control that is the primary focus of this MQP, and is the method that will be taught to students in the Unit Operations class should this project's conclusions be implemented.

2.2.2 PID Control

The term "PID Controller" stands for Proportional, Integral, and Derivative Controller. In this case the nomenclature is straightforward; the controller is made up of proportional, integral, and derivative mathematical components (University of Michigan, 2009). The three main equations in their simplest forms are;

Proportional:
$$X = K_C(x_{set} - x) * t + c \tag{2.1}$$

Integral:
$$X = \frac{1}{\tau}\int_0^t (x_{set} - x)(t)dt \tag{2.2}$$

Differential:
$$X = \tau_d \frac{d}{dt}(x_{set} - x)(t) \tag{2.3}$$

Where:

- X is the parameter of the system that can be controlled
- x is the current value of the manipulated variable
- x_{set} is the set point of the manipulated variable
- t is the time elapsed,
- c is a proportional constant,
- K_c is the proportional control gain,
- $1/\tau$ is the integral control gain (also called integral time), and
- τ_d is the derivative control gain (also called derivative time) (Omega, n.d. b).

The proportional, integral, and derivative control gain terms are generally adjusted by the user if the controller is not responding correctly or if the system is not behaving as desired.

The three terms are added together to form the main PID equation:

$$X = K_C(x_{set} - x) + \frac{1}{\tau}\int_0^t (x_{set} - x)(t)dt + \tau_d \frac{d(x_{set} - x)}{dt} + c \qquad 2.4$$

The driving force in each equation (2.1-2.3) is a given difference between the set point and the variable's current value (x_{set} – x). The gain term in each of these equations is multiplied by the driving force. They therefore determine the relative contribution of each of the PID equations (Equations 2.1-2.3) to the controller's final response to a deviation, as shown in Equation 2.4. It is these gain terms that are manipulated by the user to tune the controller.

The difference from the set point (x_{set} – x) is often abbreviated as the error (e). This simplifies to Equation 2.5:

$$X = K_C e(t) + \frac{1}{\tau}\int_0^t e(t)dt + \tau_d \frac{d}{dt}e(t) + c \qquad 2.5$$

This equation is called the "ideal" PID equation. However, this theoretically derived equation is often altered in industry to a different form (Bequette, 2003, 168-211). In a commonly used form, the proportional gain is multiplied by every other term and the constant gain (c) is left out:

$$X = K_c\left[e(t) + \frac{1}{\tau}\int_0^t e(t)dt + \tau_d \frac{de(t)}{dt}\right] \qquad 2.6$$

The reason for this change is to increase the contribution of the proportional term, which is almost always the most important term in throttling type controllers (Ziegler et. al., 1942). Proportional control alone is occasionally adequate and can be commonly found in industry, but derivative or integral control without proportional control is often an inappropriate control setup. Equation 2.6

prevents control without a proportional term defined. Exclusion of the proportional term is often an incorrect method of control, but possible using Equation 2.5.

There are many reasons to use a PID controller in a system. The main reasons are:

1) They can be used whenever a process is difficult to manually control,
2) They can control a system that is too dangerous for manual control,
3) They are accurate enough to control a process that requires great precision,
4) They can control a complex, multi-variable process (Wescott, 2000).

The proportional term in the PID equation allows the controller to respond according to the magnitude of the deviation from the set point. In other words, if the current deviation from the set point were to be large, the controller would have a proportionally large response, forcing the manipulated variable to approach the set point (Visioli, 2006, 3-4). Proportional control is especially useful as the proportional term can greatly increase the controller's speed of response. However, using proportional control alone will result in offsets oscillating around the set point. Increasing the proportional gain will result in much smaller offsets, but does not always yield good control (Love, 2007, 151).

A good controller forces the system to reach the set point quickly and remain stable at the set point. One method of determining if a controller's performance is acceptable is to examine a graph of the process variable over time. This graph generally records the change in the process variable from system startup until sometime after the process reaches the set point. It is important to evaluate this to be able to gauge the efficiency of the system to achieve stability under the prescribed PID gains.

The proportional gain term (K_c) is a user-modified parameter that directly affects how powerfully the controller will respond to an error. Figure 4 shows a number of different response curves:

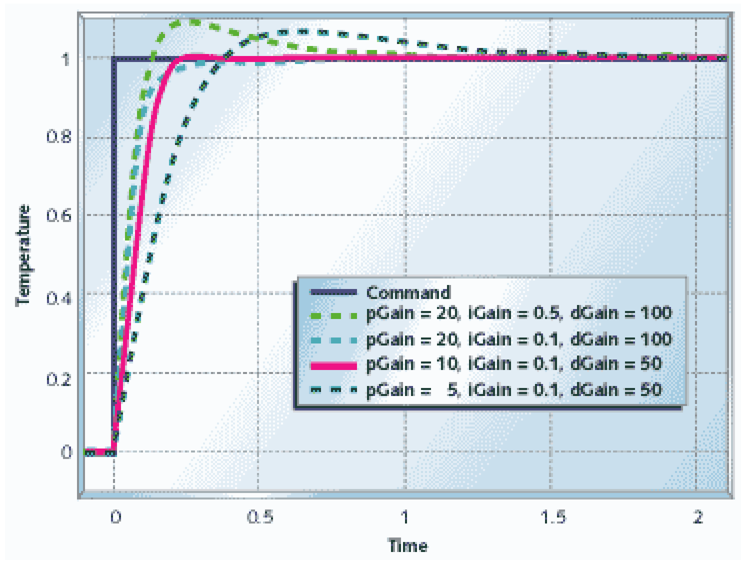


FIGURE 4: PROCESS VARIABLE RESPONSE TO VARYING PROPORTIONAL GAINS (WESCOTT, 2000).

When a small K_c value is used, the controller response is weaker, as shown by the dashed blue line in Figure 4. This results in a process that approaches the set point but slower than an ideally controlled process would. When a slow approach is unacceptable, the proportional gain is increased to speed up the initial response of the system. However, a high value of K_c could possibly result in overshooting the set point; which may be unacceptable for the process, as shown by the dashed green line in Figure 4. The desired value of the proportional gain generally lies between the two extremes; a value at which it approaches the set point rapidly, but does not overshoot the target too greatly. The best representation of this can be seen as the solid pink line in Figure 4.

The integral term in the PID equation allows the controller to anticipate errors that have already occurred. If the system has deviated from the set point multiple times in the same manner, then the integral term will force the controller to respond more powerfully to that previous deviation than to a new deviation. Figure 5 shows this:

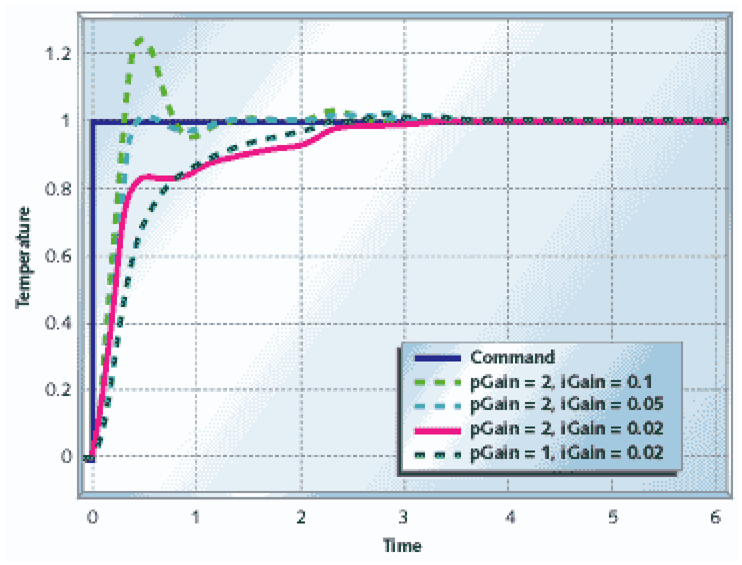


FIGURE 5: PROCESS VARIABLE RESPONSE TO VARYING INTEGRAL GAIN TERMS (WESCOTT, 2000)

The largest benefit from the integral term is that it allows the controller to eliminate the oscillations inherent in the proportional term. In doing so, the integral term adds long-term stability to the controller. One problem that can result from integral control is that it can destabilize the system by "remembering" errors from the past that are no longer occurring. This issue can be solved by properly tuning the integral gain term. System stability is important as it results in process reliability.

If the integral gain term ($1/\tau$) on a PID controller is too small (τ is too large), the controller will take too long to reach the set point. An example of this can be seen in Figure 5, by the dashed blue line and the solid pink line. Eventually the system should reach steady state at the set point; however the amount of time this takes is unacceptable. A different problem occurs if the integral gain term ($1/\tau$) is too large. This is shown in Figure 5 as the dashed green line. An ideal setting for the integral gain term should reach the set point quickly but also must not overshoot or oscillate around the set point.

Another problem with integral term is a phenomenon referred to as *integrator windup*. This is illustrated in Figure 6 below:

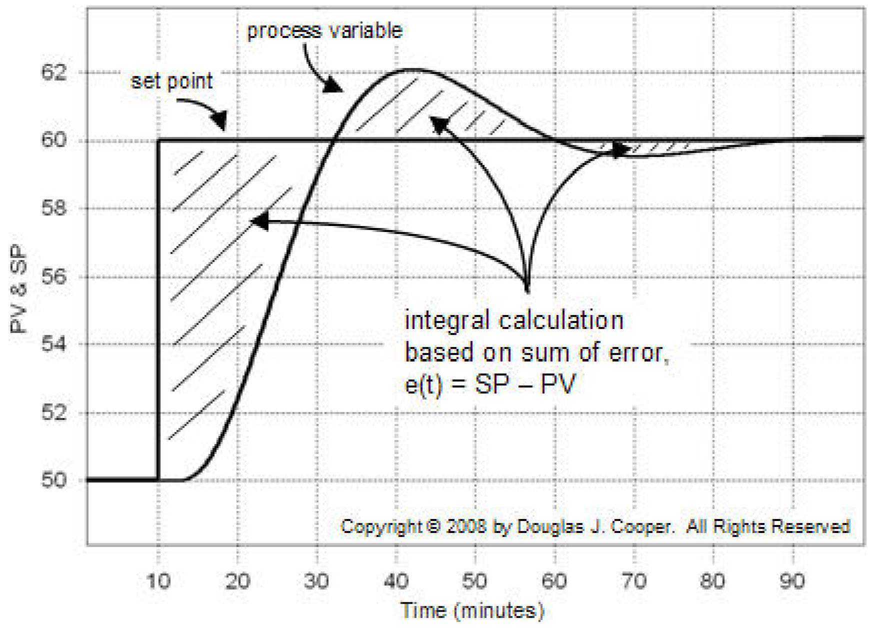


FIGURE 6: PID PROCESS VARIABLE RESPONSE CURVE SHOWING INTEGRATOR WINDUP (COOPER, 2008).

Integrator windup occurs when the integral term tries to force the controller to correct an error faster than the control device can respond. As the controlled variable takes longer to correct the error, the controller will translate this lack of response as a need for a larger correction. During startup, a controller suffering from integrator windup will attempt to run the controller at an unreal maximum output as the controller tries to correct a false error. Once the set point is reached, the controller has already been subject to windup, and as a consequence, the controlled variable will overshoot the set point significantly. Furthermore, it will take more time than usual for the controller to decrease the process variable to the set point. This error can be avoided by using sufficiently powerful equipment, as it will be difficult to hit a maximum output, thereby lowering the chance of integrator windup.

It is possible to only use the proportional and integral terms to form what is called a PI controller (Wescott, 2000). PI controllers generally control a system well, but are still susceptible to integrator windup. PI control can also fail due to the integral term's historical errors. In general, when defining a PI controller it is often necessary to introduce limits of integration and the range of the integrator to avoid these problems.

The main reason for including a differential term in a controller is that a derivative expression reduces problems inherent in PI control. Including a derivative term forces the controller to respond to deviations in proportion to their speed, as the derivative term "sees" the current slope of the line required to reach the set point. The derivative term on its own does not result in a controlled process, it merely resists change and slows the response of the controller. In doing so, the derivative expression in the main PID equation resists the integral's tendency to destabilize the system.

The main weakness with including a differential expression is the occurrence of a phenomenon called *derivative kick*. Any change in the set point of the controller will result in an infinite slope at that time it is changed. It is possible to observe this when using Equation 2.3, as a new set point will clear the time history and the dt term will be equal to 0. This will create a sudden, sharp increase, or "kick," in the controller's output. This sudden increase is usually brought back to a normal set point by the proportional term. Derivative control is also sensitive to noise, as an improperly tuned PID cannot respond as quickly to small changes. Noise occurs when the signal given by the transmitter is not accurate enough for the range prescribed by the controller. This causes the transmitter signal to jump in and out of the controller range, and the controller cannot establish a consistent response, as it sees a rapid and random deviation. Since noise is generally a high frequency signal (noise has large positive and negative slopes), the derivative control responds to the noise and can introduce instability to the process.

One of the challenges with tweaking the derivative gain term is limiting the amount of derivative kick when the system is subjected to a disturbance or change in the set point. If the derivative gain term is too large the controller will pick up the instrument noise, which will be reflected in the behavior of the manipulated variable. A large derivative gain term will also be sensitive to derivative kick. A small derivative gain term will not contribute much to the PID controller, and the controller may begin to show some integral destabilization. Again, achieving ideal behavior relies on picking a derivative gain term that is between the two extreme conditions.

Due to the differential term, PID controllers can be noise sensitive (Wescott, 2000). However, the main symptoms of poor tuning – overshooting, oscillations, noise sensitivity – are generally easy to spot if graphed against a constant set point. Proper tuning based on manual manipulation, or even better, a theoretical basis, should result in acceptable process control. One benefit of PID control is the ease at which the controller can be tuned without a dedicated mathematical model. The main advantage to PID controllers is the long-term stability offered by using the proportional, integral, and derivative terms in concert with each other.

2.2.3 PID Tuning

Tuning a controller is generally one of the more tedious tasks of PID control. PID tuning can be divided into two categories; mathematical and experimental. Mathematical tuning is often avoided due to the numerous combinations of complex equations (Ziegler et. al, 1942). Some of the auto-tuning functions performed by computers use a selection of these mathematical equations in their algorithms (Ming-da & Xin-jian., 2006). Experimental tuning can be performed by an established systematic method, or can be accomplished using a "hit-or-miss" procedure, where the experimenter adjusts the PID gain terms until one combination works. A common drawback to the hit or miss approach is that it can take a long time to reach the ideal setting or "sweet spot."

Experimental tuning "rules-of-thumb" for PID control systems were first suggested by Ziegler and Nichols (1942), and have since been simplified and expanded upon by many other researchers. This has resulted in other generally accepted tuning rules such as those created by Tyreus and Luyben (O'Dwyer, 2006, 78). The Ziegler-Nichols tuning rules involve running a handful of trial experiments where the proportional gain of a controller is increased until permanent oscillations are observed. Once this maximum or "critical" value of proportional gain is observed, each resulting gain term (proportional, integral, and derivative) is multiplied by a constant. The exact value of each constant depends on if the controller functions as P-only, a PI, or a PID. The result is generally a good "first guess," and if successful, the amount of manual tuning afterwards will be greatly reduced or eliminated entirely. Problems with the Ziegler-Nichols tuning rules have been identified, and much more complex solutions have been proposed (Astrom & Hagglund, 2004), but given the simplicity of the gas flow apparatus used in this project, it can be expected that the Ziegler-Nichols rules apply.

While the Ziegler-Nichols tuning rules give a good first solution, they do take time to perform. It was determined that the time it takes to instruct students in the theory and application of these rules-of-thumb outweigh the benefits of teaching them to students in the Unit Operations Laboratory. The tuning rules are mentioned here in case they are later determined to be valuable for student education. For the remainder of this study, tuning rules are considered to be outside the scope of this MQP.

2.3 LabVIEW

The LabVIEW software program is a visual programming application created by National Instruments. It is programmed in G, a graphical programming language, and is used to create "Virtual Instruments," or VI's. These VI's can be programmed to interact with instruments and control devices in real time, or they can be used to simulate the behavior of a system virtually.

The main layout of LabVIEW consists of two windows that can be edited. These windows are termed the "Front Panel" and "Block Diagram." The front panel is the Graphical User Interface (GUI) that is commonly found in distributed software packages. This window generally contains all of the necessary graphs, icons, and indicators to inform the user about what is happening to the system being studied. The front panel also often contains numerical inputs, controls, and other devices to allow them to manipulate the system to achieve the desired behavior.

The block diagram of LabVIEW is the area where the graphical code is manipulated (National Instruments, 2010). It is named the block diagram because the window appears to be a simplified diagram often drawn by engineers to quickly communicate the basics of a system. In the block diagram, common programming operations are represented by different icons. These icons can represent mathematical operations, loops, arrays, strings, and many other different types of operations.

2.3.1 Constructing a VI

Each programming operation in LabVIEW is connected with "wires," which are visually representative of how each operation is interdependent. Data moves down the wires from input to output of each operation. Wires are categorized into several types (string, numerical, double integer, dynamic data, et cetera), depending on which particular type of operation is being performed. These wires are all colorized depending on their data type. The wire colors used in this paper VI's are yellow (double integer), blue (numerical & dynamic data), and brown (cluster data). Wires can also be textured to aid the visual distinction between them. Ensuring the compatibility of wires is vital to creating a working VI.

LabVIEW works by using independent mathematical or programming functions. These are displayed as icons on the block diagram, and generally have a number of terminals at the left and/or right of them. These terminals define the functions input and outputs, and are generally where the required arguments are wired to the function. This is shown in Figures 7 and 8.

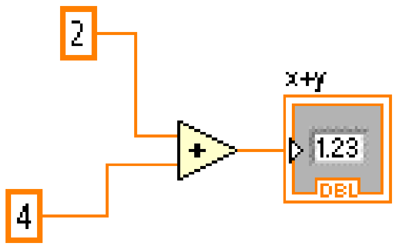


FIGURE 7: A SIMPLE OPERATION IN LABVIEW'S BLOCK DIAGRAM WHERE THE NUMBERS 2 AND 4 ARE ADDED TOGETHER. THE SYMBOL TO THE RIGHT IS AN INDICATOR, WHICH APPEARS ON THE FRONT PANEL.

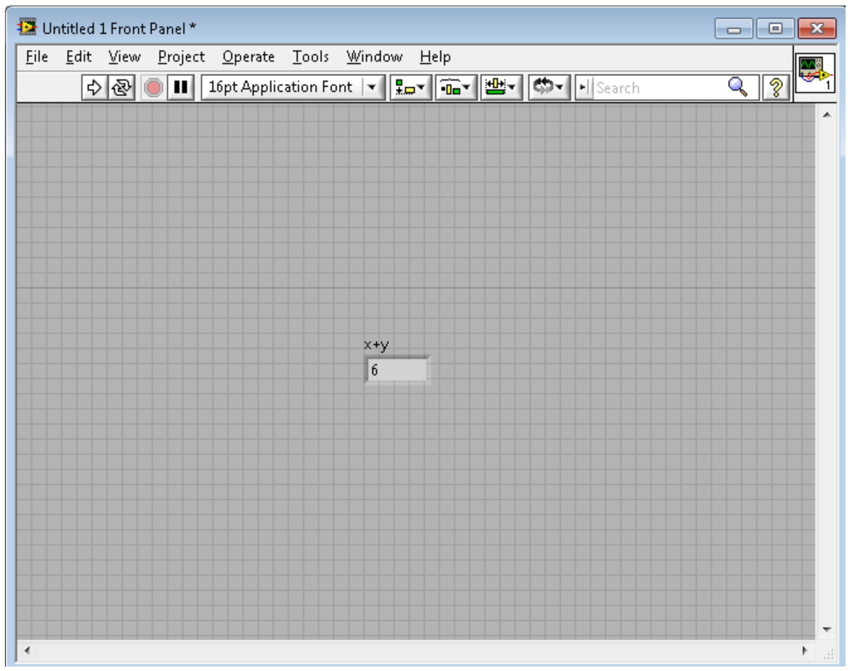


FIGURE 8: THE SAME SIMPLE OPERATION AS SEEN THROUGH THE FRONT PANEL. INITIALLY, ALL UNKNOWN VALUES ARE DISPLAYED AS 0. THE SOLUTION WILL BE CALCULATED WHEN THE RUN BUTTON (THE ARROW SYMBOL, LOCATED IN THE TOP LEFT ON THE LOWER TOOLBAR) IS PRESSED.

Some programming functions are of particular interest to this project. This includes some programming loops, which in LabVIEW are displayed as a square structure with a blank interior.

Functions placed inside these loops are only active when the loop is active, and can be turned on or off by certain conditions. One of these programming loops is the *while loop*, which is controlled by the user via a conditional button on the front panel. A while loop runs certain programming functions continuously until a specific condition is met. For the purposes of the VI's developed in this project the "stop if true" condition was used, which adds a button labeled "STOP" to the front panel. This button is a terminal condition to the while loop, and activating it stops all calculations under the while loop's control. This condition is important because until the button is pressed, the heater in the gas flow will be active. The *for loop* is another important programming loop. The for loop executes its interior functions a set number of times every time the VI is run. A *case structure* was also used, which is a way to perform different mathematical operations depending on whether or not a set loop condition is true or false. Case structures are used when calculations need to be changed based on certain conditions of the overall program. A "true" condition will activate one set of calculations, while a "false" condition will use a different set of calculations.

Another LabVIEW function important to this project is the *cluster*. A cluster is a function used to group several different data types together. This grouping reduces the number of wires and allows the values of each data source to be manipulated and recorded together. For this project, the cluster function is especially useful for graphing many different data sources together on one chart. Also important are property nodes, which can force a function to behave in a certain way. *Property nodes* of a function are created in reference to a single function and can control many details about it. For example, property nodes are used in one of the VI's produced for this project to clear the data history of a graph.

The LabVIEW software package comes with many such functions, but occasionally operators of the equipment find that more complex functions are required. For this project, the optional Control Design and Simulation Module was required, and was installed on a WPI-owned laboratory computer. A description of this module can be found on the National Instruments website (2012). There is much more that can be said about LabVIEW's functions, such as branching wires and creating complex programming arguments. However, many of these functions are outside the scope of this project.

After the VI has been completed, the programmer typically locks out the editing feature before handing it over to the user. This action ensures that the user will not be able to accidentally alter a VI, which could potentially corrupt the file and render it unusable.

LabVIEW is most useful when Data Acquisition Devices (DAQ's) are used in conjunction with it. DAQ's are real-world instruments which monitor and record various system parameters such as temperature, pressure, or voltage. A DAQ can also be used to output signals to the system, which is especially helpful for process control. This project uses 2 DAQ's. One is an NI USB-9213, which is a 16 channel USB thermocouple measurement device. The other is an NI USB-6501, which is a 24 channel low-cost USB digital input/output device. The 9213 collects thermocouple data while the 6501 sends a signal to the Omega AHP-5051 inline heater.

2.3.2 SIGNALING

In order for LabVIEW to control a device, it must send either an analog or digital signal via a DAQ. The PID sub-VI used for this project does not require any additional alterations to their export signals if an analog device is used. However, if a digital device is used, certain changes must be made to the export signal. The required changes are described below.

In electrical engineering, different types of signals can be described by their waveform. While in-depth descriptions of signals are outside the scope of this project, a brief overview is necessary in order to understand some of the basic methods used to control the LabVIEW PID. Most signals from electrical devices are the *analog signals*. The waveform of analog signals is sinusoidal, and the average voltage from an analog signal can be found from waveform characteristics, such as the amplitude or phase of the signal (Bucknell, 2012).

Digital devices are usually not powered by analog signals, as analog signals cause *noise*. Noise is random fluctuations that cause the signal to deviate from a perfect sinusoidal signal. Because of this digital devices are not usually powered by analog signals. They are instead powered by *digital signals*, which work on time-based *pulses*. A pulse is an electrical signal that is not sinusoidal, but is flat and constant at a certain value. In computer science, this value is either on or off, where on is represented by a 1 and off is represented by a 0 in binary code. Pulses and digital signals are more useful for transmitting information, as the voltage from a wire can be immediately determined if it is known that that wire is on or off (Bucknell, 2012).

PID control is designed for analog signals, as the average voltage of an analog signal can change by varying either the maximum voltage (amplitude) or the time it takes to complete a sine cycle.

However, this cannot be done with digital signals, which are designed to pulse at only one voltage level. In order to change the average voltage of a digital signal it is necessary to change the length of time the signal is pulsed.

2.3.3 Pulse-Width Modulation

Pulse-width modulation (PWM) is a technique used in digital signaling to control the power output of a device. PWM accomplishes this by sending out a certain amount of pulses throughout a *duty cycle*. A duty cycle is defined as the ratio between the time the device is active and the time the device is inactive (Massa, 2005). A continuous process runs through duty cycles for as long as necessary. The duration of a duty cycle is defined as a *PWM period* (Massa, 2005). The length of the PWM period is determined by the programmer, and the PWM periods are uniform throughout the process. This type of signaling is useful when it is necessary for a digital signal to mimic the behavior of an analog signal, such as in a PID control setup. An example of PWM is shown in Figure 9:

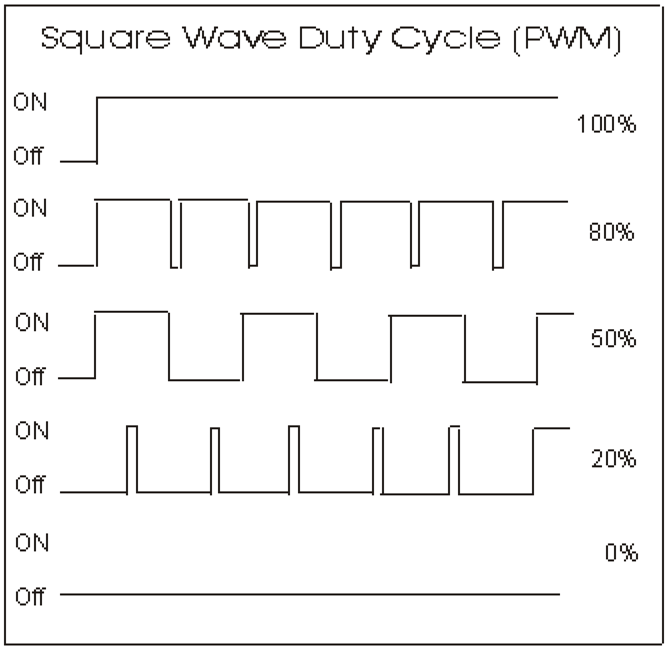


FIGURE 9: GRAPH OF A SQUARE WAVE PWM DUTY CYCLE FROM INSTRUMENTATION COMPANY (SCIENTIFIC INSTRUMENTS, 2007).

For example, consider if the desired voltage of a signal output is 10 volts, but the digital device can only output 20 volts when active. In this example the PWM period is selected to be 100 milliseconds. The PWM signal would administer maximum voltage (20 volts) for the first 50 milliseconds, and then would not administer any voltage for the remaining 50 milliseconds of the PWM period. The controlled device would see an average of 10 volts over the PWM period, and operate as if it had received a 10 volt signal throughout the whole PWM period (Technical Educational Institute of Chalkis, 2001). The PWM periods are very short, allowing the control to change duty cycles quickly to adjust to changes in the required voltage smoothly.   Figure 10 shows how a PWM signal can be generated from a comparable analog signal.

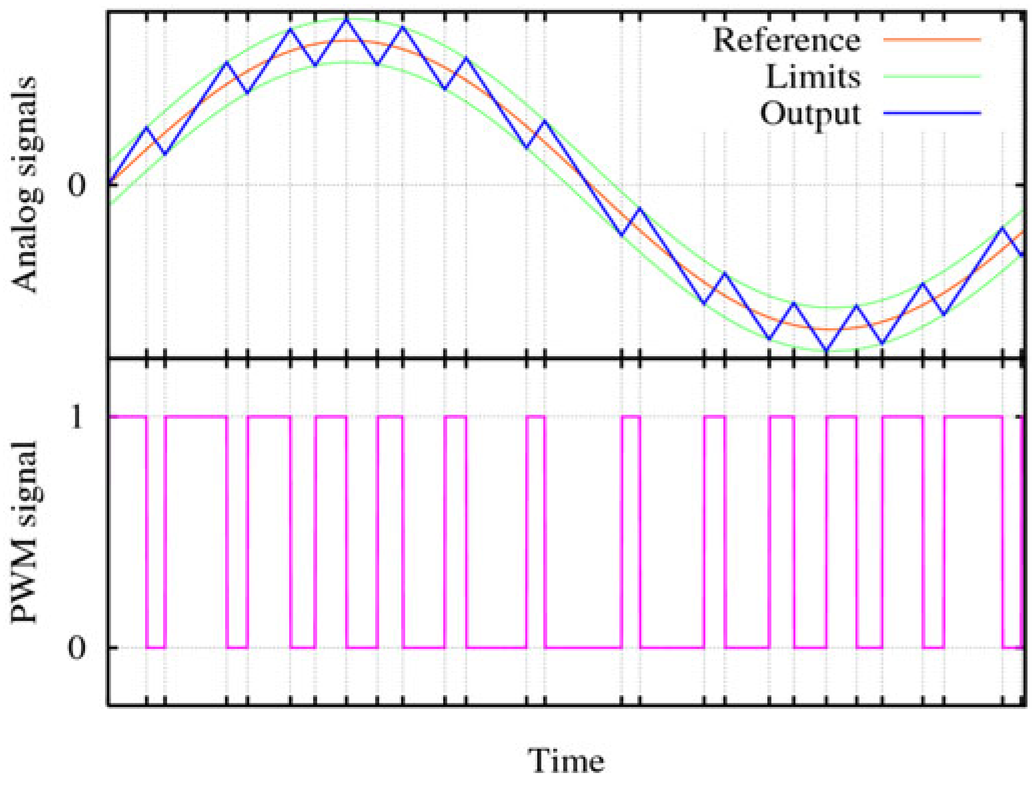


FIGURE 10: GENERATING A DIGITAL SIGNAL WITH PWM FROM AN ANALOG SIGNAL (ACRONAME ROBOTICS, 2012)

One of the disadvantages to PWM is that under large voltages, such a rapid change from "ON" to "OFF" will require a great amount of resistance and can be damaging to the equipment. It is possible to control large voltages by using smaller voltage and a solid state relay, which basically acts as an ON/OFF switch between a high voltage device and a low voltage control device (Motion Systems Design Staff, 2000).

This type of signaling can be used with an analog-based PID controller. With PWM signaling, the PID signal can be interpreted correctly by the digital device. With proper tuning of the PID the device will turn on and off rapidly while approaching the set point, allowing for smooth control.

2.4 Vortex Tube

The WPI chemical engineering department has recently purchased a Model 20025 vortex tube from AiRTX International. This vortex tube has been added to the gas flow apparatus, but has never been used in a Unit Operations laboratory. Vortex tubes take in compressed air and separate the air into hot and cold streams. Thermocouples currently record the inlet, outlet cold, and outlet hot stream temperatures. The *cold fraction* can be changed by varying a control valve on the hot stream exit. The cold fraction is the ratio of the cold mass flow to the hot mass flow. Cold fraction is examined in greater detail later in this document.

The vortex tube, sometimes referred to as a Ranque-Hilsch tube, was discovered by Georges Ranque and was later examined and described in detail by Rudolf Hilsch (Gao et al., 2005). Although the calculations predicting the behavior of the airflow inside vortex tubes can become quite complex, the basic theory is relatively simple. The tube itself is generally T-shaped, with a spiral coil spanning the length of the top of the T and the air inlet located at the bottom of the T. This coil is referred to as the swirl chamber. The inside diameter of the coil increases towards one end of the tube, out of which hot air is exhausted. This end of the vortex tube is referred to as the "hot end." A control valve is located after the coil in order to vary the flow rate of hot air. The opposite, narrower end of the coil becomes the cold end, as cold air blows through it. Figure 11 displays a simple sketch of the vortex tube currently installed in the Unit Operations Laboratory.

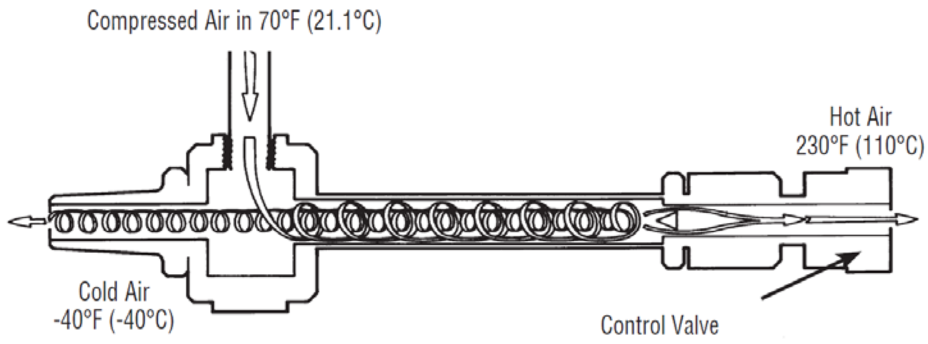


FIGURE 11: SKETCH DISPLAYING AIRFLOW THROUGH AN AIRTX VORTEX TUBE (AIRTX INTERNATIONAL, 2007).

As pressurized air is fed into the vortex tube inlet, the coil forces the faster molecules to flow in a spiral around the outside of the coil, towards the wider end. Since the faster moving molecules will have more thermal energy, the tube on the same side as the wide end of the coil will have air exit at a higher temperature than that of the inlet. The suction created by the faster moving molecules forces the slower moving molecules to flow in the opposite direction towards the narrow end of the coil. The relatively small amount of thermal energy in the slower moving particles translates into a lower exit temperature than the inlet flow (Hilsch, 1947). These flow patterns are characteristic of a vortex, which is why the equipment earned the label of "vortex tubes."

The exit temperature at the cold end of the vortex tube depends on the inlet pressure and the airflow based on the amount that the control valve is open. In general, to change exit temperature of the cold end the pressure is held constant while the control valve on the hot end of the vortex tube is manipulated. The shift in temperature from the inlet to the cold end is thermodynamically described as the cold fraction. The cold fraction can be found through a simple energy balance around the tube, and is a measure of the efficiency of refrigeration. The cold fraction (ε) is given by the following equations (Gao et.al, 2005);

$$\varepsilon = \frac{\dot{m}_c}{\dot{m}_h} \tag{2.7}$$

$$\varepsilon = \frac{T_h - T_i + 4}{T_h - T_c} * 100 \tag{2.8}$$

Since vortex tubes have no moving parts and use no electricity, they are often used as *spot cooling* solutions for manufacturing processes. Cooling in manufacturing is usually done with liquid coolants, but not every process can use these methods. Spot cooling is cooling from a point source, and is one solution in manufacturing when the use of a liquid coolant is inappropriate. Because vortex tubes require no electricity they also have an advantage over other devices where the possibility of an electrical fire is of concern. An application where vortex tubes offer a distinct advantage is when combustible substances are involved. Another example of this technology's application is the use of vortex tubes in refrigerated suits for workers in foundries or other extreme heat environments.

2.5 GAS FLOW APPARATUS EXPERIMENT

The previous gas flow apparatus experiment instructed students to analyze the effects of operating pressure and mass flow on the pressure drop through the system. There was also a section that required students to analyze the effects different temperatures would have on the compressible flow. Lastly, students were required to perform an energy balance around the heater on the gas flow apparatus. A schematic of the gas flow apparatus can be found in Appendix 6.3.

In the Unit Operations Laboratory, an Omega cn9000A PID controller is currently located on the gas flow apparatus. The controller is used to regulate the amount a heater changes the air temperature within the airstream. The heater used in the unit operations laboratory is a powerful device (with a maximum output of 400 watts). When the heater is operating at full power it can easily surpass the maximum operating temperature of the Coriolis meter attached further downstream. For this process, manual control is difficult and could result in an unacceptable operating condition, or in the worst case a complete system shutdown. PID control allows the system to be stable for suitable study and safe for the equipment.

2.5.1 OPTIMIZING THE OMEGA CN9000A

Tuning the Omega cn9000A PID controller is an arduous task. The nature and equations of a PID controller have been previously described, but are displayed again below for reference:

Proportional: $X = K_C(x_{set} - x) * t + c$ (2.1)

Integral: $X = \frac{1}{\tau}\int_0^t (x_{set} - x)(t)dt$ (2.2)

Differential: $X = \tau_d \frac{d(x_{set}-x)t}{dt}$ (2.3)

Students in the Unit Operations course should never have to optimize the Omega PID. The following sections describe how one would optimize the Omega PID. This is necessary to understand the experimentation that was done to tune the controller.

There are a number of parameters within the Omega PID controller that can be fine-tuned to produce an "optimized" process control. This research will investigate how the PID parameters affect system dynamics (Zhong, 2006), such as:

1. *Rise time*: Defined as the time it takes for the output value to reach 90% of the set point value for the first time.

2. *Overshoot*: Expressed by how much the highest output value is different to the set point value, normalized against steady-state.

3. *Settling time*: Defined as the time it takes for the system to converge to steady-state.

4. *Steady-state error*: Defined as the difference between the converged steady-state output and the desired set point value.

These attributes represent the four major characteristics of a closed-loop step response system. Within this experimental framework, it is possible to use these concepts as guidelines to evaluate how effective the LabVIEW and Omega PID controllers are.

2.5.2 Explanation the Omega PID controller

This section offers a thorough explanation of the Omega PID system, as students will still be able to use it, even after the LabVIEW PID is installed. Figure 12 below shows the front view of the controller and labels the main interface:

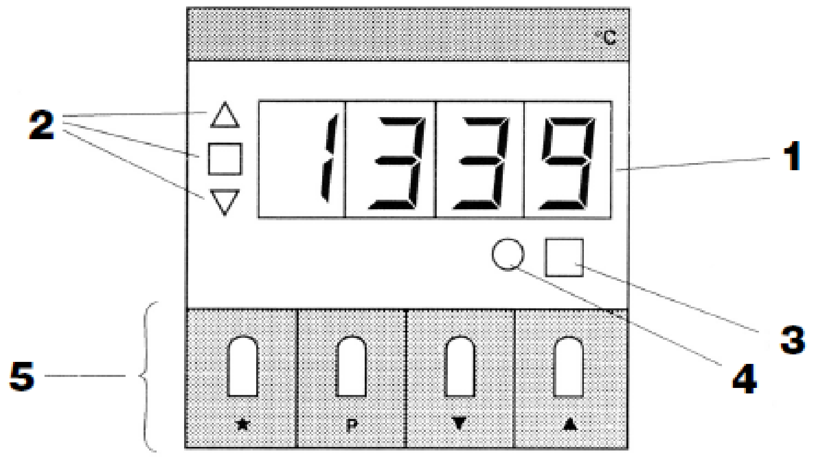


FIGURE 12: FRONT VIEW OF CN9000A PID CONTROLLER (OMEGA, N.D.A)

Where each numbered feature is explained below (Omega, n.d.a):

1. LED Display – a four-digit display that usually shows the process temperature of the system. When in set-up mode, it shows Function number on the right and Option numbers on the left, relative to the floating decimal point.
2. Deviation Indicators – these three lights show the difference between the set point temperature and the process temperature, as detailed below:
3. Error Indicators, when:

3.1. Flashing (up-triangle) – 3% above set point value

3.2. Steady (up-triangle) – 1% to 3% above set point value

3.3. Steady (square) – ±1% about set point

3.4. Steady (down-triangle) – 1% to 3% below set point value

3.5. Flashing (down-triangle) – 3% below set point value

4. Set Point Indicator

4.1. Set point 1 Indicator – the light (green) is displayed when the SP1 output is on.

4.2. Set point 2 Indicator – the light (red) is displayed when the SP2 output is on.

5. Control Keys – allow for the display of set values and change of control parameters:

5.1. (star) – displays the SP1 temperature

5.2. (star)(up-arrow) – pressed simultaneously will allow the increase of the set point.

5.3. (star)(down-arrow) – pressed simultaneously will allow the decrease of the set point.

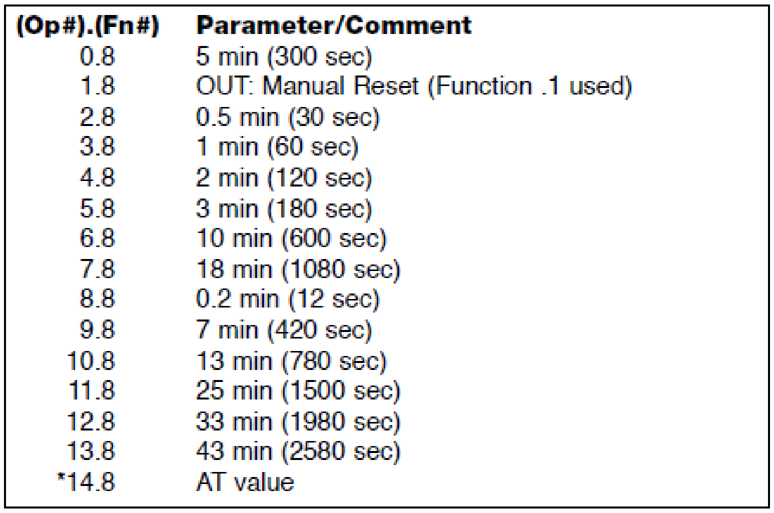
5.4. P – allows entry intro set-up mode, where Function and Option commands can be changed at user's criteria.

5.5. (up-arrow)(down-arrow) – allows change of Function and Option numbers when in set-up mode.

5.6. (star) – When in set-up mode, allows the change between sub-modes, from Function to Options, and vice-versa.

In the gas flow apparatus experiment a set point temperature value, described above as set point 1 (SP1), has already been chosen at 70° C. There is no second set point (SP2). The deviation indicators are only useful for steady state error, as different indicators will flash quickly and will not be useful during the rise time or settling time of the temperature of the system. A thermocouple is located close to the heater outlet in the piping configuration, and this is used to monitor the real-time temperature (the process variable) response curve instead of the built-in indicators.

Any analysis by the students in the lab must therefore be focused on changing the control parameters, described above as the Function and Option commands, via the control keys to optimize the process. The Function commands refer to specific parameters which will define how the PID controller operates, while the Option commands refer to specific sub-parameter within each Function command. An easier way to represent this Function and Option concept is through an example, shown below:



(Op#).(Fn#)	Parameter/Comment
0.8	5 min (300 sec)
1.8	OUT: Manual Reset (Function .1 used)
2.8	0.5 min (30 sec)
3.8	1 min (60 sec)
4.8	2 min (120 sec)
5.8	3 min (180 sec)
6.8	10 min (600 sec)
7.8	18 min (1080 sec)
8.8	0.2 min (12 sec)
9.8	7 min (420 sec)
10.8	13 min (780 sec)
11.8	25 min (1500 sec)
12.8	33 min (1980 sec)
13.8	43 min (2580 sec)
*14.8	AT value

FIGURE 13: AVAILABLE OPTIONS FOR THE INTEGRAL TIME FUNCTION (OMEGA, N.D.A)

The Option and Function commands are represented as (Op#).(Fn#), where the decimal separates the two numbers. Figure 13 displays the Integral Time function (Fn# = 8) and the several Options (Op# = 0 – 14), each of which will change the integral time function itself. As can be observed, the Option numbers are assigned a specific integral time value, which will affect how the set point is maintained.

The Omega cn9000A PID Controller has 50 adjustable Functions (.1 - .50), which are separated into two groups. Functions .1 to .25, are designated Standard Functions, which are accessible to the user and are often manipulated to change how the controller operates. Functions .26 to .50 are labeled as Advanced Functions, and are not designated to be changed in the controller's normal operation. It is recommended by the manufacturer that the Advanced Functions should only be changed by qualified personnel (Omega, n.d. a). For the purposes of this research, only the Standard Functions will be investigated and changed in attempting to achieve a better, more stable, set point temperature value. Table 1 shows all the 25 Standard Functions included in the PID controller.

TABLE 1: AVAILABLE FUNCTIONS IN PID CONTROLLER (OMEGA, N.D.A)

Function #	Function
.0	Operating Modes
.1	Manual Reset
.2	Set point 2 Adjust
.3	Set point 1 Lock
.4	Cycle-time/On-off (Output 1)
.5	Proportional Band/Deadband (Output 1)
.6	Derivative Time/Rate (Output 1)
.7	Derivative Approach Control (Output 1)
.8	Integral Time (Output 1)
.9	Sensor Offset (Calibration)
.10	Cycle-time/On-off (Output 2)
.11	Proportional Band/Deadband (Output 2)
.12	Loop Break Alarm Time
.13	Activate Advanced Functions (.26 to .50)
.14	Not Used
.15	Resetting Functions (.0 to .24) to Default Settings
.16	Input Sensor Select & Range Table
.17	Negative Temperatures
.18	Display Resolution
.19	
.20	Sensor Break Protection (Output 1)
.21	Sensor Break Protection (Output 2)
.22	C/F Selection
.23	Software Version Number
.24	Upper Set point Limit (SPAN)
.25	Not Used

Not every function is relevant to this investigation. As examples, Function .9 is only used for calibration and Function .11 controls an unused second output. This research will only focus on the functions that are seen as especially significant in stabilizing the set point temperature value for the heater in the gas flow apparatus.

The main part of this investigation is focused on researching how changing certain Functions of the PID controller could affect the stability of the set point temperature. As seen in Table 1, there are a significant number of functions that can be varied that change how the controller functions. After analyzing the list carefully, it was determined that only five of these Functions would be investigated in this research, as they should have the greatest effect on the operation of the controller. This shortlist is shown in Table 2, with a much more thorough explanation of the functions and their effect on the quality of control of the system:

TABLE 2: RELEVANT FUNCTIONS FOR THIS INVESTIGATION

F#	Function	Comment
.4	Cycle time/On-Off	Also known as the *Proportional Time*, cycle time will determine the total length of time that the output will cycle on and off when the temperature is within the Proportional Band. This means that if the PID control calls for an output duty of 25% and the cycle-time is 4s, the output will be on for 1 second and off for 3 seconds. It can be deduced that the shorter the cycle time, the more precise the control. This is also known as a *PWM period*. See section PWM SECTION
.5	Proportional Band/Deadband (K_P)	The output from the controller is proportional to the error or change in measurement. This means that if a large error needs to be corrected, a large correction will be applied.
.6	Derivative Time/Rate (K_D)	The output from the controller is proportional to the rate of change of the measurement or error. This should inhibit more rapid change of the measurement than from the proportional band. This function is usually used to avoid overshoot.
.7	Derivative Approach Control	Used to eliminate the action of Derivative Time during warm-up of the system. During warm-up, the user is not looking to control any parameters, but is trying to get close to the set-point value.
.8	Integral Time (K_I)	The output from the controller is proportional to the amount of time the error is present. This function should eliminate steady state offset by outputting a signal that corrects repeating errors.

Changing these functions (by using the different preset options within each of these) will affect the way the PID controller responds to any offset. Table 3 shows how the three PID parameters from Equations 2.1-2.3 affect the system dynamics when their values are increased. The changes on system dynamics from altering these values affect the four major characteristics of a closed-loop system (Carnegie Mellon & University of Michigan, 1997).

TABLE 3: HOW PROPORTIONAL, INTEGRAL AND DERIVATIVE GAIN AFFECT THE CONTROLLER'S PERFORMANCE

Response	Rise Time	Overshoot	Settling Time	S-S Error
K_P	Decrease	Increase	No Trend	Decrease
K_I	Decrease	Increase	Increase	Eliminate
K_D	No Trend	Decrease	Decrease	No trend

Using these three functions, important changes can be made to the overall operation of the system. Logic dictates that a desirable system would be one that had a low rise time, reduced overshoot, reduced settling-time and no steady-state error. Auto-tuning and manual tuning are two methods of tuning that are available for the Omega cn9000A PID Controller. A simple tuning approach is one of trial-and-error, yet this method alone may not yield the most desirable PID response.

2.5.3 AUTO-TUNING

The Omega cn9000A PID Controller comes with an auto-tuning feature. This feature allows for the PID controller itself, through complex mathematical algorithms, to determine the optimum operating conditions for the controller based on the deviation of a process variable during a representative run. A logical starting point in investigating how to optimize the PID controller and its functions is to use the auto-tune feature as a first step. From previous testing it was determined that the auto-tune option does not work for this specific experiment; however the reason for this failure had not been thoroughly examined. This MQP will attempt to examine why the auto-tuning feature of the Omega cn9000A controller was inadequate, and will solve the problem if possible. Some of the anticipated errors expected during the investigation and their possible solutions are shown in Table 4.

TABLE 4: POTENTIAL ERRORS ENCOUNTERED IN AUTO-TUNING (OMEGA, N.D.A)

Error code	Problem	Potential solution
EE5	Outside time limit	Make adjustments to Function number .4
EE6	Overshoot exceeds limit	Adjust derivative gain
EE7	Unable to run Auto-tune, SP1 in ON/OFF mode	Make adjustments at Functions number .20 and .21

A more thorough explanation on how to correct any of the errors displayed in Table 4 is detailed in the user manual. However, it is entirely possible that other errors (and not only the ones displayed above) could be found in the system. For a detailed method of troubleshooting the auto-tuning function, refer to the methodology.

2.5.4 *MANUAL TUNING*

The "Simplified Method" found in the PID controller's user manual is a method of manual tuning using a logical, step-by-step approach. The simplified method uses the default factory settings of the PID controller for the initial run. The intention is for the user to gather data of this run by taking readings at regular intervals in order to create a graph of how the process variable changes over time. Depending on how well the pre-set settings of the controller fit the current system, the resulting graphs will show how the process variable differs from the desired system behavior. A sample graph is illustrated below:

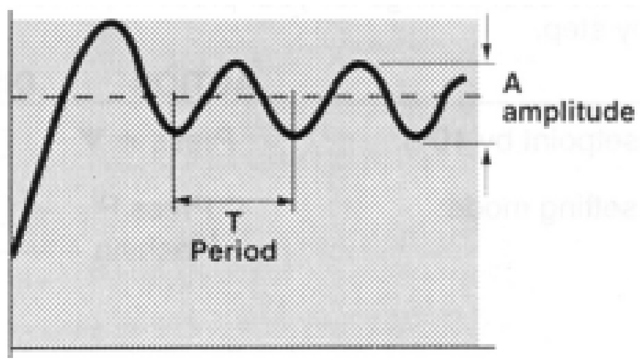


FIGURE 14: CHARACTERISTICS FROM DATA NEEDED TO OPTIMIZE PROCESS (OMEGA, N.D.A)

As shown in Figure 14, once a graph is plotted, information such as the response period (T) of a complete cycle and the amplitude of the oscillation is extracted from the graph. Parameter values can be calculated using a set of equations (shown below) once the period and amplitude of a cycle are recorded.

Operator Adjustments (Omega, n.d.a):

1. Proportional Time = T/20 *[If 10 seconds or less, use pulse output model]*
2. Proportional Band % = (A*1.5*100)/(Full Scale) *[Set to next larger % setting]*
3. Derivative (Rate) Time = T/20 *[Set to next shorter setting]*
4. Integral (Reset) Time = T

The resulting values from these equations are input into the controller instead of the initial default PID gains. The system is run again and re-evaluated under these new conditions. This method is usually good for estimating values for controlling a system and often yields acceptable results. These values can be further adjusted to give optimum performance.

3 METHODOLOGY

3.1 EQUIPMENT

The gas flow apparatus consists of a series of interconnecting ½ inch schedule 40 piping. Several instruments are installed between the inlet and outlet of the apparatus. The only pieces of equipment that will be listed in this section are those devices that are vital in the implementation of process control for this project. For a schematic refer to Appendix 6.3.

The heater used to heat the gas flow is a 400-watt model AHP-5051 “T” Type Air Process Heater from Omega Engineering, Inc (Stamford, CT). The original PID controller is a CN9000A from Omega Engineering. All temperature data is provided by 10 J-Type (Iron-Constantine) Thermocouples, which are read through the model NI USB-9213 Data Acquisition Device from National Instruments (Austin, TX). The data from the USB-9213 is transmitted into a computer using the LabVIEW 2010 software with the optional Control Design and Simulation Module installed. To turn the heater on, the LabVIEW VI transmits a 5-Volt signal via a model NI USB-6501 Low-Cost USB Digital I/O Device from National Instruments. This signal is sent through a model SSR330DC10 solid state relay (Omega Engineering) to Variable Autotransformer (Variac) Controller. For the purposes of this project, the Variac controller was always set to the maximum setting to supply 120 volts.

The vortex tube used in this study is a Model 20025 Stainless Steel Vortex Tube from AiRTX International (Cincinnati, OH). The vortex tube was installed into the gas flow apparatus as a bypass component, and does not receive airflow from the apparatus until all the necessary ball valves are opened. For the initial water-bath experiments, a model EMH-060-120V Screw Plug Immersion Heater was purchased from Omega Engineering. A model NI USB-TC01 Thermocouple Measurement Device from National Instruments was used to read the temperature of the water bath experiments and transmitted it to the computer for the LabVIEW PID to read. This model of thermocouple reader was also used for the gas flow apparatus while the NI USB-9213 was on back-order.

3.2 Omega Tuning

The Omega PID had already been tuned by WPI faculty before the project began. It was necessary to validate that the PID was tuned to the best possible settings if it was to continue being used for the gas flow apparatus experiments. In addition the LabVIEW PID would likely use similar settings, and it was possible that changing the most important functions would result in a better understanding of the necessary PID parameters.

3.2.1 Manual Tuning

The only functions changed were functions 5, 6, and 8. These functions correspond to the 3 gain terms from each PID parameter. The faculty tuning settings are shown in Table 5.

TABLE 5: TUNED SETTINGS FOR THE OMEGA PID.

Operation #	Function #	Function	Operation Value
13	.5	Proportional Band/Deadband	14%
7	.6	Derivative Time/Rate	1 second
8	.8	Integral Time	12 seconds

Function 5 (Proportional Band) was at a setting where both higher and lower values were possible. Three new options were tested, one which had the next possible higher percent proportional band, and two which had the next possible lower percent proportional bands. In other words, there were no other possible predefined options within the range of values that were studied. Function 6 (Derivative Time) was already set to the lowest value. For the Derivative Time function, the next two highest option values were studied. It was determined from the results of the second test that no additional values were needed for the derivative time function. For Integral Time, the next three highest settings were selected for testing. The exact values used for every test are shown in Table 6.

Only one function was changed at a time in order to isolate the effect of each PID variable. Whenever one function was changed the other two were set to the previous faculty tuned values, as shown in Table 5.

TABLE 6: THE SETTINGS FOR EACH FUNCTION THAT WERE STUDIED. NO ADDITIONAL OPTIONS REMAINED IN THE RANGE OF VALUES STUDIED.

Function #	Setting 1	Setting 2	Setting 3
.5	7 (20%)	1 (0.5%)	11 (7%)
.6	2 (30 seconds)	3 (1 minute)	0 (5 minutes)
.8	11 (15 seconds)	10 (7 seconds)	N/A

The results from each run were graphed, using an NI-USB TC-01 thermocouple reader to gather the temperature data. The results are described qualitatively, as it was obvious the data trends did not require any statistical analysis to determine which tuning parameters were superior.

3.2.2 Auto-Tuning

It was understood that Auto-tuning should be plausible for the gas flow apparatus experiment, as it is not of a highly complex nature and does not use custom settings on most of the provided functions. The following steps were taken during attempts to auto-tune the PID:

1. The gas flow apparatus was turned on.

2. The P button on the controller was pressed to enter set-up mode. The numbers "0.0" (Option 0, Function 0) were displayed on the screen. The Auto-tune routine is located in Function "0", which is the Operating Modes function.

3. The "⋆" (star) button was pressed to switch from the Function menu to the Option menu.

4. The default Option number was changed from the Option "0" to Option "1" to turn on the Auto-tune program.

5. The P button was again pressed to start the Auto-tune process. The heater was turned on simultaneously when the button was pressed.

At this point, the display alternated between the words AT and TEMP, which indicated that the Auto-tune program was running. Any error messages were recorded. The temperature results from each run were graphed using the NI-USB TC01 thermocouple reader.

3.3 LabVIEW Methods

3.3.1 LabVIEW Overview

LabVIEW™ 2010 was used with the optional PID and control modules installed. Using these modules, it was possible to construct a Virtual Instrument (VI) that would simulate the gas flow unit operation. After the VI for the simulated experiment was complete, another VI was programmed to control the gas flow apparatus's heater. The main component to each VI design was the PID sub-VI, which read the data and controlled the temperature.

Any variable categorized as "user-defined" in the following paragraphs can be manipulated via the front panel.

3.3.2 LabVIEW Virtual Experiment

Assumptions

The virtual VI had to use a representative data set in order to be accurate. For this data set, the inlet volumetric flow was assumed to be 6.6 SCFM at 20°C, which was the operating condition used for all data in this experiment. This should be similar or identical to the operating conditions used by students during the gas flow apparatus laboratory. The following assumptions were made while setting up each calculation:

1. The properties of air are constant.
2. The inlet flow temperature is constant.
3. The set points and initial conditions of operation will not be changed.

These assumptions made it possible to construct a simple VI that still properly demonstrates both the gas flow apparatus conditions and the basics of PID control.

A while loop was essential to the simulated VI, as this loop can operate continuously. Continuous operation of a VI allows the user to view the development of a process parameter over time. The difficulty with modeling the gas flow apparatus with a while loop is that the loop can only make calculations based on each iteration. This leads to a few different possible approaches when determining how the while loop will calculate the process temperature:

1. Start with a constant initial temperature for each run,
2. Use the temperature from the previous iteration, or
3. Use a combination of constant initial temperature and the temperature from the previous iteration.

Approach #1 allows for accurate readings, but the virtual experiment behaves unrealistically. Because the initial temperature is a constant value, the PID module controls the process variable perfectly. There is no settling time at all, and the temperature is steady and constant as soon as it rises to the set point. Approach #2 allows for realistic control, but the power output of the heater is incorrectly shown as being low – around 1-3 watts. It was determined that students needed an accurate gauge of the heater's power output during the virtual experiment in order to see if PID was controlling the heater properly. Oftentimes certain PID settings can cause the output signal to fluctuate between high and low power, which can lead to wear on the equipment, and it is this situation that students should be wary of.

Approach #3 was used to construct the simulated VI, as it allowed for the benefits of approaches 1 and 2 without either of the major errors. However, there was some difficulty in programming this loop, as it required more variables and was more complex than other options.

The following sections are extremely technical. Please refer to the LabVIEW Appendix (Appendix 6.4) for an image of the entire block diagram if the following sections become difficult to understand.

Variables

The Virtual Instrument designed for the virtual experiment of this MQP was originally constructed using a simulated signal-based approach. This approach proved to be very difficult, and was shortly abandoned in favor of a purely mathematical VI. In this approach three variables (*Initial Temperature*, *Setpoint*, and *Output* (Q)) were defined as double integers.

Initial Temperature is user defined in degrees Celsius and was manipulated by the VI to produce the *Temperature* (which is the current temperature at the completion of an iteration) double integer. This is described in detail under the "Passive Heating/Cooling" section of this document. The PID sub-VI reads the Temperature double integer as the process variable input. *Setpoint* is also user-defined in degrees Celsius and is the set point the PID attempts to approach. The *Output Q* variable was read from the resulting PID signal. Here the letter Q in the variable name is the heat transfer, in Watts, that the PID produces. A block diagram of this process is shown in Figure 15:

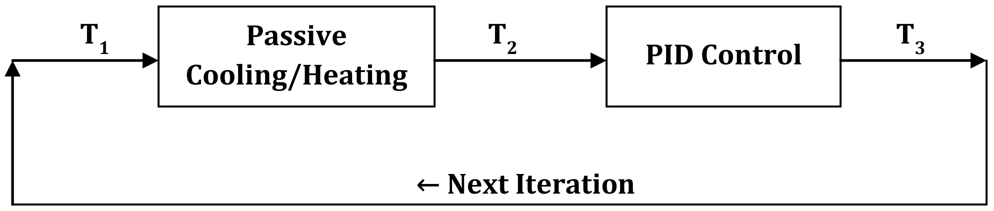


FIGURE 15: BLOCK DIAGRAM FOR SIMULATED VI WIRING. THE INITIAL TEMPERATURE (T_1) VALUE CHANGES AFTER THE PASSIVE COOLING & HEATING LOOP TO T_2. THIS TEMPERATURE IS CONTROLLED BY THE PID AND IS CHANGED TO T_3. THE VALUE OF T_3 IS USED AS THE VALUE OF T_1 FOR THE NEXT ITERATION.

The entire program was enclosed in a while loop, with a few exceptions. This loop was given a *stop* control, which placed a button on the front panel. This *stop* button allowed the user to end the simulation by halting the while loop. The number of iterations of the while loop were recorded and displayed as *Data Points* on the front panel. A time delay of 200 milliseconds between iterations was created, which corresponds to a sampling rate of five data points per second.

The *Temperature*, *Setpoint*, and *Output Q* variables were all bundled together into a cluster. From this cluster, a graph was created on the front panel. Numerical indicators of the values of each variable at the current iteration were also included on the front panel. All three variables were programmed to be graphed simultaneously, which allowed the user to view the quality of the PID control on the simulated airflow. The property node *History Data* was created for the graph, placed outside the while loop, and given a constant input. This purpose of this property node is to reset the graph history every time a new simulation was run.

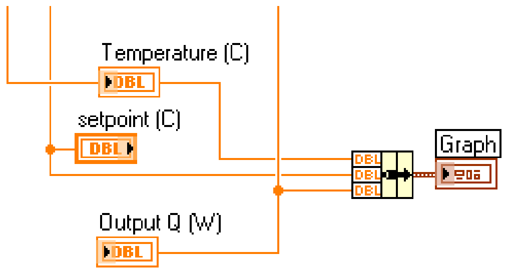


FIGURE 16: ALL THREE VARIABLES ENTER A CLUSTER AND ARE GRAPHED IN REAL-TIME.

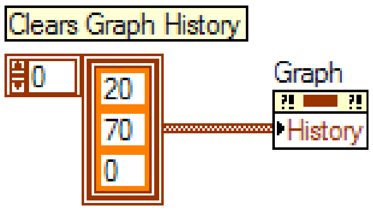


FIGURE 17: THE PROPERTY NODE CREATED TO CLEAR THE GRAPH'S HISTORICAL VALUES.

It is recommended that the students using this VI set the temperature set point to 70°C and the initial temperature to 20°C (room temperature), which are the values used during the unit operation experiment. This is because the history data property node is set to clear the graph's historical values for each variable to the values defined in Figure 17. If the values are set to anything else, it would result in an initial jump on the graph. This initial jump in temperature may confuse students.

Passive Heating/Cooling

One of the early concerns about the VI was the active heating that takes place when the heater is switched on without PID control and the passive cooling that takes place when the heater is off. For the purposes of the simulated experiment VI, it was assumed that the heater would not be on without PID control, and therefore active heating was not considered. Passive cooling to room temperature (20 degrees Celsius) was considered and an internal loop addressing the issue was programmed. With the addition of the vortex tube, it was shown that temperatures could drop below 20 degrees Celsius, and therefore passive heating could occur. The internal loop was designed with both conditions in mind.

A for loop was programmed inside the while loop, with the number of iterations of the for loop set to one per every execution of the while loop. A True/False statement was created, with the condition being true when the total number of iterations in the while loop is equal to zero, and false otherwise. When the condition is true, the for loop would execute using the *Initial Temperature* variable. When the condition is false, a passive heating and cooling equation described in the next few paragraphs would execute in the for loop.

A *local variable* of the *Initial Temperature* was created and programmed outside both the for loop and while loop. A local variable is a LabVIEW function which allows the program to either read or write a double integer value to a variable. This particular local variable was set to read the value from *Initial Temperature* and write the value to the *Temperature* variable, so that the initial temperature would be used for the first set of calculations during startup (T_1, refer to Figure 15). The calculations then used the value after each iteration (T_3, refer to Figure 15) for the *Temperature* variable at every other point in time. This setup also allowed the *Temperature* variable to read the *Initial Temperature* and display it on the graph at zero time.

The local variable was first wired via a *shift register* into the while loop. A shift register allows the variable entering a loop to be affected by the function of the loop. The alternative to shift register is *tunneling,* where the variable travels through the loop independent of the loop's functions. The *Temperature* variable was then wired via another shift register into the for loop, which contained the calculations for passive cooling and heating.

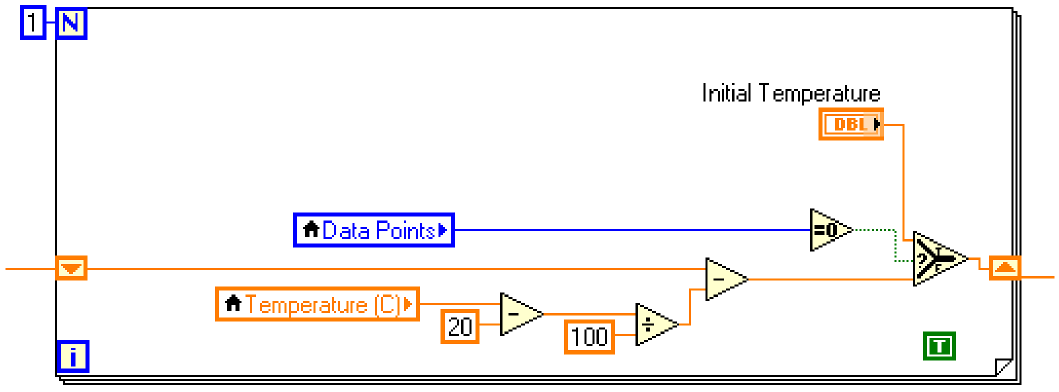


FIGURE 18: THE INTERNAL FOR LOOP THAT CONTROLLED PASSIVE HEATING AND COOLING.

Once the data from the "Temperature" variable was output into the for loop, it was subjected to the Equation 3.1:

$$T_2 = \frac{T_1 - 20}{100} \qquad 3.1$$

where T_1 is the entering temperature and T_2 is the temperature after passive heating and cooling. The equation takes any difference between the current temperature and 20 degrees Celsius and divides this difference by 100. This allows for a slow approach of the current temperature to 20 degrees when the heater and PID are deactivated. The number in the divisor can be increased for a slower approach or decreased for a more rapid approach to 20 degrees. A value of 100 was chosen arbitrarily because it clearly displayed passive cooling to the user when the PID was deactivated without taking a long time or converging too quickly. The result (T_2) was subtracted from the temperature entering the for loop, and the final value exited the for loop via a shift register and was wired as the PID input.

PID Control

In reality, the PID controls the output of the heater connected to the airflow experiment. It is therefore necessary to attempt to simulate this behavior virtually as well. For this reason, Equation 3.2 was used:

$$q_1 = \dot{m} * c_p * dT \qquad 3.2$$

where dT is the temperature difference between the entrance and exit of the heater (T_i and T_3, respectively). The value of the heat capacity of air (c_p) is 1007 J/(kg*K) (Cengel, 2010). The heat capacity of air is constant to four significant figures from 15 to 70 degrees Celsius, and can be assumed to be constant in the airflow experiment. The value of m was found via Equation 3.3:

$$\dot{m} = Q * \rho \qquad 3.3$$

where Q is the volumetric flow rate of air (in m^3/s) and ρ is the density of air (in kg/m^3) at 70 degrees Celsius. The volumetric flow rate of air was taken from a representative run during experimentation. The results of Equations 3.2 and 3.3 are shown in Table 5.

Since the inlet temperature is constant at 20°C, and the set point is constant at 70°C, Equation 3.2 applies for every second of the heater's operation. This requires the heat transfer, q_1, to be independent of the while loop's iterations. For this reason it is helpful to define q_1 in terms of a heat transfer coefficient. Equation 3.4 was derived from Newton's Law of Cooling:

$$h = \frac{\dot{m} * c_p}{A_s} \qquad 3.4$$

where A_s is the internal surface area of the gas flow apparatus' heater. The results for Equation 3.4 are shown in Table 5. The derivation of Equation 3.4 can be found in Appendix 6.4. The temperature (T_2) after each iteration of the while loop is given by Equation 3.5:

$$q_2 = h * A_s * d\acute{T} \qquad 3.5$$

where q is the heat transfer into the flow, A_s is the internal surface area of the heater, h is the heat transfer coefficient, and dT' is the temperature difference ($T_3 - T_2$) between the initial and final temperature of each iteration. All variables except for T_3 were either known or calculated. The equation was then put in terms of the temperature after PID control;

$$T_3 = \frac{q_2}{A_s * h} + T_2 \qquad 3.6$$

The output range of the PID was bound between 400 and 0, which corresponds to the power output, in watts, by the heater. The temperature after passive heating and cooling, T_2, was wired as the PID input. The internal surface area of the pipe was multiplied by the heat transfer coefficient of the flow. The heat transfer q was divided by the resulting value, and the temperature before PID control was added, resulting in the value for T_2. All of these manipulations can be seen in Figure 19.

The application of Equation 3.6 into the VI resulted in stable PID control with easily tuned parameters and a realistic temperature response curve. Unfortunately, the output power of the PID was only 3 watts. This was unrealistically low, as the PID output was shown to be around 170 watts experimentally. The flaw with Equation 3.6 was that it did not include the required heat transfer per second; it only showed the required heat transfer per iteration. In order to model the PID power output properly, the heat transfer from Equation 3.2 had to be included. This produced Equation 3.7:

$$q_2 = h * A_s * d\acute{T} + q_1 \qquad 3.7$$

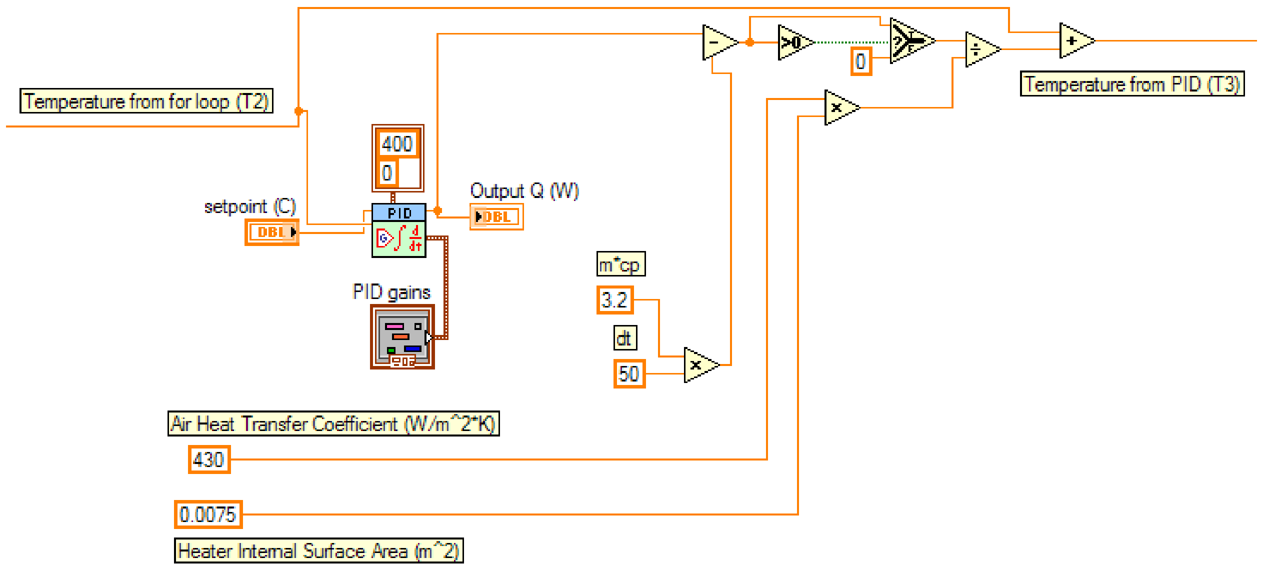
or, in its extended form;

$$q_2 = h * A_s * d\acute{T} + \dot{m} * c_p * dT$$

and Equation 3.6 was rearranged to become Equation 3.8:

$$T_3 = \frac{q_2 - q_1}{A_s * h} + T_2 \qquad 3.8$$

The result for T_3 was then connected to the while loop via a shift register, and was displayed on the graph located on the front panel. At the next iteration, all calculations would occur again. This process is shown graphically in Figure 19:



Temperature from for loop (T2)
setpoint (C)
DBL
400
0
PID
Output Q (W)
DBL
PID gains
m*cp
3.2
dt
50
0
Temperature from PID (T3)
Air Heat Transfer Coefficient (W/m^2*K)
430
0.0075
Heater Internal Surface Area (m^2)

FIGURE 19: THE PID SUB-VI AND RELATED EQUATIONS AS THEY APPEAR IN LABVIEW. THE PID GAINS WERE SET BY THE USER. SIGNALS TRAVEL FROM LEFT TO RIGHT.

The application of Equation 3.8 does result in a few unfortunate consequences, as q_1 is constant when the total temperature difference ($dT=T_3-T_i$) is defined. Firstly, the set-point temperature cannot be very different from 70. Otherwise, the wattage output by the heater will not be accurate. This issue was originally addressed by having LabVIEW read the values of the initial temperature (T_i) and the current temperature (T_3), instead of defining dT as 40°C. However, this resulted in some unusual bugs in the VI, and was abandoned in favor of a constant value of q_1.

Secondly, the temperature response curve shows no change until q_2 becomes larger than q_1. This is inaccurate, as even a minuscule wattage will result in some appreciable temperature difference.

Luckily, this delay is usually only a second or so long and is not considered to be significantly disruptive to the laboratory experience.

The results from Equations 3.1-3.5 are shown in Table. The physical properties of air used to calculate some of the results in Table 7 are listed in Table 8. Full calculations are discussed in Appendix 6.4. A full-page screenshot of the entire VI is also in Appendix 6.4 for further reference.

TABLE 7: CALCULATIONS FOR VARIOUS PARAMETERS REQUIRED FOR EQUATIONS 3.1 TO 3.8.

Inside Diameter (in)	0.62
Inside Diameter (m)	0.016
Heated Length (in)	4.5
Heated Length (m)	0.1143
Internal Heater Surface Area (m^2)	0.11
Cross-Sectional Area (m^2)	0.0019
Representative Volumetric Flow (SCFM)	6.6
Representative Volumetric Flow (m^3/s)	0.0031
Representative Velocity (m/s)	16
Mass Flow Rate (kg/s)	0.0032
Heat Transfer (Watts)	160
Heat Transfer Coefficient (W/m^2*s)	29

TABLE 8: PHYSICAL PROPERTIES OF THE AIR FLOW AT 70^{o}C (CENGEL, 2010)

Viscosity (m^2/s)	0.00002052
Density (kg/m^3)	1.028
Specific Heat (J/kg*K)	1007
Prandtl Number	0.7177
Reynolds Number	123000

Summary

The behavior of the VI is as follows:

1. The *Initial Temperature* (T_1), *Setpoint*, and PID gains are set by the user.
2. When the experiment is run, the *Initial Temperature* is read outside the while loop, and is sent through a shift register and inside the while loop.
3. The *Temperature* is recorded before entering the for loop.
4. In the for loop, the exiting temperature (T_2) is calculated by Equation 3.1
5. The first true/false statement is applied. If it is the first iteration, the *Initial Temperature* (T_i) is chosen. If it is not the first iteration, the temperature calculated in step 4 (T_2) is chosen. The result exits the for loop.
6. The temperature (T_2) exiting the for loop is used as the PID input, while also continuing on to solve the controlled temperature (T_3) in Equation 3.8.
7. The PID responds according to the user-defined gains, and tries to approach the set point. The PID *Output* (q_2) is recorded.
8. The current temperature, S*etpoint*, and PID output (q_2) are graphed.
9. The static heat transfer (q_1) is calculated. If the difference between the static heat transfer and the PID output is negative, then a value of 0 for q_1 is used in Equation 3.8 instead.
10. Equation 3.8 is solved, yielding the controlled temperature (T_3).
11. 200 milliseconds later, the value of T_3 from the while loop's exit shift register is sent to the entrance shift register, and the calculations are repeated.
12. The experiment ends when the loop is stopped or the VI is aborted.

3.3.3 LabVIEW Physical Experiment

In order to develop a VI for the physical LabVIEW PID, the Data Acquisition Devices (DAQs) needed to be physically wired to the appropriate hardware. The first DAQ used was the NI USB-6501. This DAQ is a digital input and output (Digital I/O) device capable of sending a digital signal of 5 volts. For this project, the NI USB-6501 sent an output signal to a solid state relay. A solid state relay is a type of electrical switching device. It requires a device sending a smaller signal to activate the switch, completing a circuit for a larger load. The signal sent to the solid state relay is 5 volts when the DAQ was operating at the high setting. 3 volts are sent to the solid state relay when the DAQ was operating at the low setting. The solid state relay needs 4 volts to activate, so the high setting is essentially the "on" setting and the low setting is essentially an "off" setting in the case of this experiment. When the solid state relay is active, the heater would be turned on. When the solid state relay is inactive, the heater does not receive power from the Variac controller and is off.

The second DAQ installed was the NI USB-9213. This DAQ is purely an input device, and is used to read the ten thermocouples attached to the gas flow apparatus. The thermocouples were already physically wired to an analog thermocouple reader that displayed one temperature at a time. The wires were moved from the analog reader to the NI USB-9213.

Next, the VI was developed. For a picture of the complete VI, please refer to Appendix 6.4. First, the entire VI is encased in while loop. *DAQ Assistants* were created for NI USB-6501 and NI USB-9213. A DAQ Assistant is a programming function in LabVIEW that processes data from a DAQ. The NI USB-9213 DAQ Assistant had channels one through 10 selected, where each channel is a signal pathway from an individual thermocouple. Each channel had an indicator created, so that the front panel would display all the temperatures read by the thermocouples. Channel three, the reading from thermocouple 3, was used as the current process variable input in the PID sub-VI. Also, a graph of the temperature recorded by thermocouple 3 was created. This allowed the user to watch the temperature change as a function of time and also to see the approach, settling time, and steady state of the PID control. Thermocouple 3 was the PID input used for the Omega PID as well (Figure 20).

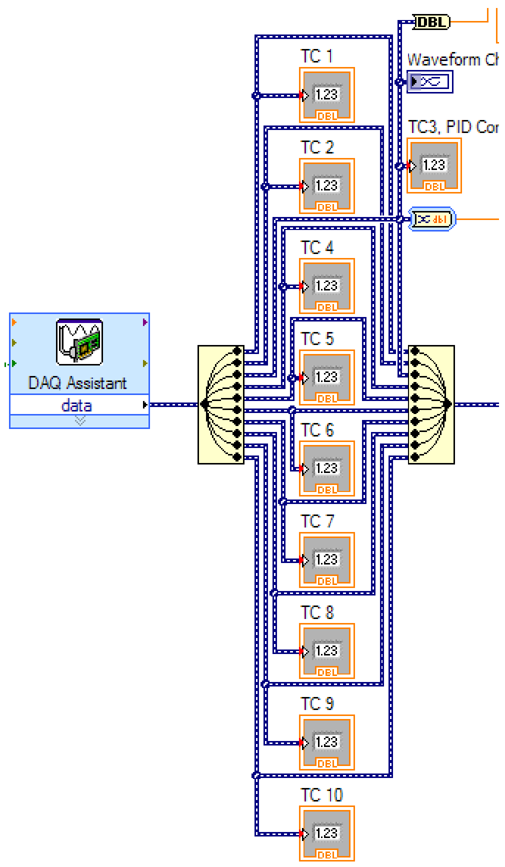


FIGURE 20 NI USB-9213 DAQ ASSISTANT AND INDICATORS

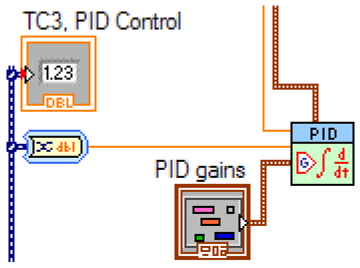


FIGURE 20A: THERMOCOUPLE 3 SENDING SIGNAL TO PID

Since the NI USB-6501 sent a digital signal, pulse-width modulation (PWM) had to be programmed into the VI in order to simulate an analog signal and achieve actual PID control. The sequence for the PWM began by receiving the signal from the PID sub-VI. After the PWM sequence occurred, it delivered a *Boolean* array response of 0 or 1 to the DAQ assistant for the NI USB-6501. A value of 0 would set the DAQ to its low setting, deactivating the solid state relay and turning off the heater. A value of 1 would set the DAQ to its high setting, activating the solid state relay and turning on the heater (Figure 21).

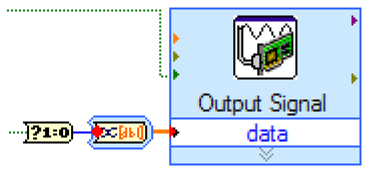


FIGURE 21: NI USB-6501 DAQ ASSISTANT WITH BOOLEAN INPUT

In order to program PWM, the PID signal was given a range of 0 through 100, which is the duty cycle. A for loop was created and programmed to operate throughout the PWM period of 100 milliseconds. The for loop was set to run an iteration every 10 milliseconds, and ran ten iterations per PWM period. The duty cycle was divided by 100 to express it as a percentage, and then multiplied by 10. This was done because there are 10 iterations for every PWM period, and the percentage multiplied by 10 is the number of pulses administered throughout the PWM period.

The duty cycle percent value (multiplied by 10) is compared to the number of for loop iterations. If the duty cycle percent value is greater than or equal to the iteration number, then this creates a true response to the case structure. In this case, the case structure does not alter the value, and a 1 response would be sent to the Boolean array. As previously mentioned this triggers a high output from the DAQ and ultimately activates the solid state relay and powers the heater. If the duty cycle percentage (multiplied by 10) is less than the iteration number, then this creates a false response in the case structure. In this case, a *not* node is triggered and a value of 0 is sent to the Boolean array, signaling a low output from the DAQ and leaving the solid state relay deactivated.

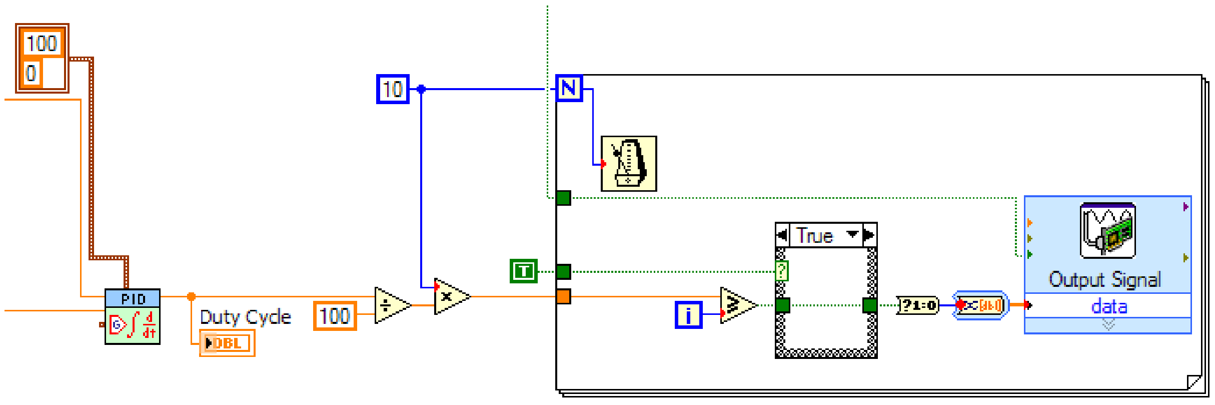


FIGURE 22: PWM PROGRAMMING. THE *TRUE* CASE IS SHOWN FOR THE CASE STRUCTURE.

As an example, the PID outputs a signal with a value of 50 out of a possible 100. This would create a 50% duty cycle, or a 0.5 duty cycle. This number would be multiplied by 10, so the duty cycle would supply a 5 to the for loop for the PWM period. For the first 5 iterations, the number the duty cycle supplied would be greater than or equal to the iteration number, so the output signal would be high. This would turn on the heater for the first 50 milliseconds of the PWM period. During the sixth through tenth iteration, the number provided by the duty cycle would no longer be larger than the iteration number, so the Boolean would command a low output, shutting off the heater for the remaining 50 milliseconds of the PWM period. After each PWM period, the for loop would restart and would receive a new value from the PID, restarting the process again with 10 new iterations.

It was noted that once the experiment stopped, the DAQ would sometimes still administer a high signal, and the heater would be continuously active despite the VI being shut off. This could potentially damage the equipment, as the heater would have no control while operating. In order to solve this issue, a condition was added to the VI. The condition was true when the while loop encasing the VI was stopped and false while the while loop and the VI was active. The true condition administered a set point of -100 ^{0}C to the PID, so that when the while loop was stopped, the PID would automatically shut off the heater due to the temperature being above the set point. When the condition is false, and the VI and while loop are active, the set point is user defined.

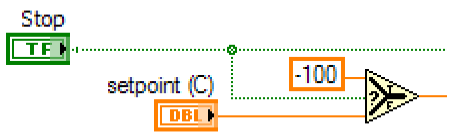


FIGURE 23: CONDITION TO SHUT OFF HEATER

Once all issues were resolved, a *write-to-file* block was added so that the data collected in each run could be exported to a Microsoft Excel file. The data from all thermocouples was bundled and sent to the *write-to-file* block. A control for the *write-to file* block was added so that when the indicator says “enabled” the data is being exported to a file, and while it says “disabled” the data is not being exported. This block was included so that students do not have to export data they feel they do not need to record. The data exported is displayed in Excel as 20 columns. The even numbered columns, titled “Untitled,” are the readings of the thermocouples, and the odd number columns, titled “X value,” are the times that the readings from each thermocouple were taken (in seconds). The time is displayed next to every temperature reading rather than in one column due to

LabVIEW's limited options for displaying data. Controls were added to the front panel so that students can choose which file path the data should be exported.

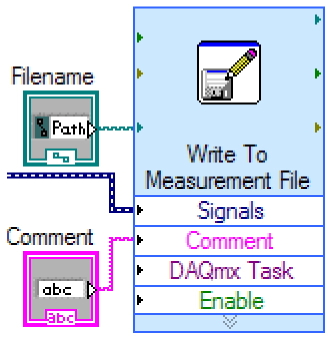
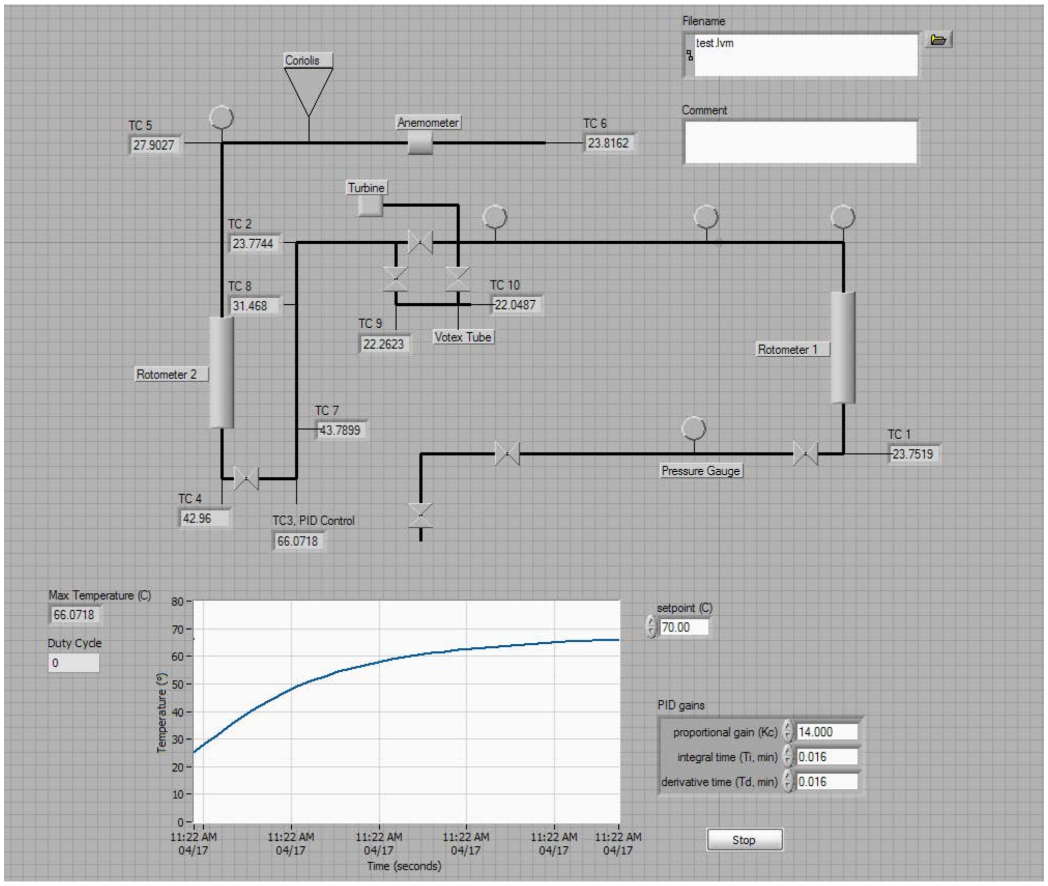


FIGURE 24: WRITE-TO-FILE BLOCK

Once the programming on the back panel was complete, a design of the gas flow apparatus was drawn on the front panel, and the thermocouple indicators were moved to illustrate where in the gas flow apparatus the reading was located. The controls were organized so that they were in the same area.



Filename test.lvm

Comment

TC 5 27.9027

TC 6 23.8162

TC 2 23.7744

TC 8 31.468

TC 10 22.0487

TC 9 22.2623

TC 7 43.7899

TC 1 23.7519

TC 4 42.96

TC3, PID Control 66.0718

Max Temperature (C) 66.0718

Duty Cycle 0

setpoint (C) 70.00

PID gains

proportional gain (Kc) 14.000

integral time (Ti, min) 0.016

derivative time (Td, min) 0.016

FIGURE 25: FRONT PANEL

3.3.4 EXPERIMENTATION WITH LABVIEW

In order to determine if the VI functioned as a proper PID controller, and to prevent possible damage to the equipment on the gas flow apparatus, a water bath experiment was performed. The water bath experiments consisted of an Erlenmeyer flask filled with water. A Screw-Plug Immersion Heater was fully inserted into the water. For the water bath experiment, the NI USB-6501 was wired to a different solid state relay than the one used with the gas flow apparatus. When activated, the solid state relay would complete a circuit that would send power to a heater in a water bath via the Variac. A thermocouple was placed in the water bath as well and connected to a TC01 thermocouple reader.

The NI USB-9213 was not used for the water bath experimentation, so the VI constructed for the water bath experiment required using a DAQ assistant for the TC01. The signal from the TC01 DAQ assistant replaced the signal from channel 3 of the NI USB-9213 as the current process variable input for the PID sub-VI. The rest of the VI is identical to the one described in the section "LabVIEW Physical Experiment."

Once the VI was constructed, the circuit was plugged into the Variac controller for power. When the VI was running, the activity of the solid state relay was observed. Since the specific heat of water is very high, the water bath would not cool down quickly enough to observe the basics of PID control (i.e. oscillations, settling time, etc.). The goal of the experiment was simply to have the solid state relay be active until the water bath temperature reached the set point. The solid state relay would then become inactive when the water bath temperature was above the set point. If the heater could switch off at all, it was an indicator that all PID signals were being properly sent. This is further discussed in results and discussion.

After it was determined that the VI functioned properly, the NI USB-6501 was physically wired into gas flow apparatus via the existing solid state relay. There was a switch installed before the solid state relay to allow users to switch between using the NI USB-6501 or the Omega PID to send a signal to the solid state relay. This was included so that it would be easy to compare the Omega PID to the LabVIEW PID. Also, the switch allows for a user to switch quickly if one PID should be broken, rather than having to rewire the system. Power to the in-line airflow heater was still supplied by the Variac.

3.4 VORTEX TUBE

The purpose of integrating the vortex tube into the airflow experiment is to have students gain a better understanding of the first law and second law of thermodynamics. This is because the vortex tube separates the incoming air into a hot stream and a cold stream, and therefore creates an interesting energy balance. Figure 26 below shows a diagram of the vortex tube, with its inlet and outlets:

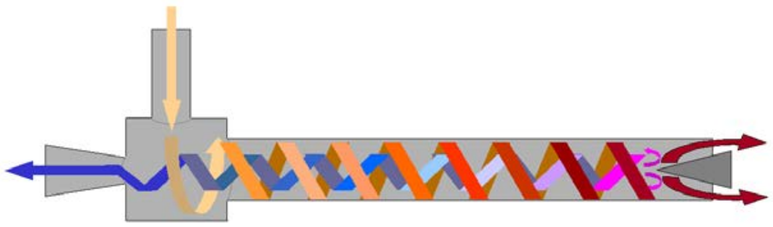


FIGURE 26: VORTEX TUBE FLOW DIAGRAM. IMAGE CREDIT: WWW.PDBUCHAN.COM

Before the vortex tube was physically used, an energy balance had to be derived. This was achieved by assuming the hot and cold outlets as completely separate streams throughout the entirety of the process, even though this does not accurately represent how the system works; this is shown in Figure 27. Working with these assumptions, the next step was to find the difference in enthalpy in each stream. In accordance with the first law of thermodynamics, if there is not heat loss these two enthalpies should be equal and opposite to one another. The derived energy balance is shown with Equations 3.9-3.13:

$$\dot{m}_i = \dot{m}_c + \dot{m}_h \tag{3.9}$$

$$0 = \Delta H_c + \Delta H_h \tag{3.10}$$

$$\Delta H = m \int c_p \, dT \tag{3.11}$$

$$0 = c_p[\dot{m}_c(T_c - T_i) + \dot{m}_h(T_h - T_i)] \tag{3.12}$$

$$0 = c_p[\dot{m}_c(T_c - T_i) + (\dot{m}_i - \dot{m}_c)(T_h - T_i)] \tag{3.13}$$

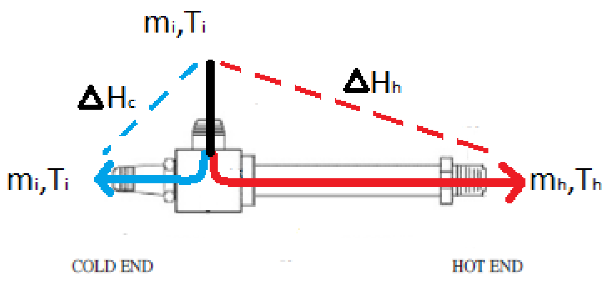


FIGURE 27: ASSUMED SPLITTING OF STREAMS

Having derived this energy balance, the gas flow apparatus was run using the vortex tube, to determine if Equation 3.13 was an accurate representation of the physical experiment. The measuring equipment on the gas flow apparatus can be used to find every unknown in the derived equation; however some numerical discrepancies arise when specific variables are used to solve the energy balance. Some of the discrepancies in the energy balance can be explained through heat loss. Even still, it was necessary to treat the system as isothermal process for the purpose of the classroom procedures.

To solve the problem, all possible information provided by airflow instruments was recorded, but the energy balance was performed with all but one of the values supplied by the instruments. Two values were chosen to be calculated through the energy balance exclusively. One such variable was cold mass flow, usually provided by the second rotometer. The recorded value was then compared to the calculated mass flow of air in the cold stream. The other variable calculated using an energy balance was the temperature reading from thermocouple 10 (located on the hot stream), which was then compared to the recorded value from thermocouple 10.

Initially the values were far too different. It was determined that a draft from a vent in the laboratory was blowing on thermocouple and causing an erroneous reading. In addition, the thermocouple was initially too far away from the vortex tube, resulting in an error. The thermocouple was adjusted to be much closer to the vortex tube and a draft shield was created using notebook paper to protect the thermocouple from the draft. These actions helped to lessen the difference between the calculated and actual values of the cold mass flow and hot temperature. More information on the differences can be found in the results and discussion section of the report.

In addition, after a prolonged amount of time, recorded temperatures showed that the system was drifting towards room temperature. To quantify this, values were taken every minute in an attempt to evaluate whether or not this affected the validity of the energy balance. It was determined that the drifting condition did not affect the energy balance, so long as all measurements were taken within one minute. More information on the drifting can be found in the results and discussion section of the report.

Since the energy balance was found to be within reason, classroom procedures that instruct students to derive and justify an energy balance around the vortex tube were designed. The mathematical analysis of entropy throughout the vortex tube was determined to be beyond the scope of the Unit Operations course, so it was decided to include the second law of thermodynamics in the classroom procedures qualitatively.

4 Results and Discussion

4.1 Manual Tuning Results

Before the LabVIEW PID was constructed and implemented into the airflow experiment, it was necessary to understand the Omega cn9000A PID. Even in the event that the LabVIEW PID was successful, the Omega PID would be maintained as a safe backup. As the backup, the Omega PID had to be set up to run conservatively, and must not trip the system alarms.

Four main goals were required of the Omega PID:

1. The settings must result in a quick rise time.
2. The temperature must not initially overshoot.
3. The settings must result in a rapid settling time.
4. The temperature must be maintained at the set point for the entirety of the experiment, with very little steady-state error.

The Omega cn9000A PID controller is programmed with 24 functions, and most functions come with a number of options. These are selected as two numbers on the Omega controller, separated by a decimal. To the user, this appears as the form "Option.Function." For example, option 3 on function 5 would be punched into the controller as "3.5."

All of the results discussed are not quantitative, but instead record general observations from monitoring the effect of changing each parameter.

Goals #1 and 4 were met by changing the proportional band. The proportional band (function 5) is the allowable span (i.e. distance) from the set point. If the current temperature is inside this defined range the proportional band is active according to the set percentage (Omega, 2011). Above this span the proportional band is off (0%) and below this span the proportional band is on (100%). The results of changing this function are shown in Table 9.

TABLE 9: RESULTS WHEN THE PROPORTIONAL BAND IS CHANGED.

Proportional Band Results	Setting	% Gain	Comparison to Initial
Initial	13.5	14%	N/A
Run 1	7.5	20%	Less stable@ set point
Run 2	1.5	0.5%	Oscillations
Run 3	11.5	7%	Oscillations

Goals #2 and 3 were achieved by changing the integral time function (function 8). Integral time is the length of historical data the PID "remembers." The function was found to already be set to the absolute minimum at 12 seconds. Changing this to a higher value consistently resulted in a longer rise time and a longer settling time. The results of changing this function are shown in Table 10.

TABLE 10: RESULTS WHEN THE INTEGRAL TIME IS CHANGED.

Integral Time Results	Setting	Parameter	Comparison to Initial
Initial	8.8	12 Sec.	N/A
Run 1	2.8	30 Sec.	Long Settling Time
Run 2	3.8	1 Min.	Longer Settling Time
Run 3	0.8	5 Min.	Longest Settling Time

The derivative band was the last parameter studied to see if it had any effect on goal #4. This parameter was also set to the minimum possible time. It was found that any increase in this function would result in integral destabilization. Results are summarized in Table 11.

TABLE 11: RESULTS WHEN THE DERIVATIVE BAND IS CHANGED.

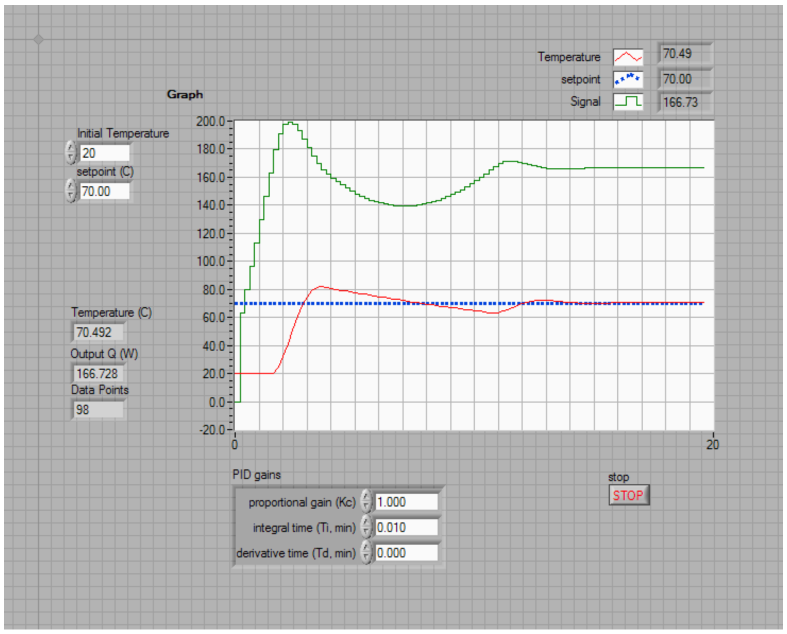
Derivative Band Results	Setting	Parameter	Comparison to Initial
Initial	7.6	1 Sec.	N/A
Run 1	11.6	15 Sec.	Large Integral Destabilization
Run 2	10.6	7 Sec.	Integral Destabilization

It was found that the initially selected parameters for Integral Time, Derivative Band, and Proportional Band were already optimized for the system. These values are the currently used parameters for the controller.

4.2 LabVIEW

4.2.1 *Virtual LabVIEW PID*

The final product of the virtual VI was a functioning digital replica of the existing gas flow experiment. For the complete front panel and back panel, please refer to Appendix 6.4. Following the completion of the VI, the system was set at the same set point and initial temperature that occur in the gas flow apparatus. The set point was 70°C, and the initial temperature was 20°C. With a functioning VI, the program was used to test different scenarios that could be presented to students during the laboratory.



Temperature 70.49
setpoint 70.00
Signal 166.73

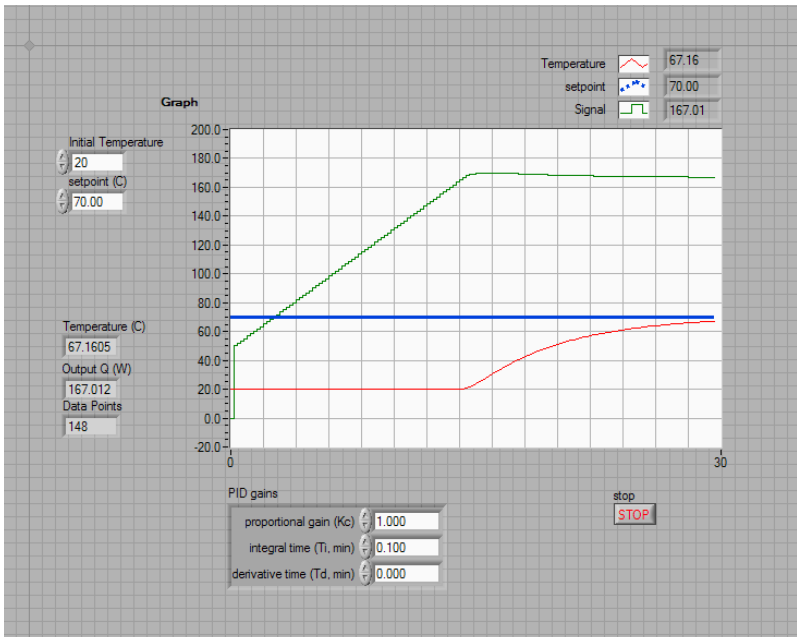
Initial Temperature 20
setpoint (C) 70.00

Temperature (C) 70.492
Output Q (W) 166.728
Data Points 98

PID gains
proportional gain (Kc) 1.000
integral time (Ti, min) 0.010
derivative time (Td, min) 0.000

FIGURE 28: THE VIRTUAL PID WITHOUT DERIVATIVE CONTROL

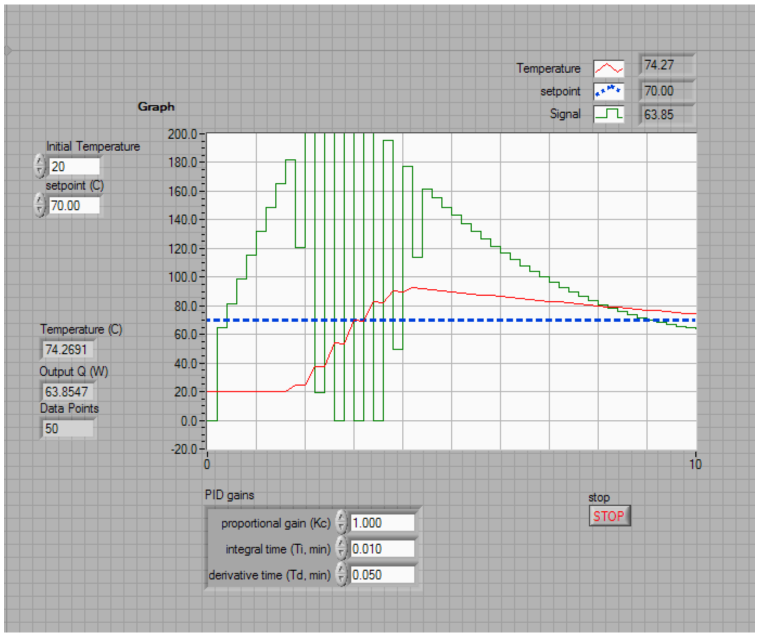
Figure 28 above demonstrates how a PI controller works, with a $K_c = 1.00$ and a $K_i = 0.010$ minutes. As seen, a reasonable approach was achieved and a steady state occurred in an acceptable time. The initial overshoot of the set point was an error that is usually unwanted; tuning of the integral time would fix this error.



Initial Temperature 20
setpoint (C) 70.00
Temperature (C) 67.1605
Output Q (W) 167.012
Data Points 148
Temperature 67.16
setpoint 70.00
Signal 167.01
proportional gain (Kc) 1.000
integral time (Ti, min) 0.100
derivative time (Td, min) 0.000

FIGURE 29: VIRTUAL PI WITH RE-TUNED INTEGRAL

In Figure 29 above, the integral time was changed to 0.1 minutes, whilst the proportional and derivative terms remained unchanged. By increasing the K_i term, the controller output a larger response to the perceived error. This resulted in a much more controlled approach to the set point with no overshoot; however the rise time was long relative to previous simulations. The rise time in the first PI simulation (Figure 28) was about four seconds while the rise time with the re-tuned integral term was about 30 seconds. The addition of an appropriate derivative time would reduce the magnitude of the overshoot in the system and introduce overall system stability.



Temperature 74.27
setpoint 70.00
Signal 63.85

Initial Temperature 20
setpoint (C) 70.00
Temperature (C) 74.2691
Output Q (W) 63.8547
Data Points 50

PID gains
proportional gain (Kc) 1.000
integral time (Ti, min) 0.010
derivative time (Td, min) 0.050

FIGURE 30: ADDITION OF A DERIVATIVE TERM

Figure 30 above displays a control graph with unchanged gains except for the derivative term. The derivative term is relatively large. This caused a much larger overshoot, reaching a temperature 90 ^{0}C, where the PI controller with the same proportional and integral term reached a max temperature of about 80 ^{0}C on the overshoot. The signal displayed in Figure 30 shows that the PID was rapidly reacting to error. This led to overcorrection and a higher overshoot. This was because the derivative term was too strong and influenced error correction on the rise time.

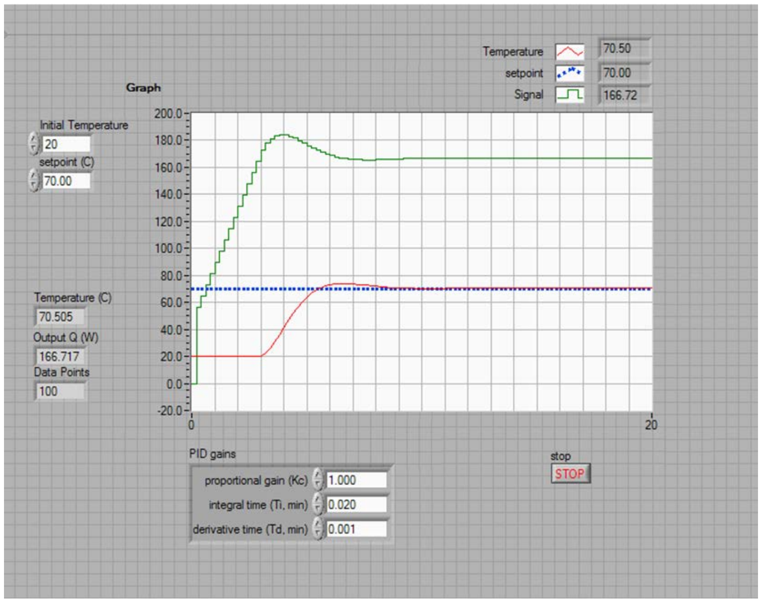
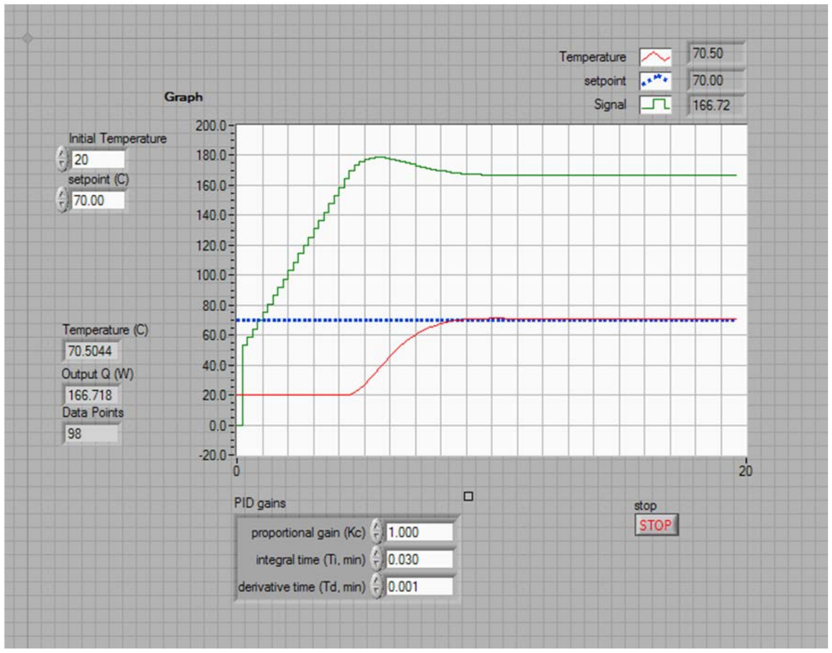


FIGURE 31: INCREASED INTEGRAL TERM AND DECREASED DERIVATIVE TERMS

Figure 31 displays the system being controlled with gains of K_c = 1.00, K_i = 0.020 minutes and K_d = 0.001 minutes. A much faster rise time and minimal overshoot of the set point was observed, and overall system stability was achieved. The increased integral term and decreased derivative term allowed for correction of error without overcorrecting or destabilizing the PID. Eliminating the overshoot and reaching a steady state were the goals for tuning the PID. In order to do this, the integral term had to be increased so the controller would output a larger signal to correct this error.



Temperature 70.50
setpoint 70.00
Signal 166.72

Initial Temperature 20
setpoint (C) 70.00

Temperature (C) 70.5044
Output Q (W) 166.718
Data Points 98

PID gains
proportional gain (Kc) 1.000
integral time (Ti, min) 0.030
derivative time (Td, min) 0.001

FIGURE 32: FINAL TOUCHES TO TUNING

Figure 32 displays the final tuned gains used for the virtual VI. As seen, a low rise-time, no overshoot, virtually no settling time, and negligible steady-state error are all achieved. This means all of the parameters mentioned in section 2.5.1 were optimized to produce a stable process. This demonstrates how the VI can be helpful for students in the pre-lab phase as they start getting a feel for tuning a PID controller. The students will be able to find how to tune the PID gains with respect to each other in order to develop a stable process.

With proper tuning, the virtual PID can properly control a process. On this basis, it was determined that this VI would be helpful as part of the pre-lab where students would be able to gain a feel of how LabVIEW works and how the PID gains can be tuned. More importantly, by using this program before the actual experiments, students can understand what is safe and what is unsafe in the actual laboratory experiment. For example, students will be able to play with the PID gains without any consequence of damaging equipment.

Furthermore, the front panel was created and organized in such a way that the students have a user-friendly experience, as well as an intuitive approach to the program. Also, the simultaneous plot of three data sets (temperature, set point, and signal) allows the user to easily see how the temperature relates to the set point and how the signal strength of the PID relates to the error and PID gains.

However, one minor error was detected in the final version of the virtual PID. It was discovered that whenever the initial temperature had a value different from 20°C, the graph would spike to zero and would then immediately settle at the prescribed initial temperature. Although it was detected early in the project stages and a correction was attempted several times, it was not possible to solve the issue because of the passive cooling equation that is essential to the process. Although this error is small and innocuous to the long-term controlling functions of the PID package in LabVIEW, it did represent a setback for two reasons. First, the virtual PID package would not be able to replicate the GFA experiment with 100% accuracy because of varying room temperatures. This is also relevant because it will not be able to replicate the process when using the vortex tube for proposed disturbances. Second, this deficiency in the virtual experiment could lead to confusion by students and also to misleading conclusions because they would not be able to comprehend or explain this phenomenon.

One last deficiency to the virtual PID is that the gains used were not the same as those used in the Omega PID or physical PID. This is because the virtual PID does not count for heat loss and other factors that could affect the system. Even though the gains will not be identical, this will not detract from the students' ability to learn the art of tuning a PID.

4.2.2 Physical LabVIEW PID

For an image of the front panel and back panel, please refer to Appendix 6.4.

The physical PID was very different than the virtual PID. The most noticeable change on the front panel was the GFA schematic, which is shown by itself in Figure 33 below:

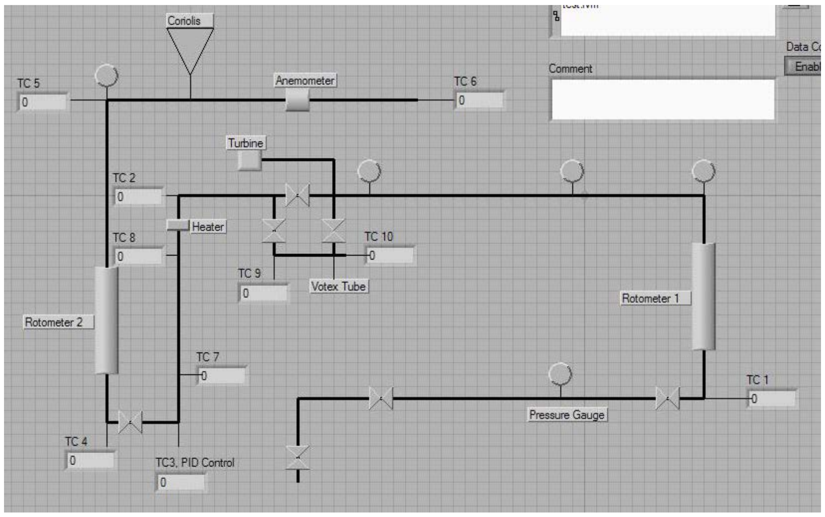


FIGURE 33: GFA SCHEMATIC

The schematic simplified the task of viewing and recording temperatures because the analog temperature display was replaced by the LabVIEW interface. In Figure 33, on the top right corner, a white rectangular box can be seen with a button on the side labeled "Data Collection". This feature allowed for the collection of temperature from all 10 thermocouples with the option of exporting the data using a program such as excel. This made it possible for students to not only display the real-time temperature on the screen, but also to log it so that they could graph it, analyze it, and use it in their report.

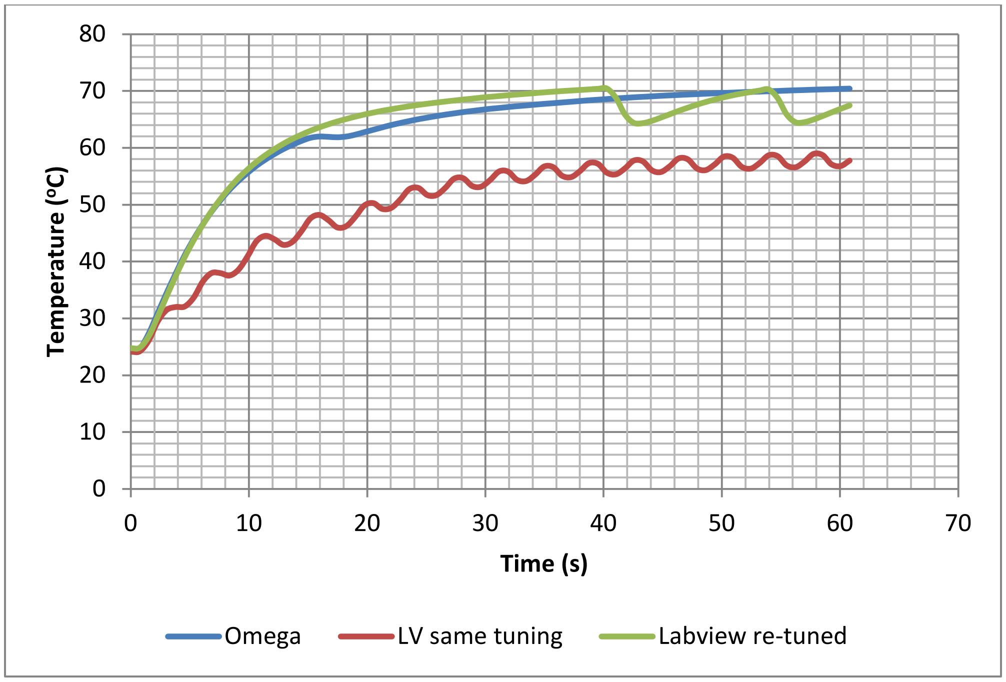


FIGURE 34: OMEGA PID VS. LABVIEW PID

Several tests were done to study the behavior of the physical PID in comparison to the Omega PID. As shown by Figure 34, when the same gains are used for both controllers, the physical PID shows the signal oscillating around 60°C. This is an appreciable steady-state error. The Omega PID still had very good control, implying that the LabVIEW physical PID tuning process would be slightly different. To correct this error, the physical PID integral term was retuned from 12 seconds to 2 seconds. As shown above in Figure 34, the rise time is similar to the Omega PID. There was still steady-state error. This error is not characteristic of steady-state error due to tuning because the temperature does not oscillate around the set point; the temperature reaches the set point and drops. Other tunings were attempted, but the steady state error did not change. This led to the conclusion that the issue stemmed from the PWM programming used to operate the LabVIEW PID. If the low signal was too long during a PWM period, the gas flow would cool down too quickly. One way to solve this issue would be to reduce the length of a PWM period, but it is possible that the equipment could not operate with a smaller PWM period. Another solution would be to use distributive PWM instead of one pulse PWM.

Despite these observations, it was concluded that for the purposes of this process and the experiment done by students, this error should not result in poor data. This error should not affect the study of heat on a compressible flow. However, this error might confuse students as they try to tune the LabVIEW PID. It possible that students could use both the Omega PID and LabVIEW PID and compare their control results.

4.3 Vortex Tube

4.3.1 Calculated Hot Temperature vs. Recorded Hot Temperature

"Hot Temperature" is T_h, or the temperature of air exiting the hot end of the vortex tube. This temperature was recorded using thermocouple 10. This temperature can also be considered an unknown, and the mass flow rate can be used to calculate the value of thermocouple 10 using Equation 3.13 (for a derivation of this equation, please refer to Appendix 6.5):

$$0 = c_p[\dot{m}_c(T_c - T_i) + (\dot{m}_i - \dot{m}_c)(\boldsymbol{T_h} - T_i)] \tag{3.13}$$

Rearranged, the equation is:

$$\frac{\dot{m}_i T_i - \dot{m}_c T_c}{\dot{m}_i - \dot{m}_c} = T_h \tag{4.1}$$

All other values were recorded using equipment. Refer to Appendix 6.3 for details as to how each variable was recorded.

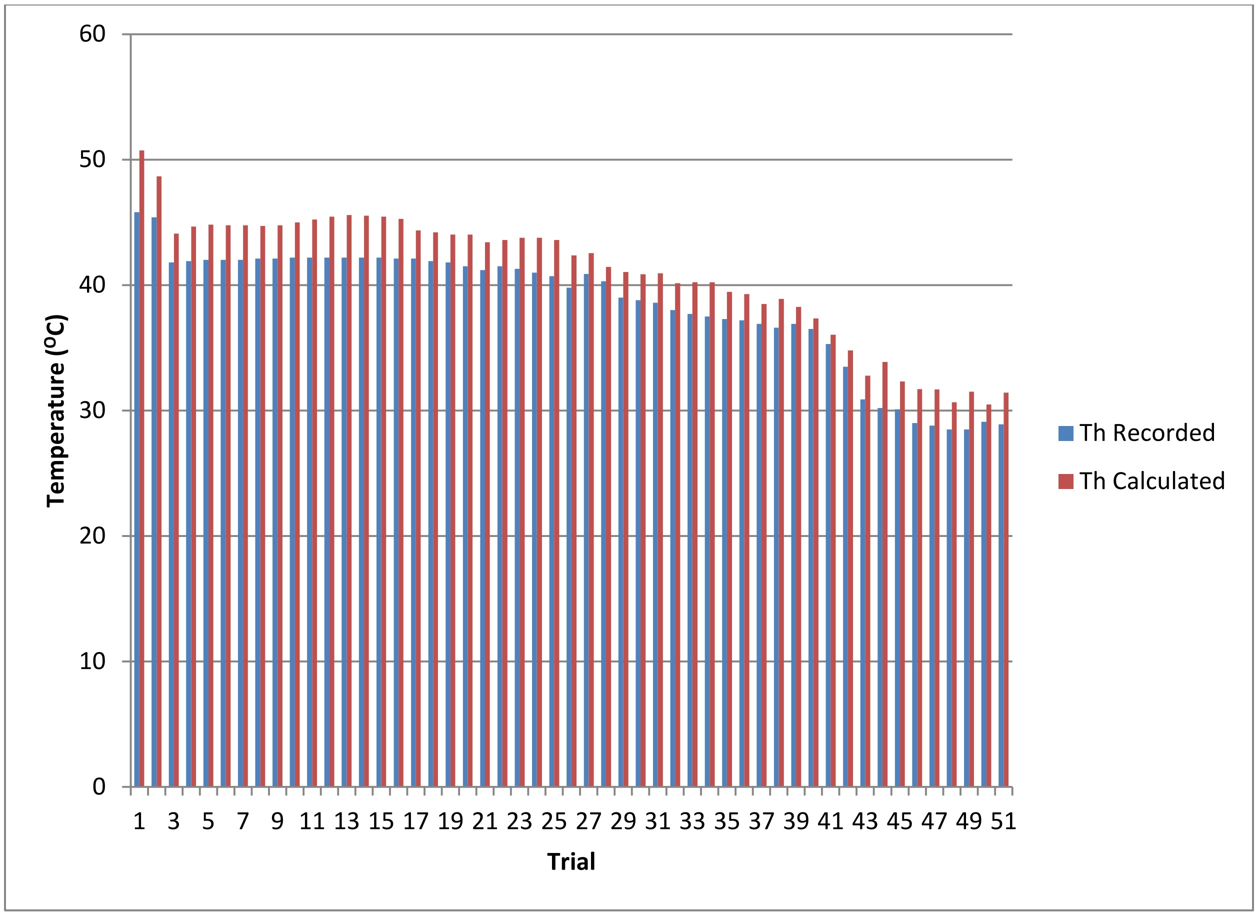


FIGURE 35: RECORDED TEMPERATURE VS. CALCULATED TEMPERATURE BY TRIAL

As seen in the chart, calculated temperatures were consistently greater than recorded temperatures. This can be attributed to heat loss throughout the system, which was not accounted for in the experiment. Also, the thermocouple is currently positioned slightly outside the vortex tube, and so the hot air had some time to diffuse into the surrounding cold air in the room and drop in temperature before reaching thermocouple 10.

Below is a chart illustrating the difference between the percent differences between the recorded T_h and calculated T_h throughout each trial:

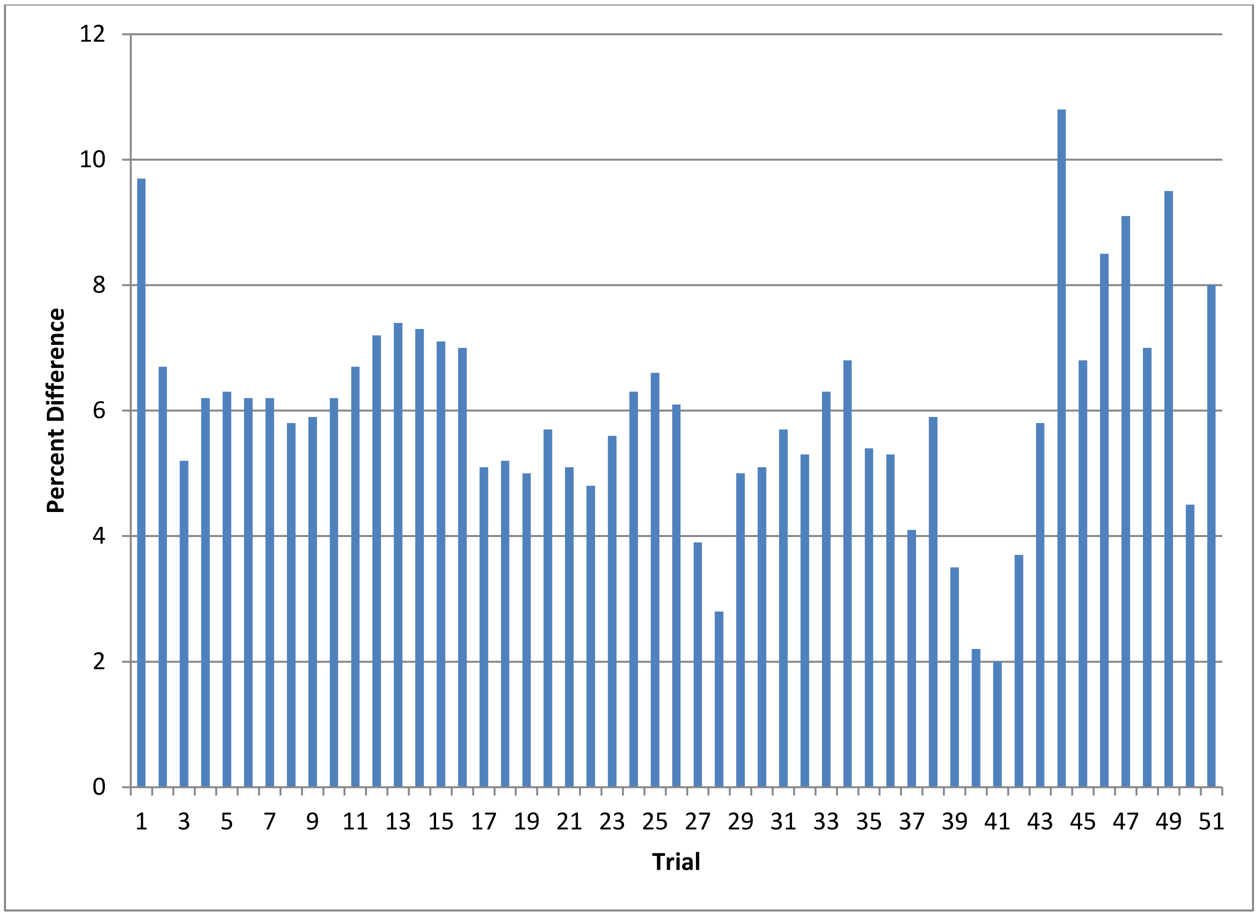


FIGURE 36: PERCENT DIFFERENCE BETWEEN T_H

The chart illustrates that there is not much significant difference between the calculated T_h and recorded T_h. This similarity between calculated and recorded values holds true as long as all measurements are taken within the same minute to account for drifting in temperature as the airflow experiment runs. If the experimenters wait too long to take measurements, the temperatures or flow rate may drift. The largest difference was 10.8 percent. The average percent difference is 5.99%, and there are very few outliers throughout the process. The standard deviation of these percentages is 1.72%. This means that most results lie in the range of 4.27%-7.72% difference. 75% (38 of 51) of the values were within the standard range. 15% (8 of 51) were within two standard deviations (a range of 2.5%-9.4%). This leaves 5 values that had percent difference greater than two standard deviations, but none of the values were greater than three standard deviations from the average (a range of 0.8%-11.2%). Of the values outside of one standard deviation, only 6 of the 51 were above 7.72%, and only 3 were above 9.4% error. 7 values were below 4.27% difference, and 2 values were lower than 2.5% error. The lack of outliers and consistent percent differences also justifies the experiment as an accurate way to estimate the temperature exiting the hot stream, so long as all data points are recorded within the same minute.

The largest percent difference in this experiment (10.8 %) between the calculated and recorded values for hot end outlet temperature is likely due to a drift towards room temperature that was observed over 45 minutes. The consequences of drifting will be addressed in section 4.3.3.

4.3.2 Calculated Cold Mass Flow vs. Recorded Cold Mass Flow

"Cold Mass Flow" is $\dot{m}_c$, or the airflow exiting the cold end of the vortex tube. The mass flow was recorded by gathering the volumetric flow from rotometer 2 and converting the flow from standard cubic feet per minute (SCFM) to grams per minute. This was accomplished using the following series of equations, which are used in the existing classroom experiment to correct the rotometer reading to account for gas compressibility:

$$Q_A = Q_R \sqrt{\frac{P_A T_S}{P_S T_A}} \qquad 4.2$$

$$\dot{m} = Q_A * \left(\frac{.3048\,m}{ft}\right)^3 * \rho_{air} \qquad 4.3$$

To find the $\dot{m}_c$ calculated through energy balance, the Equation 3.13 was used from the methodology:

$$0 = c_p[\boldsymbol{\dot{m}_c}(T_c - T_i) + (\dot{m}_i - \boldsymbol{\dot{m}_c})(T_h - T_i)] \qquad 3.13$$

This was rearranged to:

$$\dot{m}_c = \frac{\dot{m}_i \left(\frac{T_h - T_i}{T_c - T_i}\right)}{\left(\frac{T_h - T_i}{T_c - T_i}\right) - 1} \qquad 4.4$$

For discussion, the differences in temperature will be represented by the dimensionless value theta:

$$\left(\frac{T_h - T_i}{T_c - T_i}\right) = \theta \tag{4.5}$$

So that:

$$\dot{m}_c = \frac{\dot{m}_i \theta}{\theta - 1} \tag{4.6}$$

All other values were recorded using equipment. Refer to Appendix 6.3 for details as to how each variable was recorded.

Below is a chart comparing the values of recorded cold mass flow to calculated cold mass flow:

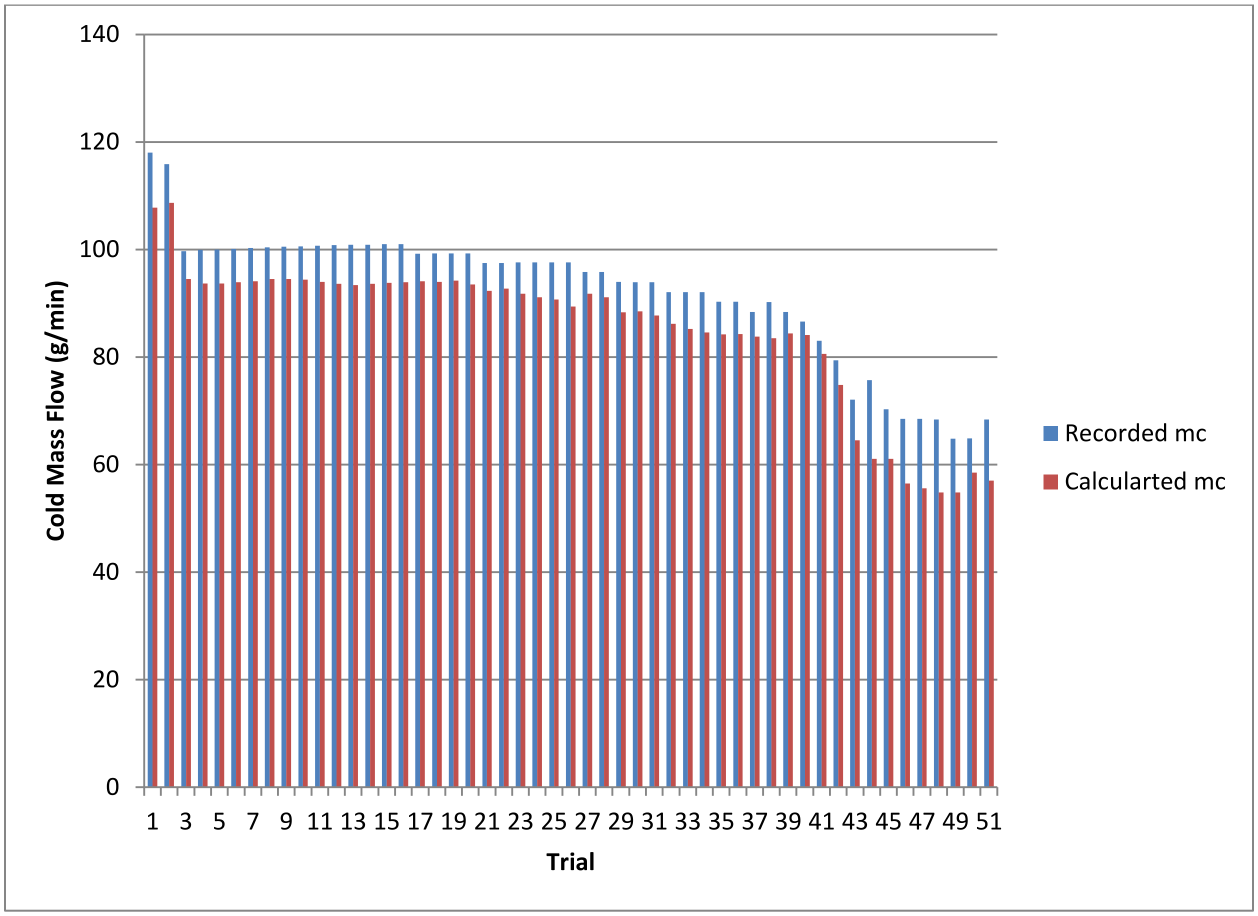


FIGURE 37: RECORDED COLD MASS FLOW VS. CALCULATED COLD MASS FLOW BY TRIAL

The recorded mass flow was always greater than the calculated mass flow. This is because the calculated mass flows depend on the temperatures from the thermocouples. Any heat loss from the vortex tube to the surroundings will increase the value of θ, the dimensionless temperature difference. As seen in previous equations, this will multiply the inlet mass flow by a number less than 1, which will decrease the predicted cold mass flow rate.

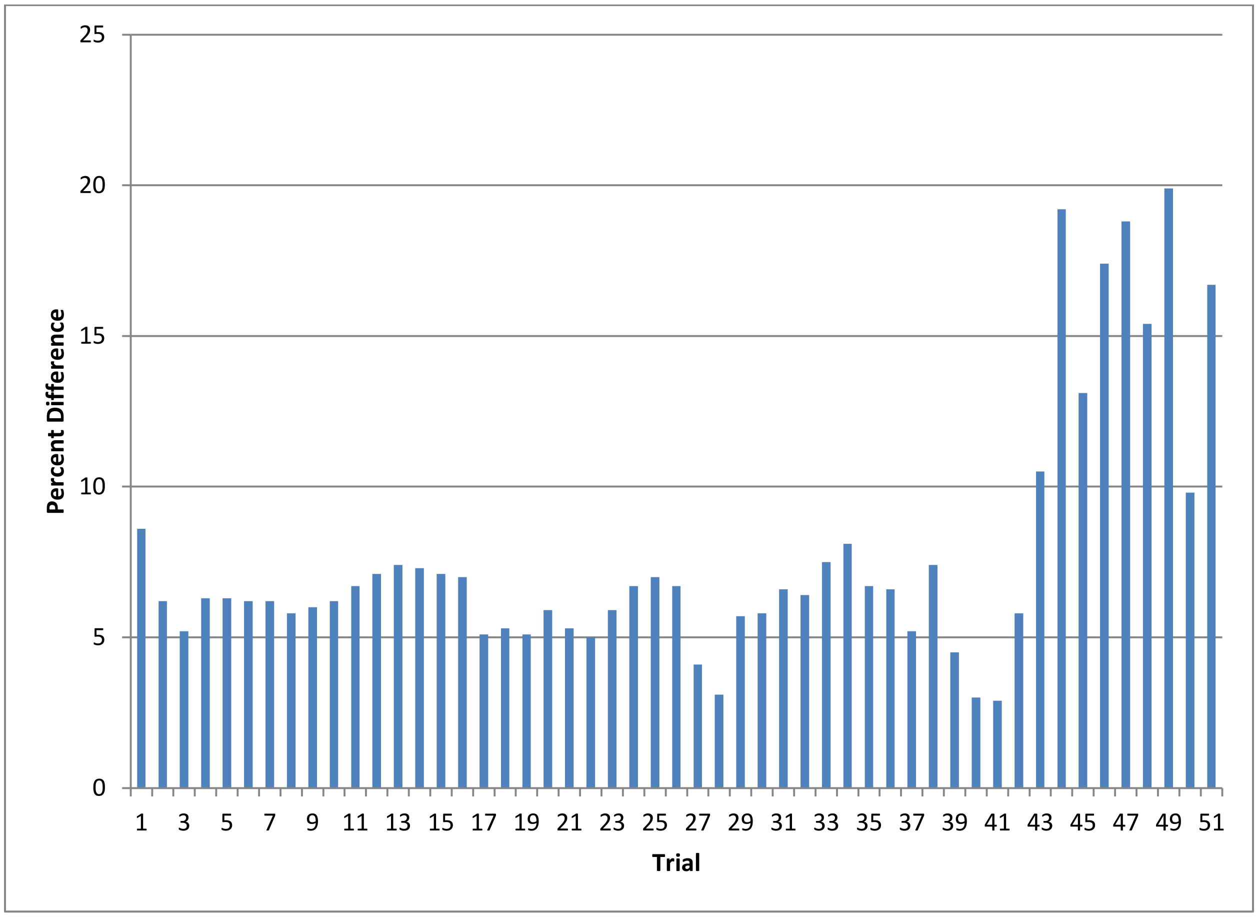


FIGURE 38: PERCENT DIFFERENCE BETWEEN RECORDED AND CALCULATED COLD MASS FLOW

The percent differences were slightly higher than the differences found between calculated and recorded hot temperatures. The average percent was 7.72%. The standard deviation of the percent difference was 4.11%. This means that all values in the range of 3.6%-11.8% difference were all within one standard deviation of the average. 80% (41 of 51) values were within this range. 10% of the values (5 of 51) were between one and two standard deviations away from the average. Three of the values were below 3.6% difference, and two of the values were between 11.8% and 15.9% different. 10% (5 of 51) were more than two standard deviations above the average. No values were more than two standard deviations below the average, and no values were greater than three standard deviations above the value. Since the standard deviation is so large, if a calculated value was more than one standard deviation above the average, then it is an indication that the data being recorded may be flawed since. There were seven trials that had this issue. All of these trials occurred after significant drifting had occurred. The later trials show an increase in percent difference, but this is due to a lack of precision in reading rotometer 2, and is an example of why due diligence is required while recording data; the rotometer readings were taken while the rotometer reading was fluctuating. This consistency of the percent difference through the most trials suggests that this is an appropriate method to estimate mass flow, while the rise in percent difference in later trials serve as a reminder that it is only appropriate if readings are done within a minute of each other.

Of the 102 trials, 93 trials had a percent difference less than 10% between the recorded and calculated values. With this established agreement between calculated values and recorded values at most experimental trials, it is likely that a simple energy balance will reasonably estimate the temperature or mass flow of a stream exiting the vortex tube. The consistent percent differences between the calculated and recorded values support the reliability of this method. Should this particular method be adapted for classroom use, it is recommended that any difference between recorded and calculated data be within 10%. This level of accuracy has been demonstrated to be easily attainable, and any larger error is an indication that there is either something wrong with the energy balance or that the data was recorded in an inappropriate manner.

4.3.3 DRIFTING

For some reason, the cold end outlet flow of the vortex tube decreases with time. This phenomenon cannot be explained by anything observed. Possible causes are that the temperature change affects the physical shape of the vortex tube, decreasing its ability to separate air before it exits the hot end. Another suggestion is that as the cold temperature decreases, the pressure in the swirl chamber decreases. The vortex tube’s ability to separate the air is dependent on the pressure of the incoming air. If the pressure decreases, then the vortex tube’s ability to separate decreases as well.

Since the cold mass flow is decreasing, this causes the hot mass flow to increase. Since the hot mass flow is larger, the hot temperature must decrease so that the energy balance is still satisfied.

Drifting occurs after an extended period of time, and is illustrated in the figures below.

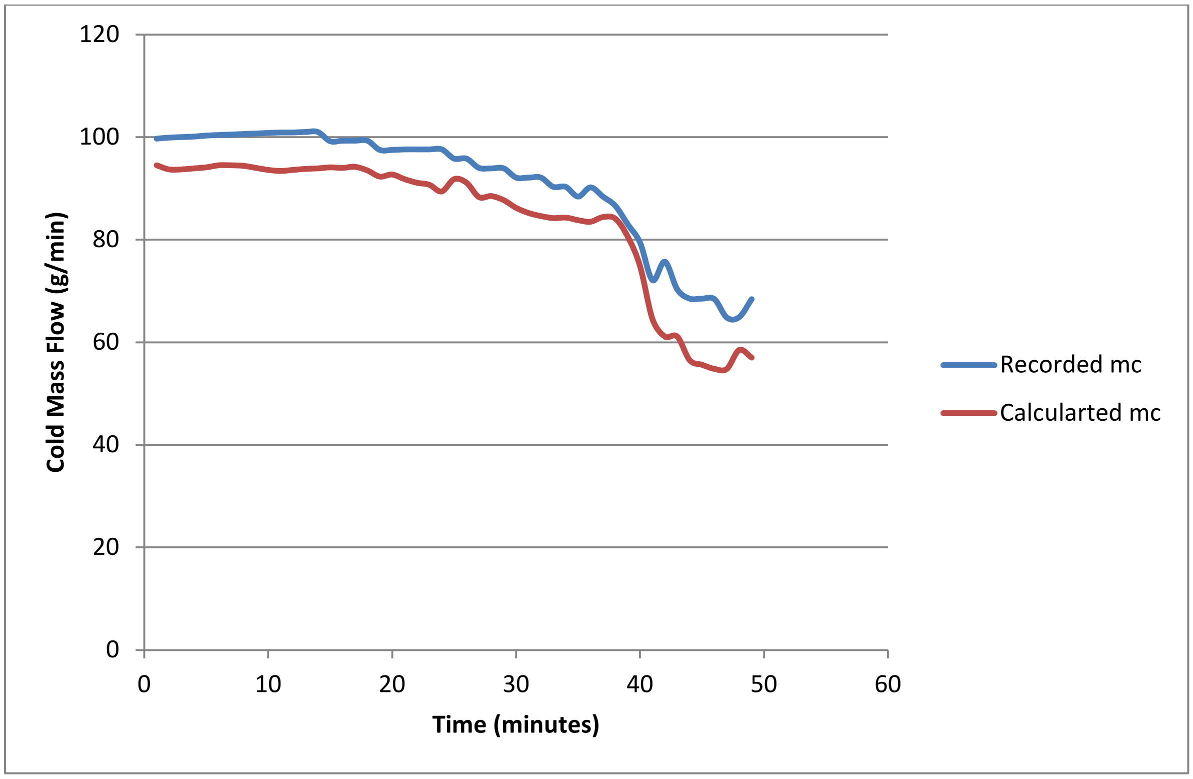


FIGURE 39: RECORDED VS. CALCULATED COLD MASS FLOW THROUGH DRIFT

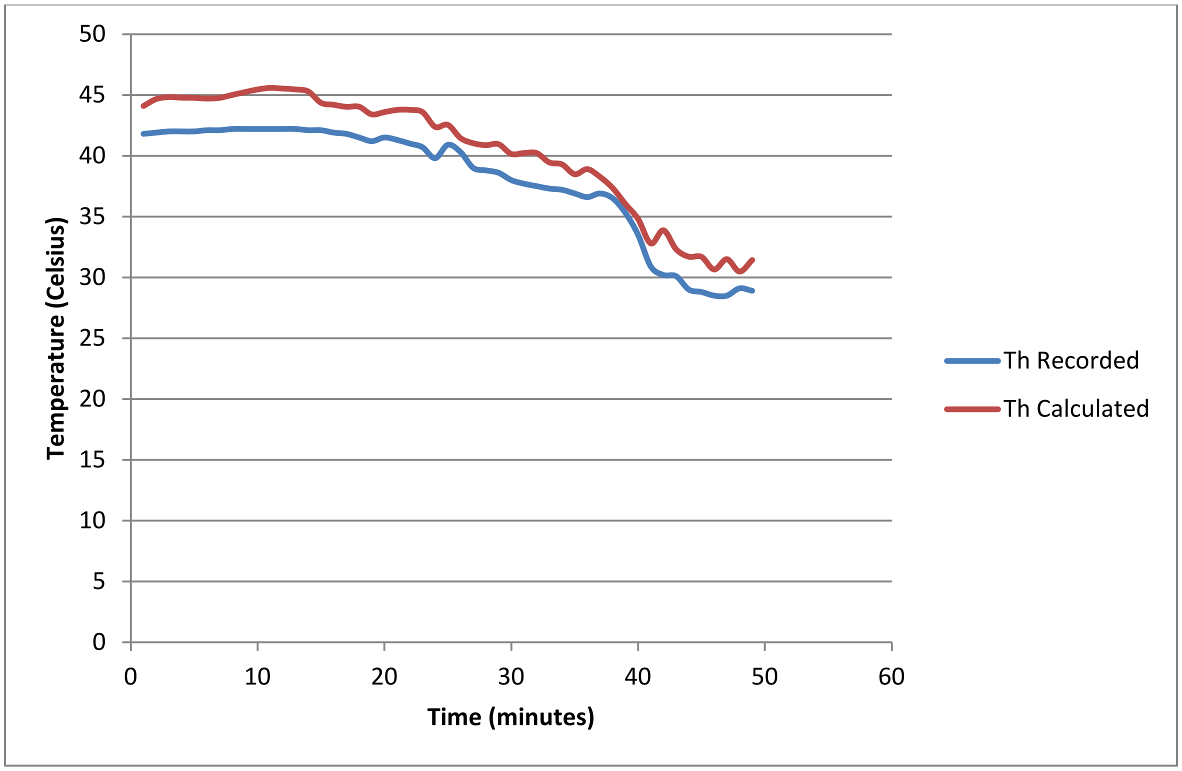


FIGURE 40: RECORDED TEMPERATURE VS. CALCULATED TEMPERATURE THROUGH DRIFT

As discussed previously, the drifting does not detract from the validity of the energy balance as long as values are all recorded within the same minute.

5 CONCLUSION

This study determined that both the LabVIEW PID and the vortex tube improved the existing Gas Flow Apparatus experiment, in both content and length. Classroom procedures were developed which will explain to students how the additions to the GFA could be effectively used in the Unit Operations course. These procedures also includ instruction in a digital replication of the PID in LabVIEW, which will allow students to learn the basics of PID control without the danger of damaging equipment on the Gas Flow Apparatus.

The LabVIEW PID successfully controlled the temperature of the compressible fluid in the experiment, and offered a more user-friendly interface to users than the Omega PID controller. Students using this experiment will now have a real-time display of the individual temperatures around the system and a real-time graph that will allow them to see how well controlled the process variable is with regards to the set point. Furthermore, students would have the option to collect data from the run, which will allow them to plot the recorded information and analyze these visual results.

The other component of this study, the vortex tube, was also tested and was confirmed to be reliable. The vortex tube will pose a feasible and interesting challenge in thermodynamics for future users. This study determined that the exercise of defining and solving an energy balance around the vortex tube is appropriate for students taking Unit Operations and will reinforce their knowledge of thermodynamics. Moreover, the vortex tube can also be used to introduce a disturbance for the LabVIEW PID. This disturbance will show students that a properly tuned controller can operate within a wide range of conditions.

Future recommendations include a more in-depth study of PWM. The current model of PWM creates a steady state error. It was determined that programming PWM as a distributed signal instead of a single pulse signal may solve the error, but the methods to accomplish this were beyond the scope of the project. Additionally, a more in-depth study of the drifting issue from the vortex tube could help identify the cause of this phenomenon. This was beyond the scope of the project because the drifting phenomenon did not affect the energy balance. In addition, for the defined classroom procedures, students are unlikely to operate the vortex tube for a prolonged period of time and will not encounter the phenomenon.

6 Appendices

6.1 Classroom Procedures

6.1.1 Classroom Procedures for PID Control

BACKGROUND

What is Process Control?

Process control is defined as the control of system parameters in a specified process. This discipline is applied widely in academia and industry to maintain stable processes and experiments; activities ranging from home heating systems to nuclear plants use process control systems of varying complexity.

For the control of a specific process a set-point is needed for the controller. Based on this set value the controller will drive the actuator, a device that performs a mechanical response to an input signal, as close to the desired value or effect as possible.

As a representative example, consider the cooling system of a room. The person inside the room sets the temperature of Air Conditioning (AC) to a certain temperature, given his needs to cool down on a hot summer day. This temperature is the set-point of the system. The AC will work to reach and maintain that temperature. The AC will have a sensor that monitors temperature and provides feedback to the controller. This means that if the temperature sensor gets a reading off the set-point, it will drive the cooling equipment to get it to target; either by varying the supply of cold air to the room, or by ceasing to do so, depending if the temperature is too high or too low.

A PID controller is a controller system widely used in industry and which will be used in this experiment, via the instrument control and data acquisition program LabVIEW.

What is a PID Controller and what are its uses in industry?

The term "PID Controller" stands for Proportional, Integral, and Derivative Controller. In this case the nomenclature is straightforward; the controller is made up of proportional, integral, and derivative mathematical components (Ang, 2005), which will be discussed in-depth later in this document.

PID controllers are extremely important instruments in the chemical industry. Their usage is extremely adaptable and extends to the chemical, petrochemical, paper, food & beverage, water treatment/sewage facilities, and many more. PID controllers are used to control process variables as diverse as fluid flow, fluid level, pressure, temperature, pH, consistency, density, position, concentration, etc. Having some idea or instruction in this control device and in process control in general will be extremely beneficial when working in the industry.

This pre-lab will introduce the student to some basic theory on how a PID controller operates and how to tune a specific system. In this experiment, the heater located in the Airflow Experiment in GH 220 (UO lab) will be used. The experiment will consist of tuning an un-optimized system, with the ability to change specific functions. Specific goals will have to be achieved, including an acceptable rise-time, acceptable oscillation around the set-point and acceptable time to reach steady-state.

CONTROL THEORY

As mentioned above, the PID controller works under three variable terms. Each of these has a set of specific ways in which it affects the system it controls.

The **Proportional** term is expressed mathematically as:

$$X = K_C(x_{set} - x) + c$$

where all of the terms are set by the process, except for the K_c value that is controlled by the operator. The proportional term allows the controller to respond in proportion to deviations from the set-point. This term is especially useful for increasing the controller's speed of response, although if the term is too large, the process variable will begin to oscillate. If increased even further, then the system might become unstable and even oscillate out of control.

The **Integral** term is expressed mathematically as:

$$X = \frac{1}{\tau}\int_0^t (x_{set} - x)dt$$

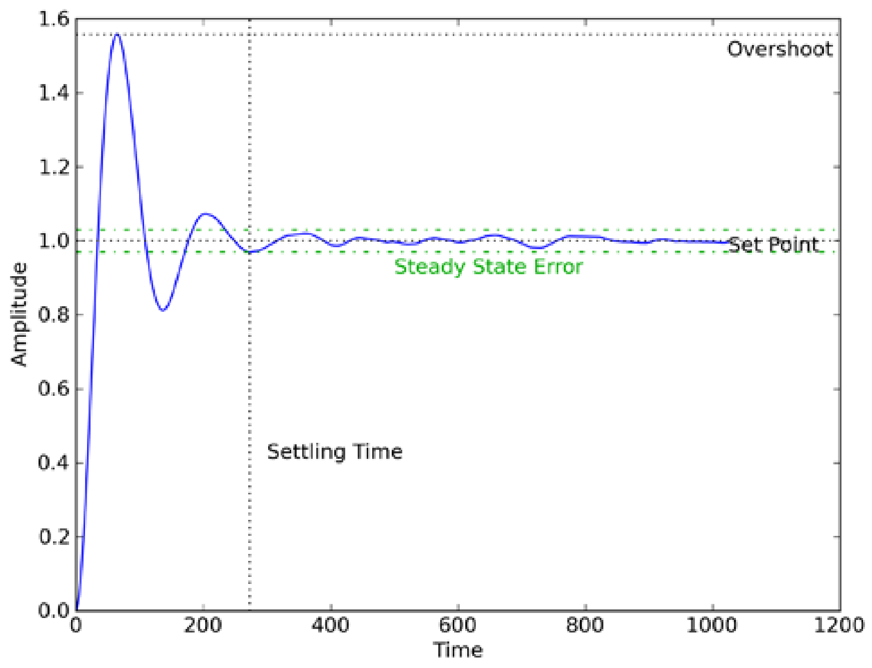
This term allows the controller to respond to errors that have occurred historically over time. So, if the system has deviated from the set-point in a consistent manner, this term will force the PID controller to correct this deviation more powerfully. This term adds long-term stability to the system, as it eliminates the repetitive deviations caused by the proportional term and serves to eliminate steady-state error. A common error associated with this term is "integral windup", where integral term is so large, that its action saturates the controller without the controller driving the error signal towards zero.

The **Derivative** term is expressed mathematically as:

$$X = \tau_d \frac{d(x_{set} - x)}{dt}$$

Knowing that the proportional term responds to present error and that the integral responds due to past errors, the derivative term sure enough responds to correct future error. The way this works is the derivative term is able to "see" where the set-point is and is able to steer the process variable regarding how fast to approach it. The derivative term especially affects the start-up of the system (rise-time), as it helps the process variable reach the set-point much faster. One issue associated with this term is that it is extremely sensitive to noise in the process variable, so if this term is too large, any small variance will cause it to destabilize the system.

How to evaluate a system



http://nasa.olin.edu/projects/2010/tec/work_introCT.htm

When attempting to optimize a system, there are many system properties which can be changed to make the process run more smoothly. Usually, optimization is focused around four specific system dynamics, listed below (Zhong, 8):

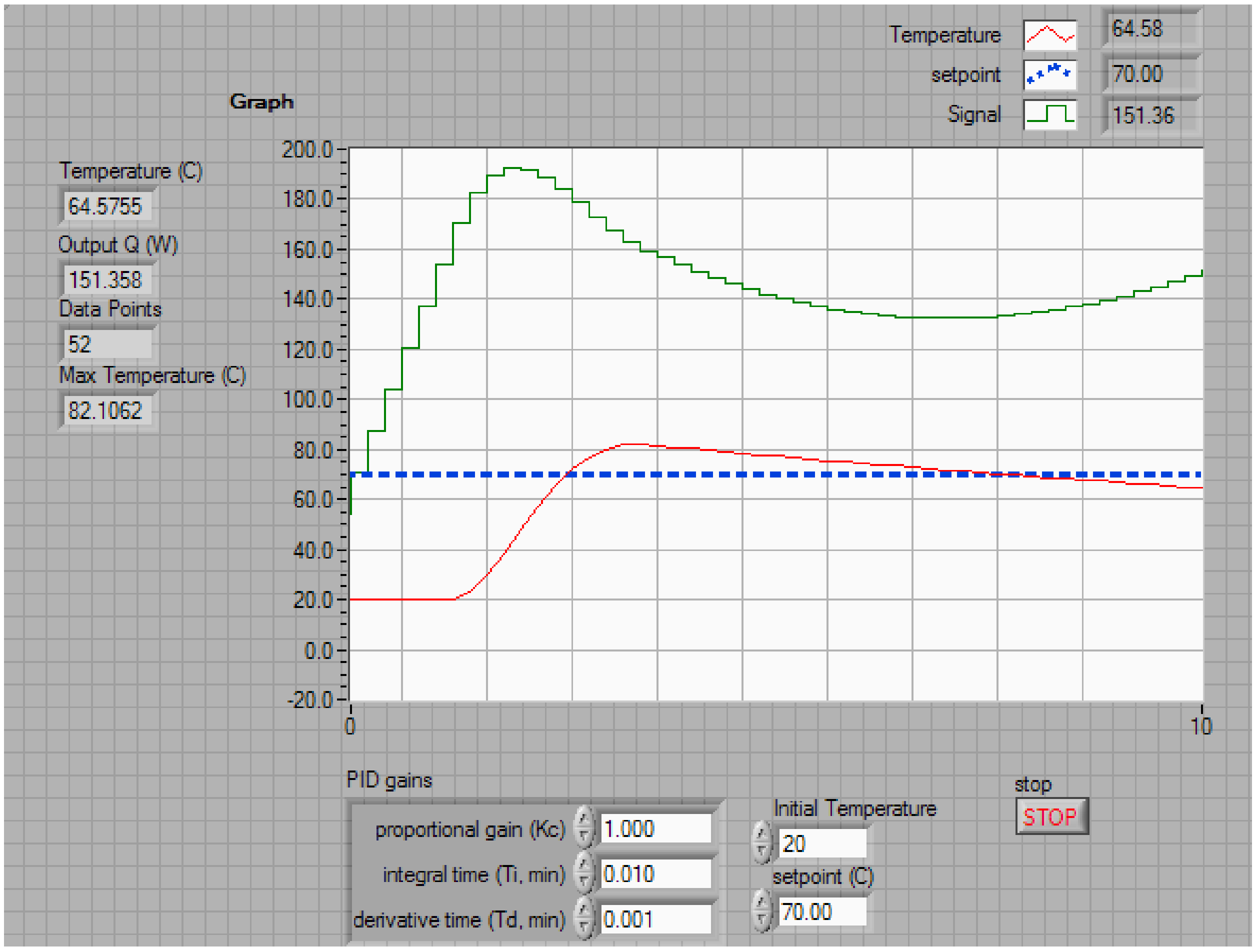
1. Rise time: the time it takes for the output value to reach 90% of the set-point value for the first time.
2. Overshoot: expresses by how much the highest output value is different to the set-point value, normalized against steady-state.
3. Settling time: the time it takes for the system to converge to steady-state.
4. Steady-state error: the difference between the converged steady-state output and the desired set-point value.

These represent the four major characteristics of a closed-loop step response system. This experiment will focus on optimizing these four properties, and each should be addressed separately when attempting to measure the progress made in optimizing the system.

Virtual LabVIEW PID

It is important that students familiarize themselves with LabVIEW's front panel and with the application of PID theory. For this purpose, a completely digitalized version of the GFA has been created and is available for students to explore.

In order to open up this file, download the VirtualPID.vi file from myWPI. Once it is loaded in LabVIEW, you should see the front panel shown in the figure below:



Temperature 64.58
setpoint 70.00
Signal 151.36

Temperature (C) 64.5755
Output Q (W) 151.358
Data Points 52
Max Temperature (C) 82.1062

proportional gain (Kc) 1.000
integral time (Ti, min) 0.010
derivative time (Td, min) 0.001

Initial Temperature 20
setpoint (C) 70.00

The gain values can be changed by entering the desired number and are located at the bottom of the graph. The graph plots 3 curves simultaneously:

1. Set point (blue)

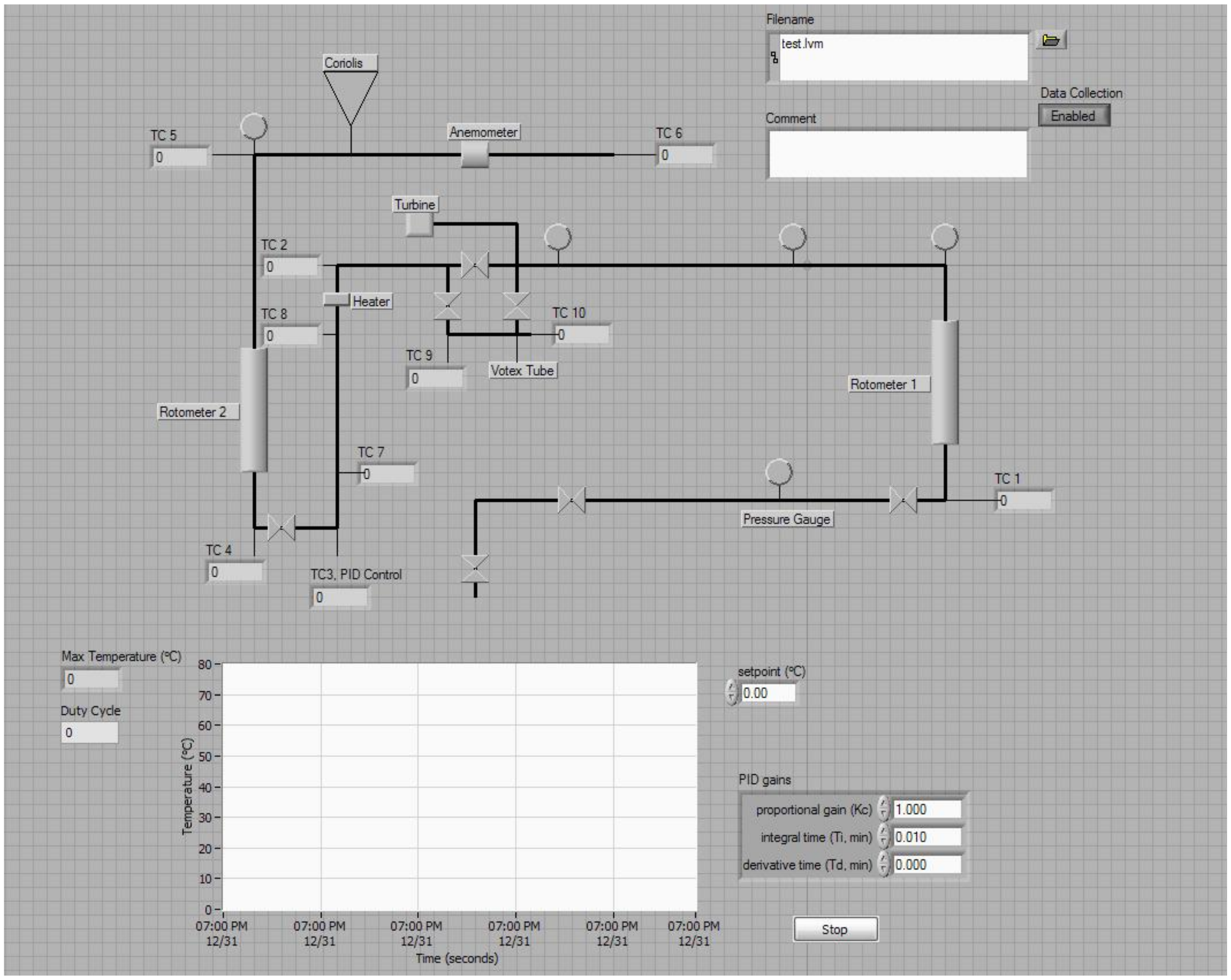
2. Process variable (red)

3. Signal (green)

For the purpose of this experiment, first take and record a run with the unoptimized PID settings; save this graph and data for reference. Then, input your initial values and keep all data for further processing.

Physical LabVIEW PID

This experiment will be using LabVIEW for the PID controller. This means that the physical Omega controller will not be used, but a LabVIEW one will be used instead. This is an easier to tune PID and provides visuals as to what is occurring. Below is a screenshot of the entire control panel of the LabVIEW PID used in this experiment:



Filename: test.lvm

Data Collection: Enabled

Max Temperature (°C): 0

Duty Cycle: 0

setpoint (°C): 0.00

PID gains

proportional gain (Kc): 1.000

integral time (Ti, min): 0.010

derivative time (Td, min): 0.000

The physical PID control is the same as the virtual PID control

A schematic of the gas flow system is displayed above the graph with real-time displays of the temperatures at their respective thermocouples. This will allow for an instantaneous monitoring of temperatures around the system.

Finally, this LabVIEW controller allows you to record the data and export it to Excel. This should be done for the runs that demonstrate good control, in order to plot the data and include the graphs in your reports.

EXPERIMENT

It is imperative that during the pre-lab you use the virtual LabVIEW PID so that you understand what values to use for the PID gains. There ARE safety alarms in place at the GFA that will be activated if the max temperature exceeds 80^{o}C

For this experiment, the gas flow apparatus in the second floor of the UO lab will be used. The heater will be the focus of this experiment because it is controlled by the PID. It is required that the system operates at a constant temperature of 70 degrees Celsius. It is your job to properly tune the PID so that the system achieves this temperature.

To get the system started:

1. Log into the computer CHEM07 (right next to the gas flow apparatus)
2. Open LabVIEW and load the file PIDGFA.vi from myWPI.
3. Turn on the main air line and all subsequent valves
4. Check that the rotometers are giving a flow reading.
5. Set the flow of air to 8 SCFM
6. Type the value of 70^{o}C into the set point text box in the front panel of LabVIEW.
7. Tune the PID gains to your desired settings
8. Be sure that the switch next to the computer has the “LabVIEW PID” enabled and NOT the Omega PID.
9. If the team wishes to record the data for the run, make sure the “Data Collection” button at the top right is clicked and says “Enabled”.
10. Wait for the hot-wire anemometer to recognize that the flow of air is sufficient, by displaying a red light.
11. Turn the Variac ON, and then run the LabVIEW program.

12. After every run is complete, be sure to press the “STOP” button on the front panel.
13. Turn off the Variac
14. Allow the GFA to return to room temperature before another run is executed. Repeat steps 7-13 until you have achieved good control and have logged all necessary data. This is accomplished when the GFA reaches the established set point:
    a. In low rise-time
    b. With minimal overshooting
    c. In Short settling time
    d. And maintain steady state with minimal oscillation

Be sure to run the system within the ranges given, since this will ensure the safety of the airflow system, its equipment (which is expensive), and the students themselves who are doing the experiment.

Variable	Range
Proportional gain	0-20
Integral time (min)	.001-5
Derivative time (min)	.001-.05
Set Point	70 ^{0}C

Some questions to be thinking about as the experiment is being ran:

- How did each term affect the system?
- Were there any terms that were best kept at low values (relative)?
- Was there instability in the system that could be easily attributed to a specific term?

REPORT

For this experiment, a report is to be handed in highlighting the team’s findings, difficulties, and final results. Please discuss:

- What gains worked and what gains did not work in this experiment? Why did the final set of gains work so effectively? Include a graph of the final control run and any other appropriate runs.
- What was the hardest gain to tune in the experiment?

- Imagine the Vortex Tube was turned on during the middle of the experiment. What kind of effect would this have on the control of the process and the PID?
- What potential sources of error can you think of (human, ambient, etc.)?

PRE-LAB QUESTIONS

1. What does the term PID stand for?
    a. Proportional Instantaneous Derivation
    b. Proportional, Integral and Derivative
    c. Piping and Instrumentation Diagram
    d. The acronym cannot be defined, as the context is not clear

2. Which of these is not considered a system property to be optimized?
    a. Rise time
    b. Sampling time
    c. Steady-state
    d. Overshoot

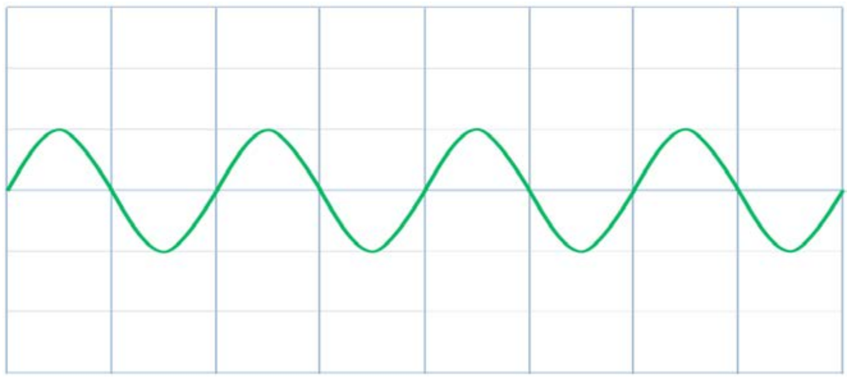


FIGURE A: OSCILLATION OF PROCESS VARIABLE AROUND A SET POINT

http://www.widexconnect.ca/hip/images/sine-wave-lg.gif

3. Figure A shows the oscillation of a process variable around the set-point. If the system's speed of response is set by outside parameters, what term could be changed to obtain less oscillation around the set-point?
    a. Proportional term
    b. Integral term
    c. Derivative term
    d. Oscillation is only dependent on the quality of the controller

4. In reference to Figure A, if decided that a specific term would decrease oscillation around the set-point, would the term be:
    a. Increased
    b. Decreased
    c. Oscillation is only dependent on the quality of the controller

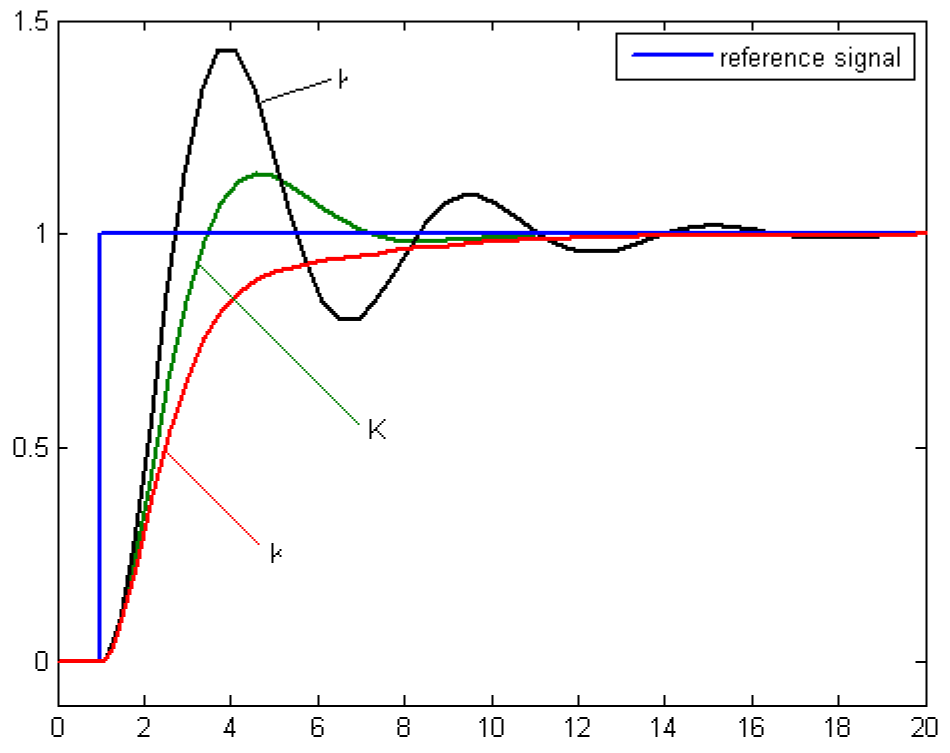


Figure B: Three process variable curves, each with distinct integral term values

http://wiki.aasimon.org/lib/exe/fetch.php?cache=cache&media=marvin:lab4-ki.png

5. Figure B shows three process variable curves plotted on the same graph. These graphs have the same P and D terms, but different I terms. Which curve is the most likely to display the highest integral term?
    a. Black
    b. Green
    c. Red
    d. Blue

6. In reference to Figure B, which term would be best to reduce the overshoot of the process variable above the set-point?
    a. Proportional term
    b. Integral term
    c. Derivative term
    d. Overshoot is a property inherent of the system itself and can't be controlled.

7. What is the main drawback of using a high derivative term?
   a. Will affect set-point value
   b. Extreme sensibility to process variable noise
   c. Increased overshoot
   d. None of the above

6.1.2 *Classroom Procedures For Vortex Tube*

Goal of the experiment: Develop an energy balance around a Ranque-Hilsch Vortex Tube

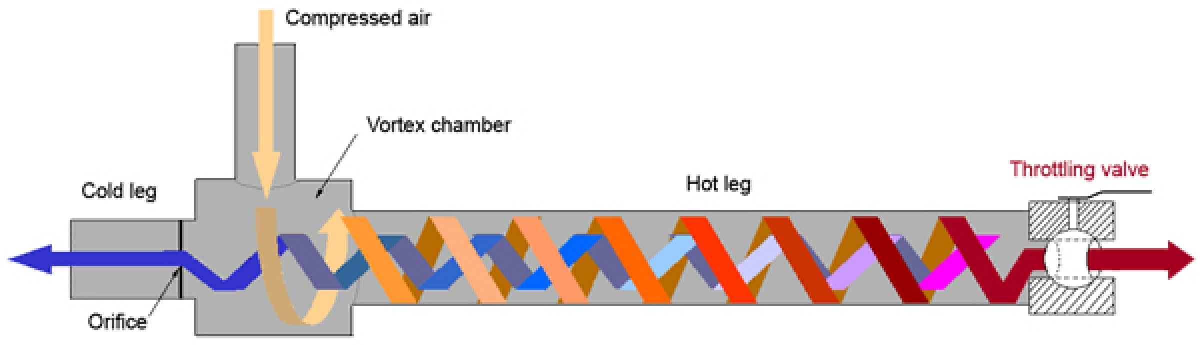
<u>What is a Vortex Tube?</u>

Vortex tubes are T-shaped tubes that are able to separate compressed air into simultaneous hot and cold streams. A picture of a Vortex Tube is shown below:

(http://www.process-controls.com/techsales/Nex_Flow/images/vortex_tubes/vortex_tube_new.jpg)

This equipment requires no moving parts to separate the streams. Inside has a corkscrew shaped lining, referred to as a swirl chamber. It is this design that separates the faster moving particles from slower moving particles. This can be used in multicomponent separation by density. Vortex tubes have several practical applications, such as a method of point cooling in manufacturing or as a method for enriching Uranium. In Goddard Lab, we will be performing a single component separation using compressed air.

In this single component separation, the faster and slower particles correspond with hot and cold molecules. As compressed air is fed through the inlet at the top of the T-shaped tube, the air flows into the swirl chamber, resulting in a vortex. The understanding of the in-depth fluid mechanics that cause this vortex is beyond the scope of this course.



http://www.pdbuchan.com/ranque-hilsch/ranque-hilsch.html

The diagram above shows the Vortex Tube and a simple explanation of how it works. A control valve at the end of the "hot leg" determines the quantity and temperature of both streams. This particular experiment does not need a specific position of the valve, but it does need to be open (some hot air must be flowing out of the "hot leg").

The First Law of Thermodynamics

From the "Applied Chemical Engineering Thermodynamics" class, you should be familiar with the three laws of thermodynamics. This experiment is specifically concerned with the first law. With the temperatures that the Vortex Tube allows on both streams, it gets hot enough to boil water at the hot end and freeze water at the cold end. How is this possible?

This would seem to violate the first law of thermodynamics, which postulates the conservation of energy. In order to prove that the Vortex Tube is not violating the First Law of Thermodynamics, an energy balance needs to be performed around it.

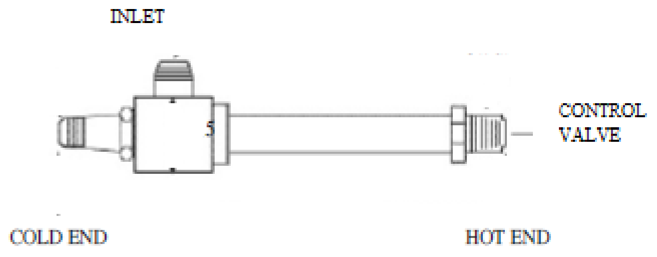
Measurements

You need to evaluate what measurements to take in order to perform the balance. At your disposal, you have thermocouples and rotometers, both before and after the Vortex Tube. *Hint: Don't forget to correct the rotometers!*

Step-by-step procedures

Steps to control the vortex tube:

1. Turn on the air flow
2. Open the ball valve before and after the vortex tube
3. Close the bypass valve
4. Verify that air is flowing out of the hot end of the vortex tube. If not, open the control valve.



5. Record temperature from the appropriate thermocouples
6. Record the reading from the rotometer after the vortex tube. Record the gage pressure.
7. Wait for the vortex tube to reach steady state
8. Record the temperature from the appropriate thermocouples
9. Record the reading from the rotometer near the beginning of the flow. Record the gage pressure.

Results

-Mathematically prove that the Vortex Tube doesn't violate the first law of thermodynamics. (Hint: Are the predicted and measured variables similar?)

-Qualitatively prove that the Vortex Tube doesn't violate the second law of thermodynamics

6.2 Glossary

6.2.1 Definitions

Analog signal – An analog signal is a continuous time varying sinusoidal electrical signal. For the purposes of this document, an analog signal represents a variable that is being read by an object. An analog signal registers all variances that occur.

Bang-bang control – Bang-bang control is a type control that administers a signal to turn on a device if the process variable is below the set point, or turn off the device if the process variable is above or at the set point. Bang-bang control does not offer variance in the power administered.

Boolean – Boolean is a logic algorithm in programming. In the scope of this project, it offers a true or false condition.

Case structure – A case structure is a programming structure that alters a signal based on conditions being met throughout the program. The condition is either true or false. The case structure offers a response for both situations.

Cluster – A cluster is a LabVIEW programming operation that bunches several different types of data together for communication with another device in the program.

Cold fraction – Cold fraction is the ratio of cold product produced to the warm product exiting the system. In the case of a vortex tube, this can be more easily defined as the ratio between the cold exhaust flow to the warm exhaust flow.

Control loop – A control loop is a system which includes a controller that communicates with a regulating device. The regulating device is examined by an indicator device. The indicator device then communicates with the controller.

DAQ assistant – A DAQ assistant is a programming device in LabVIEW that allows the programmer to use the information supplied by the Data Acquisition Device (DAQ) in his/her virtual instrument.

Derivative kick – Derivative kick is a phenomenon in which a user changes the set point, and an error is registered by the PID because the process variable is identified as being no longer at the set point. This causes a rapid response to reach the new set point, usually resulting in drastic overshooting.

Digital signal – A digital signal is a pulse train signal that switches between a high value and low value voltage.

Duty cycle – A duty cycle is the percentage of time a digital device is emitting a high signal pulse throughout a PWM period.

Feedback loop – A feedback loop is a control loop in which a controller responds to errors after the process has affected the process variable.

Feedforward loop – A feedforward loop is a control loop where the controller responds to errors before the process has affected the process variable.

For loop – A for loop is a programming structure that operates its interior functions (programming) a predetermined number of times while the program is running.

Integrator windup – Integrator windup is a phenomenon that occurs when the integrator control causes a controller to change a process faster than the device can operate. This causes the controller to see more error, and thus the problem builds on itself.

Local variable – A local variable is a LabVIEW programming function that allows the programmer to define a value to a variable.

Noise – Noise is a random fluctuation in an electric signal.

Not node – A not node is a LabVIEW programming function that sees a function and returns the logical opposite. In other words, if it sees a true function, it will return the false response.

Offset - Offset, or error, from a set point is the difference between the value of the process variable and desired value (set point) of the process variable.

Proportional time – Proportional time is the cycle time with which a proportional gain reacts to error while within a Proportional Band.

Proportional, integral, and derivative (PID) control – PID control is a self-described feedback control type. The proportional, integral, and derivative terms are all defined in a PID equation (discussed in the background) that determine the strength with which the controller responds to offset in a process.

Pulse – A pulse is a rapid change in a signaling value from a low value to a high value. This is followed by another rapid change from the high value to the previous low value.

PWM period – A PWM period is the arbitrary cycle time in a pulse-width modulation process. Each distinct duty cycle is administered throughout one PWM period.

Set point – Set point is the predetermined value that the user wants the process variable to attain.

Shift register – A shift register is a programming tool in LabVIEW that lets a loop structure carry over a value in the structure from iteration to iteration.

Tunneling – Tunneling is a LabVIEW programming option where a value entering a loop structure is unaffected by the loop's functions and is available for indexing throughout each iteration.

While loop – A while loop is a programming structure that operates its interior functions continuously until a certain condition is met.

6.2.2 *Notation*

A_S: Internal surface area of heat element

c: Proportional constant

c_p: Specific heat capacity

dT: Temperature difference at entrance and exit of heater

ε: Cold fraction

H: Enthalpy

K_c: Proportional control gain

$\dot{m}$: Mass flow rate

P: Pressure

ρ: Density

q: Heat transfer

Q: Volumetric flow rate

T : Temperature

t: Time elapsed

τ_d: Derivative control gain

$1/\tau$: Integral control gain

τ : Integral time

θ: Dimensionless temperature

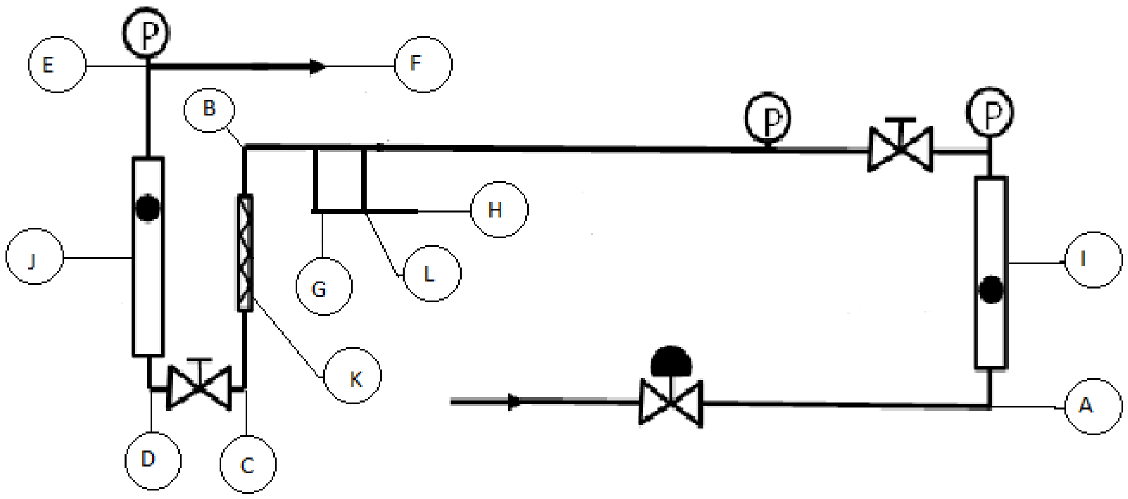
X: Parameter of system that can be controlled

x: Current value of manipulated variable

x_{set}: Set point of manipulated variable

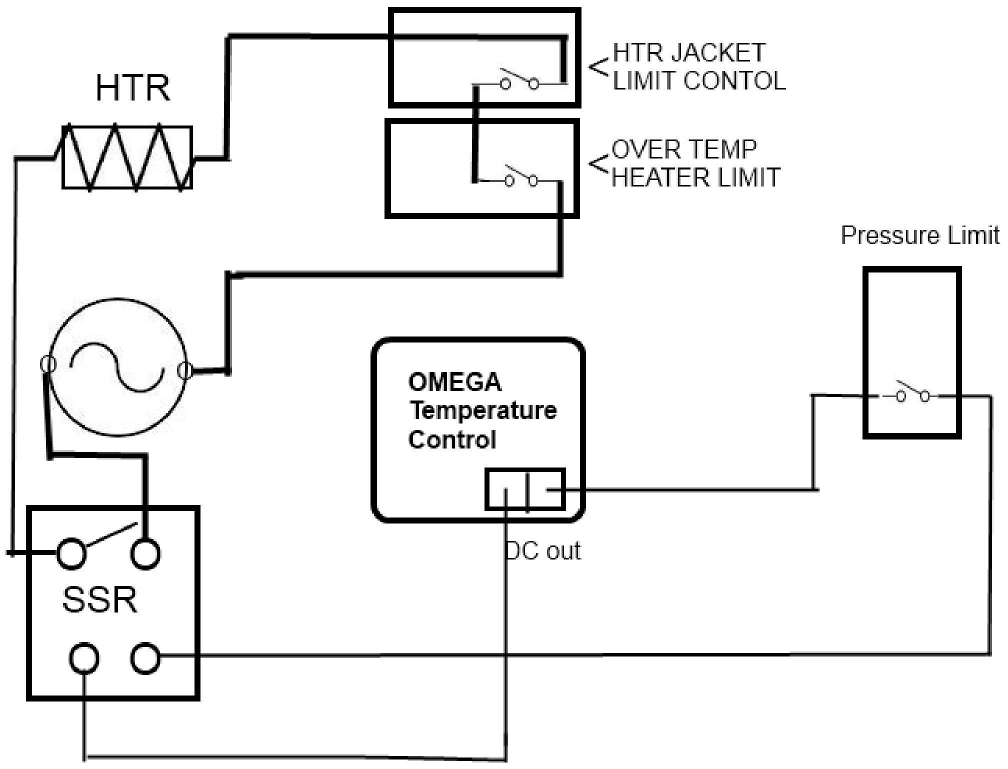
6.3 SCHEMATIC

6.3.1 *GAS FLOW APPARATUS*



Designation	Equipment	Description	Variable (calculations)
A	Thermocouple 1	Inlet temperature	T_i
B	Thermocouple 2	Temperature before heater	
C	Thermocouple 3	Temperature after heater PID process variable	T
D	Thermocouple 4	Temperature after valve	T_4
E	Thermocouple 5	Temperature after rotometer 2	T_5
F	Thermocouple 6	Temperature exiting system	
G	Thermocouple 9	Temperature of cold mass flow	T_c
H	Thermocouple 10	Temperature of hot mass flow	T_h
I	Rotometer 1	Inlet volumetric flow (Use thermocouple 1 for temperature)	Q_i ; m_i
J	Rotometer 2	Outlet volumetric flow (Use an average of T_4 and T_5)	Q_c ; m_c
K	Heater	Controlled by PID	-
L	Vortex Tube	Bypass option	-

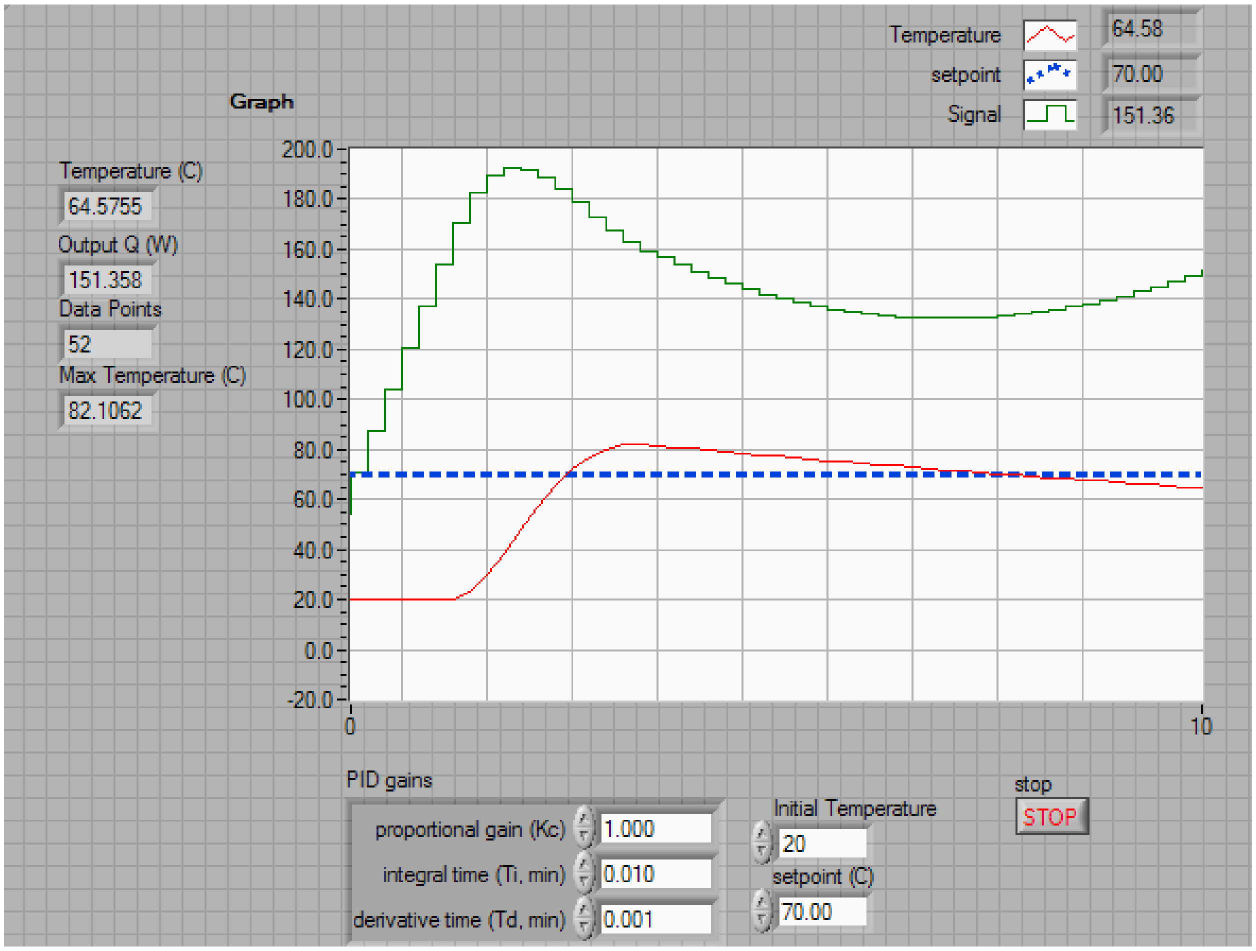
6.3.2 *Wiring Diagram*



6.4 LabVIEW

6.4.1 Virtual

Front Panel



Temperature	64.58
setpoint	70.00
Signal	151.36

Temperature (C): 64.5755

Output Q (W): 151.358

Data Points: 52

Max Temperature (C): 82.1062

proportional gain (Kc): 1.000

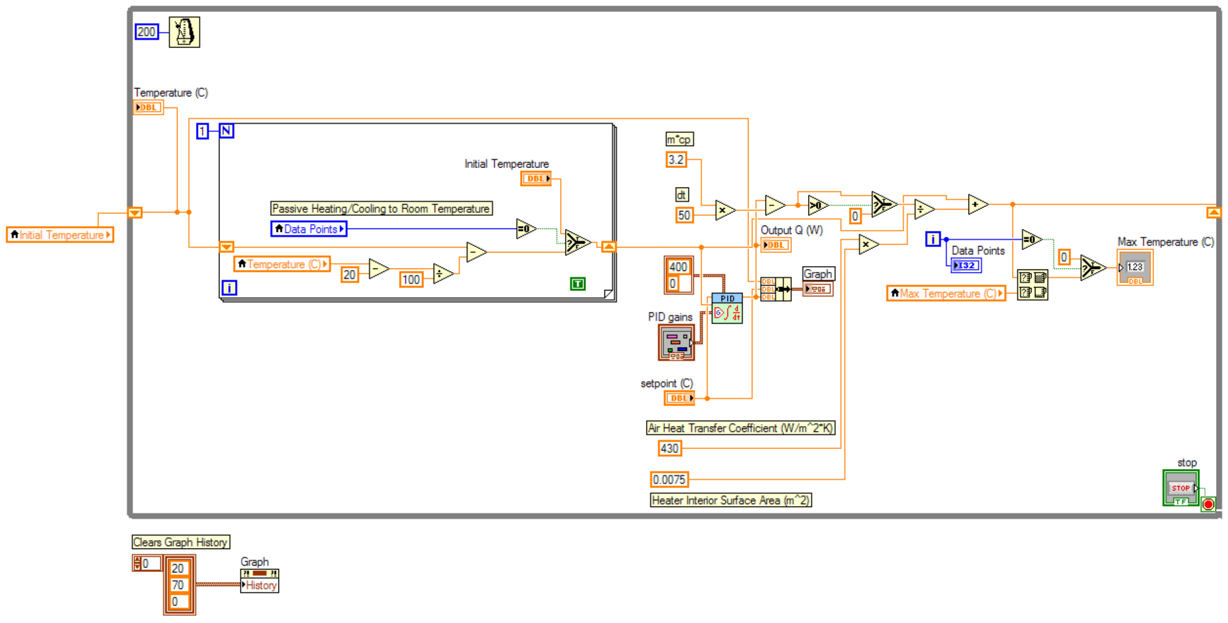
integral time (Ti, min): 0.010

derivative time (Td, min): 0.001

Initial Temperature: 20

setpoint (C): 70.00

Back Panel



Derived Equation and Sample Calculations for PID output

Representative values from a sample experimental run are given in Table 12:

TABLE 12: DATA FROM A TYPICAL EXPERIMENT

Inlet Temperature (°C)	20
Controlled Temperature (°C)	70
Inside Diameter (in)	0.62
Inside Diameter (m)	0.015748
Heated Length (in)	4.5
Heated Length (m)	0.1143
Internal Heater Surface Area (m^2)	0.111316052
Cross-Sectional Area (m^2)	0.000194778
Representative Volumetric Flow (SCFM)	6.628931334
Representative Volumetric Flow (m^3/s)	0.003128502
Representative Velocity (m/s)	16.06185681

The only values that have the proper significant figures in Table 12 are the inside diameter (in inches) and the heated length of the heater (in inches). The extraneous figures are retained for accurate calculations. The volumetric flow rate (SCFM) was taken from Run 3 in the vortex tube calculation sheet (Appendix 6.6.2). The required heat transfer to make any change in temperature is given by equation 3.2:

$$q_1 = \dot{m} * c_p * dT$$

where dT is given by:

$$dT = (T_3 - T_i)$$

The mass flow rate is unknown for a given experiment. However, the volumetric flow rate is known. Assuming density will not greatly change over 40 degrees, the mass flow rate can be calculated via equation 3.3:

$$\dot{m} = Q * \rho$$

Substituting values and unit analysis yields:

$$\dot{m} = (0.003128502) * (1.028) = 0.0032161 \left(\frac{kg}{s}\right)$$

$$\dot{m}[=] \left(\frac{m^3}{s}\right) * \left(\frac{kg}{m^3}\right) = \left(\frac{kg}{s}\right)$$

Obviously the figures for the resulting mass flow rate of air are not significant. However, they will be retained for accurate calculations, as will the other equations in this section.

With the mass flow rate solved, the required heat transfer for the flow can be solved:

$$q_1 = \dot{m} * c_p * dT$$

$$q_1 = (0.0032161) * (1007) * (70 - 20)$$

$$q_1 = 161.9306402\ (W)$$

$$q_1[=] \left(\frac{kg}{s}\right) * \left(\frac{J}{kg * K}\right) * (K) = \left(\frac{J}{s}\right) = Watts$$

This result for the required heat transfer applies at every second of the heater's operation. Heat transfer resistance theory states that since the driving force of heat transfer is a temperature gradient within a system, any other effects can be "lumped" into a single, total resistance.

$$q = \frac{dT}{R_{Tot}}$$

Newton's Law of Cooling can be used to define heat transfer (q) into a heat transfer coefficient if the heater's internal surface area is known.

$$q = h * A_s * dT$$

With this equation the total resistance is:

$$\frac{1}{R_{Tot}} = h * A_s$$

However, h must first be solved.

$$h = \frac{q_1}{A_s * dT}$$

$$h = \frac{(161.9306402)}{(0.111316052) * (70 - 20)}$$

$$h = 29.09385243 \left(\frac{W}{m^2 * K}\right)$$

The units of h are apparent.

The heat transfer coefficient can then be applied to more general situations. As an aside, if the temperature difference dT and dT' (the difference between the controlled temperature and the inlet temperature) is the considered to be the same, then the heat transfer coefficient can be found in a different way. This is true initially:

$$h = \frac{q_1}{A_s * dT} = \frac{\dot{m} * c_p * dT}{A_s * dT}$$

$$h = \frac{\dot{m} * c_p}{A_s}$$

The result is equation 3.4. The heat transfer required of each iteration (the controller output heat transfer) in the VI is then defined as q2:

$$q_2 = A_s * h * d\acute{T}$$

where dT' is:

$$d\acute{T} = (\acute{T_3} - T_2)$$

This controller output heat transfer is considered to be dynamic, unlike the required heat transfer, which is constant given a constant set point. Here the controller output heat transfer can be considered to be the required amount of power to overcome a difference between the currently controlled temperature and the set point. The reason this dynamic heat transfer is included as an equation is out of necessity; the VI works only by via previous values of iteration. Without both heat transfers defined, the VI was shown to record a drastically incorrect PID output power.

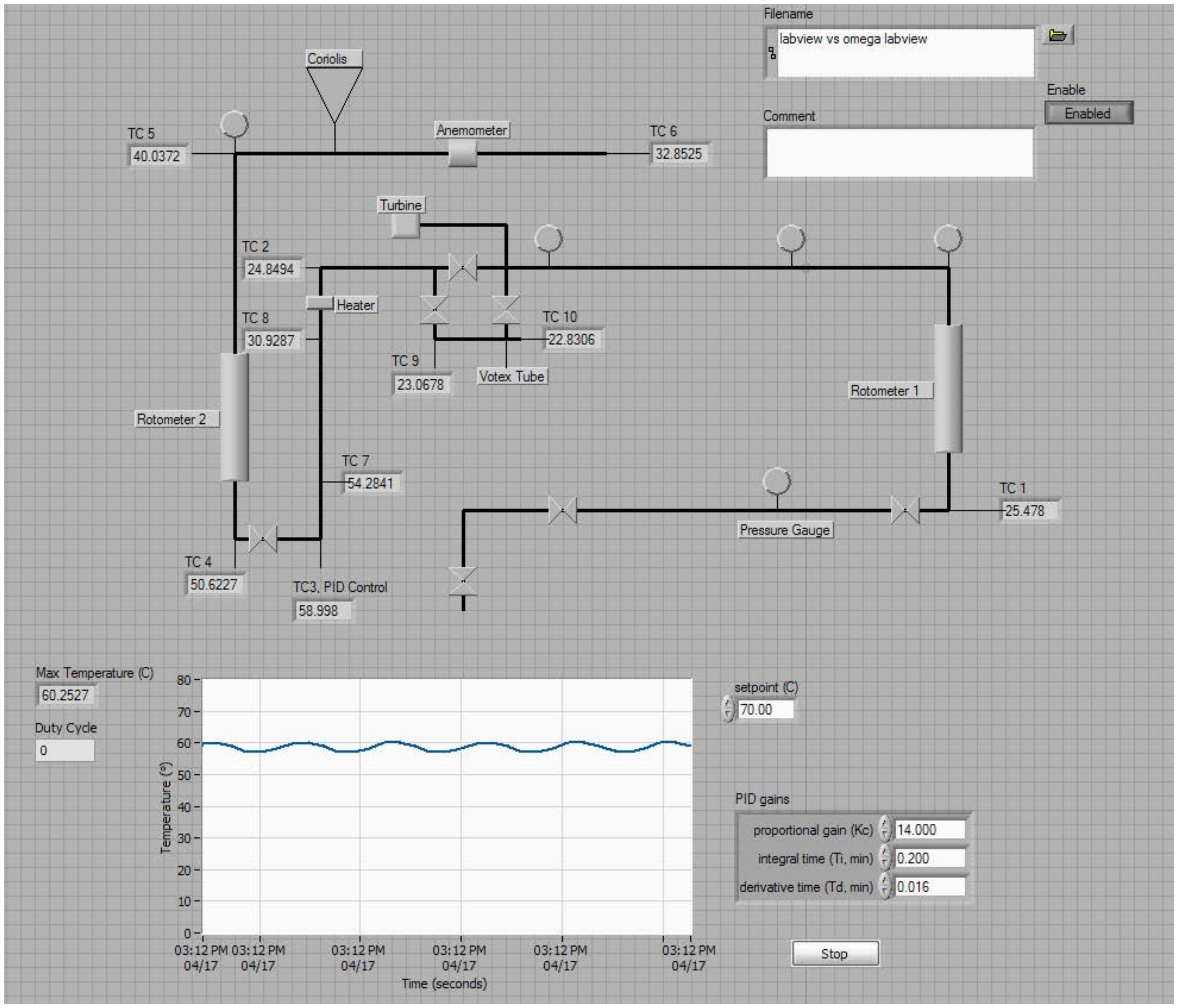
Since the PID output is recorded, the controlled temperature can then be solved for:

$$T_3 = \frac{q_2 - q_1}{A_s * h} + T_2$$

The result is equation 3.6. Since q_1, A_s, h, and T_2 are known at any given iteration, T_3 can be solved if the value of q_2 is recorded by LabVIEW.

6.4.2 *Physical*

Front Panel



Filename: labview vs omega labview

Enable: Enabled

Comment:

TC 5: 40.0372

TC 6: 32.8525

TC 2: 24.8494

TC 8: 30.9287

TC 10: 22.8306

TC 9: 23.0678

TC 7: 54.2841

TC 1: 25.478

TC 4: 50.6227

TC3, PID Control: 58.998

Max Temperature (C): 60.2527

Duty Cycle: 0

setpoint (C): 70.00

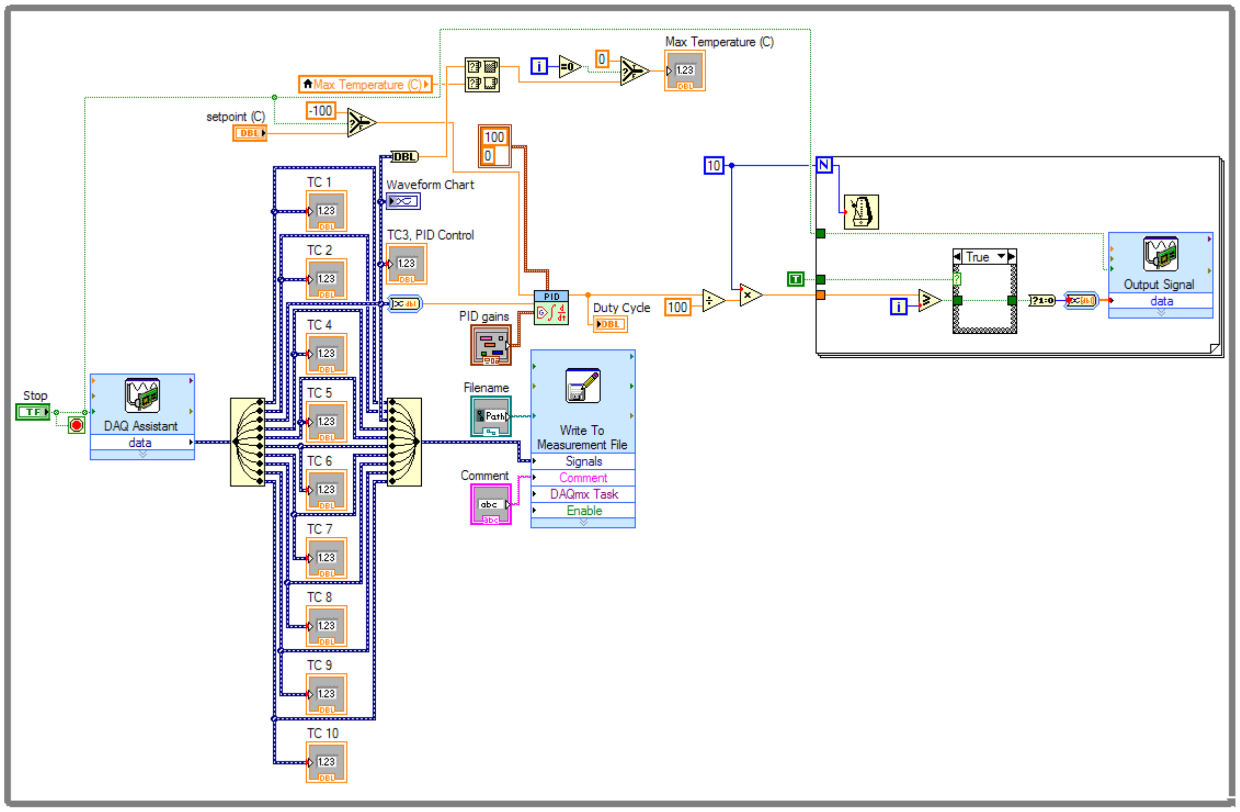
PID gains

proportional gain (Kc): 14.000

integral time (Ti, min): 0.200

derivative time (Td, min): 0.016

Back Panel


Max Temperature (C)
Max Temperature (C)
setpoint (C)
-100
0
100
0
TC 1
TC 2
TC 4
TC 5
TC 6
TC 7
TC 8
TC 9
TC 10
Waveform Chart
TC3, PID Control
PID gains
Duty Cycle
100
10
N
True
Output Signal
data
Stop
DAQ Assistant
data
Filename
Comment
Write To Measurement File
Signals
Comment
DAQmx Task
Enable

6.5 Vortex Tube

6.5.1 *Energy Balance Derived*

$$Q_A = Q_R \sqrt{\frac{P_A T_S}{P_S T_A}}$$

$$\dot{m} = Q_A * \left(\frac{.3048\, m}{ft}\right)^3 * \rho_{air}$$

$$\dot{m}_i = \dot{m}_c + \dot{m}_h$$

$$0 = \Delta H_c + \Delta H_h$$

$$\Delta H = m \int c_p \, dT$$

$$0 = c_p[\dot{m}_c(T_c - T_i) + \dot{m}_h(T_h - T_i)]$$

$$0 = c_p[\dot{m}_c(T_c - T_i) + (\dot{m}_i - \dot{m}_c)(T_h - T_i)]$$

6.5.2 *Entropy Equation Derived*

$$\frac{dS}{dt} > 0$$

$$\frac{dS}{dt} = \frac{\dot{Q}}{T} + \dot{S}$$

$$\dot{S} > 0$$

$$\dot{Q} = \Delta\dot{H} = mc\Delta T$$

$$\Delta\dot{H} = c_p[(m\Delta T)_h + (m\Delta T)_c]$$

$$\frac{dS}{dt} = c_p[\frac{(m\Delta T)_h}{T_h} + \frac{(m\Delta T)_c}{T_c}]$$

6.6 Raw Data

6.6.1 Omega Tuning

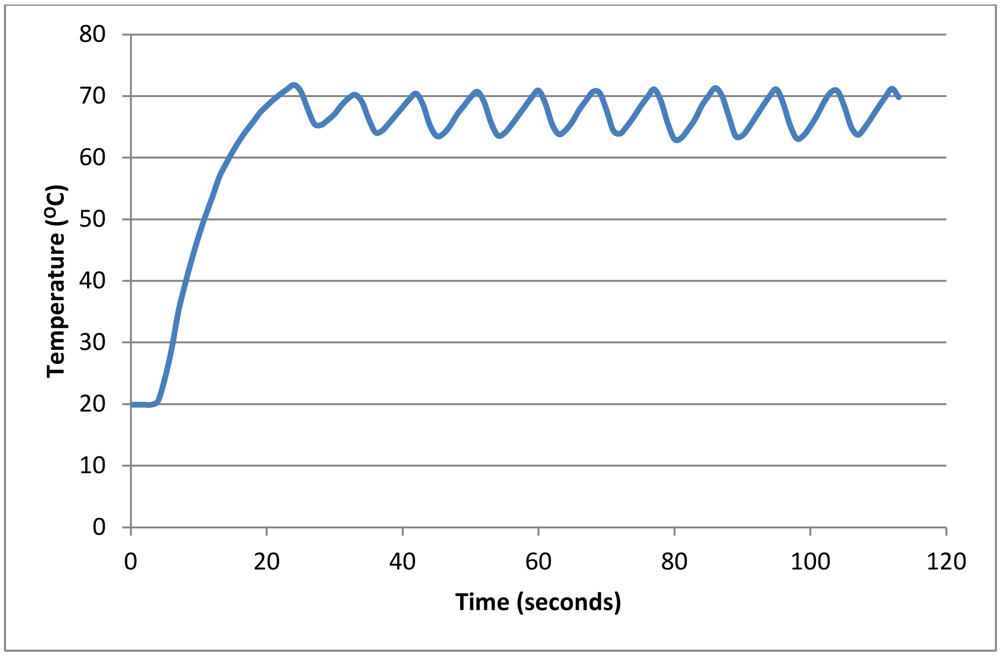


FIGURE 41: P-BAND 0.5%, INTEGRAL TIME 12S, DERIVATIVE TIME 1S

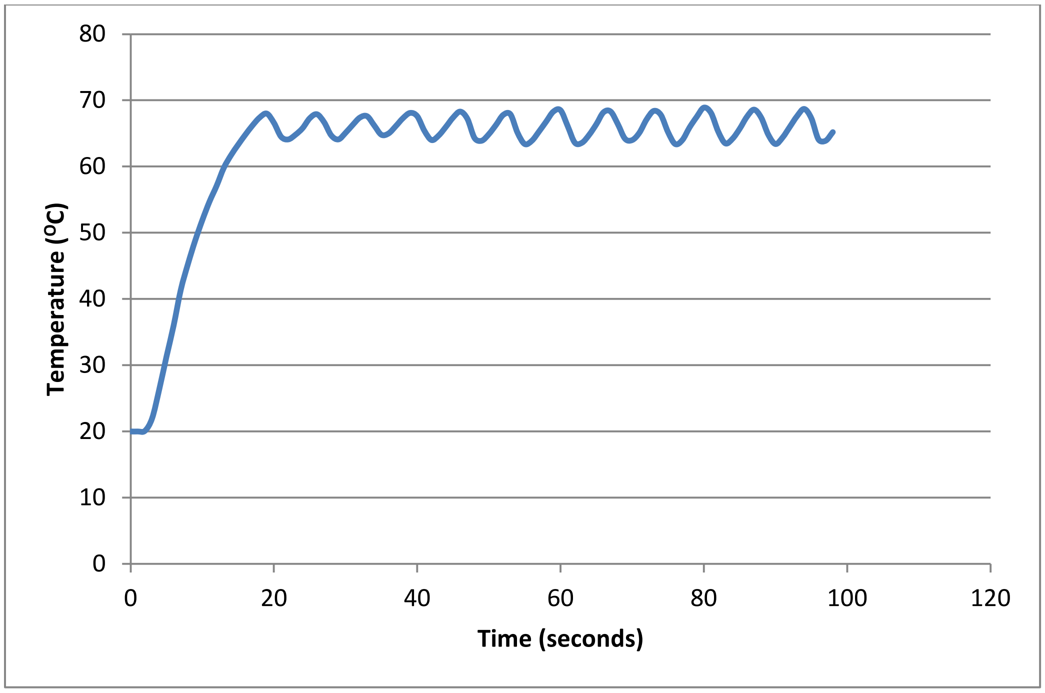


FIGURE 42: P-BAND 7%, INTEGRAL TIME 12S, DERIVATIVE TIME 1S

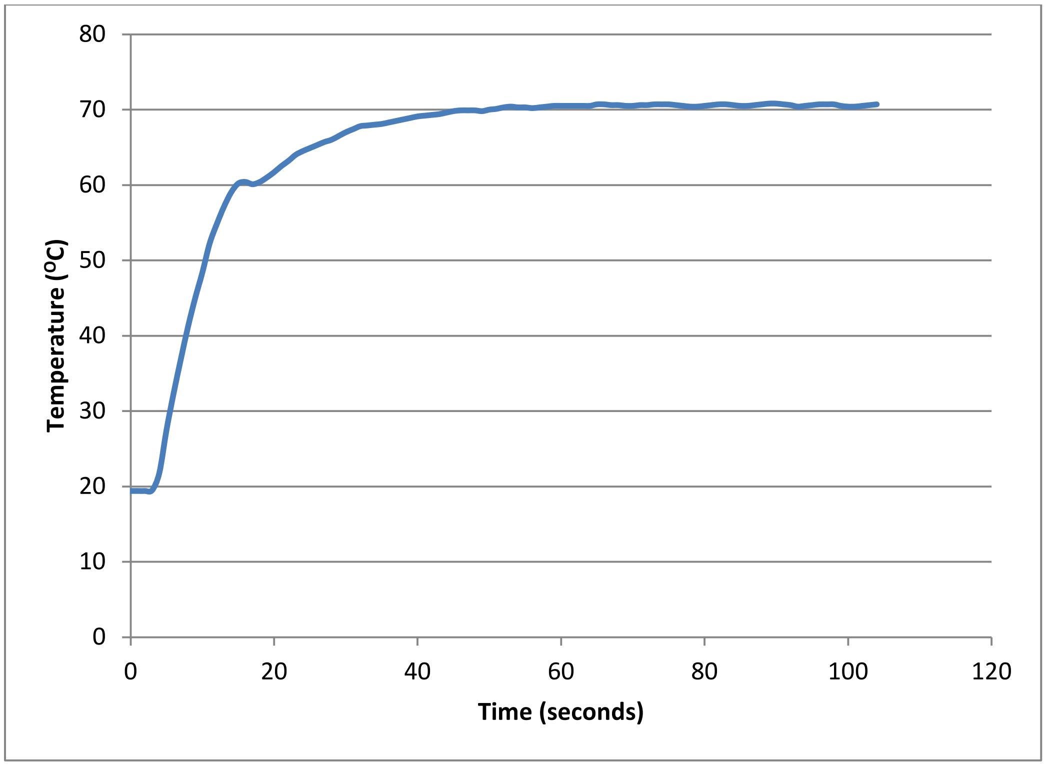


FIGURE 43: P-BAND 20%, INTEGRAL TIME 12S, DERIVATIVE TIME 1S

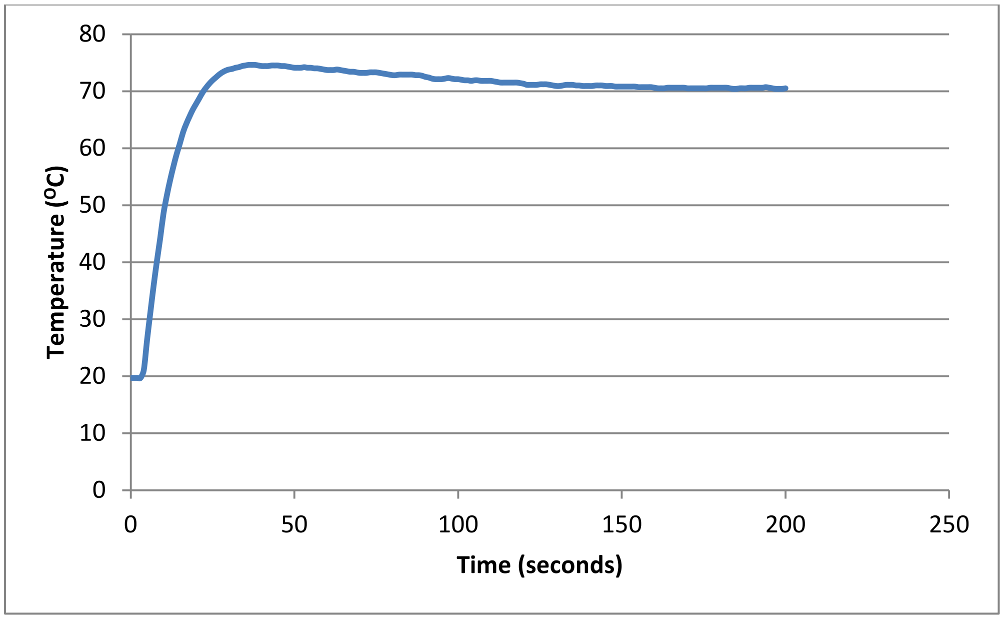


FIGURE 44: P-BAND 100%, INTEGRAL TIME 12S, DERIVATIVE TIME 1S

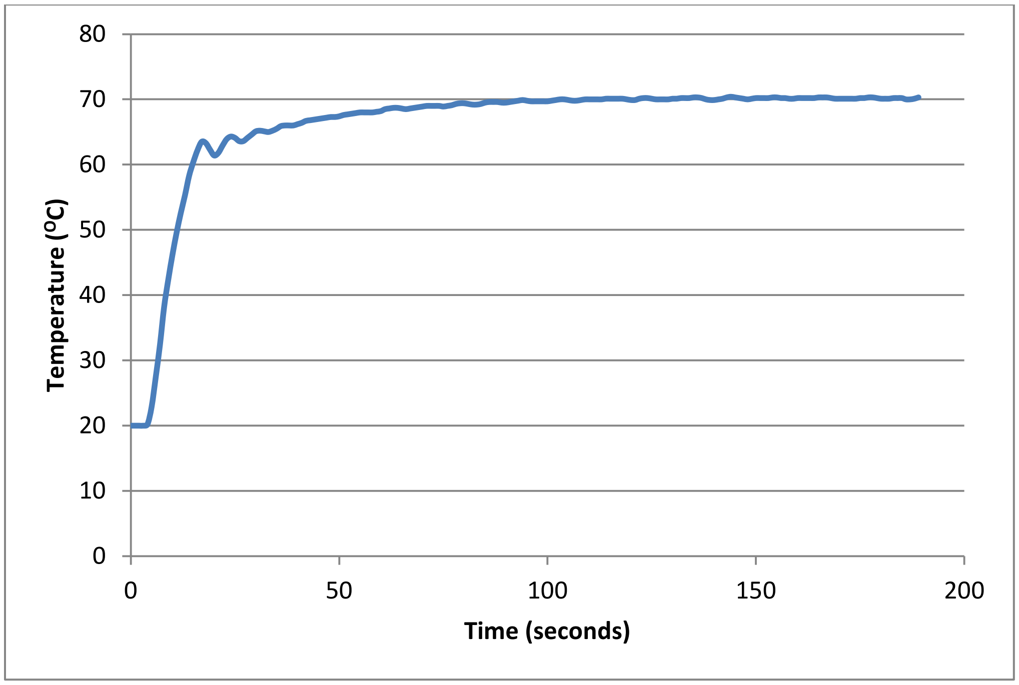


FIGURE 45: P-BAND 14%, INTEGRAL TIME 30S, DERIVATIVE TIME 1S

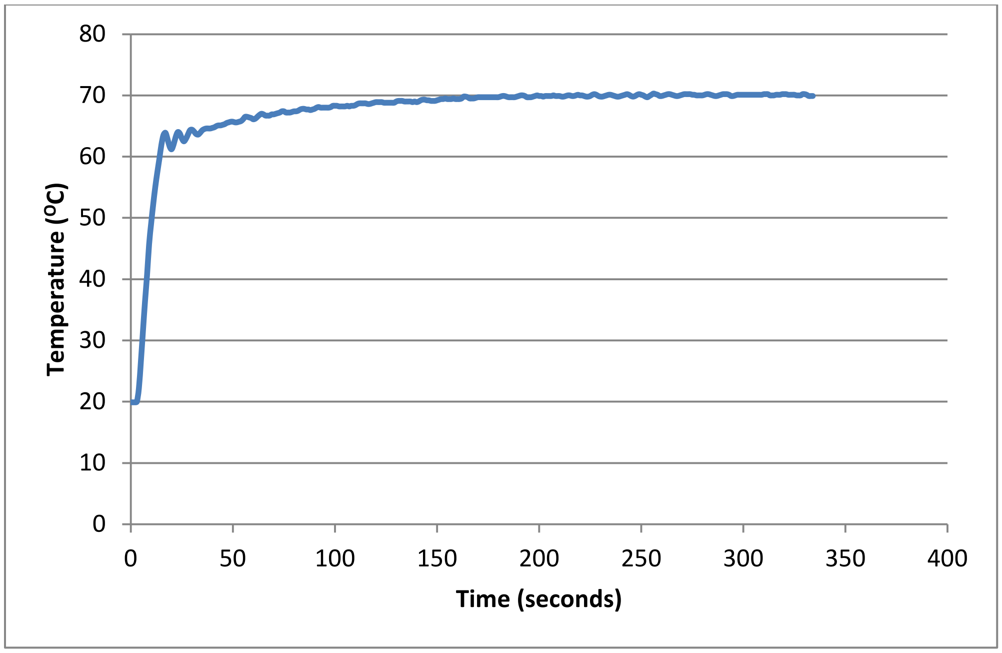


FIGURE 46: P-BAND 14%, INTEGRAL TIME 60S, DERIVATIVE TIME 1S

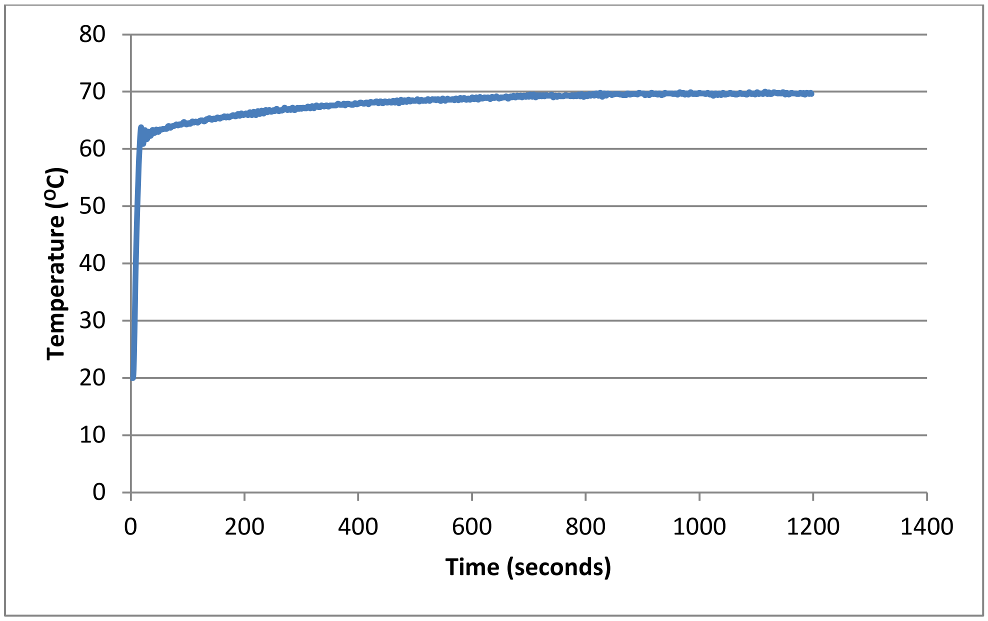


FIGURE 47: P-BAND 14%, INTEGRAL TIME 300S, DERIVATIVE TIME 1S

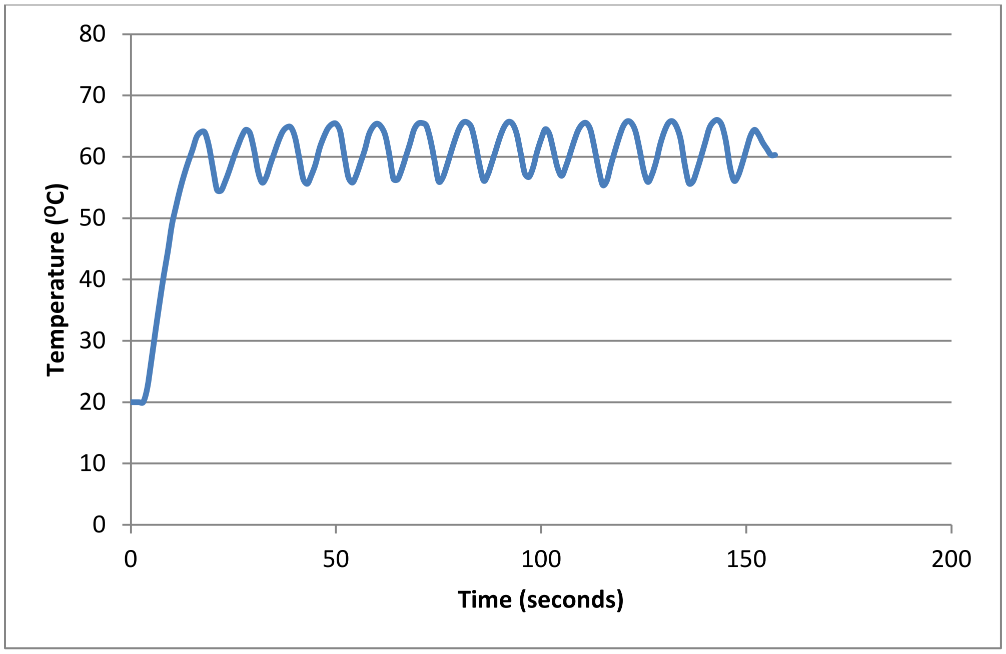


FIGURE 48: P-BAND 14%, INTEGRAL TIME 12S, DERIVATIVE TIME 7S

6.6.2 *Vortex Tube*

Trial	$Q_{R,i}$	$P_{a,i}$	T_i (^OC)	T_c (^OC)	T_h (^OC)	$Q_{R,c}$	T_4 (^OC)	T_5 (^OC)
1	0.3	65.7	21	-6	45.8	0.33	15.2	18.6
2	0.305	65.7	21	-6	45.4	0.325	18.75	18.3
3	0.29	59.7	20.2	-5.7	41.8	0.28	19	19.6
3a	0.29	59.7	20.4	-5.8	41.9	0.28	17.4	19.3
3b	0.29	59.7	20.4	-5.9	42	0.28	16.1	19
3c	0.29	59.7	20.4	-5.8	42	0.28	15.4	18.6
3d	0.29	59.7	20.4	-5.7	42	0.28	13.9	18.1
3e	0.29	59.7	20.4	-5.6	42.1	0.28	13.2	17.9
3f	0.29	59.7	20.4	-5.6	42.1	0.28	12.2	17.4
3g	0.29	59.7	20.4	-5.8	42.2	0.28	11.5	17
3h	0.29	59.7	20.4	-6	42.2	0.28	10.8	16.5
3i	0.29	59.7	20.4	-6.2	42.2	0.28	10.4	16.2
3j	0.29	59.7	20.4	-6.3	42.2	0.28	9.9	15.8
3k	0.29	59.7	20.4	-6.2	42.2	0.28	9.6	15.4
3l	0.29	59.7	20.4	-6.1	42.2	0.28	9.3	15.2
3m	0.29	59.7	20.4	-5.9	42.1	0.28	9	14.9
3n	0.29	59.7	20.4	-5.8	42.1	0.275	8.9	14.7
3o	0.29	59.7	20.4	-5.6	41.9	0.275	8.7	14.4
3p	0.29	59.7	20.4	-5.4	41.8	0.275	8.5	14.2
3q	0.29	59.7	20.4	-5.4	41.5	0.275	8.5	14.1
3r	0.29	59.7	20.4	-5.6	41.2	0.27	8.4	14
3s	0.29	59.7	20.4	-5.8	41.5	0.27	8.4	13.8
3t	0.29	59.7	20.4	-6	41.3	0.27	8.3	13.7
3u	0.29	59.7	20.4	-6	41	0.27	8.3	13.6
3v	0.29	59.7	20.4	-5.8	40.7	0.27	8.4	13.6
3w	0.29	59.7	20.4	-5.3	39.8	0.265	8.4	13.5
3x	0.29	59.7	20.4	-5.5	40.9	0.265	8.4	13.5
3y	0.29	59.7	20.4	-5.1	40.3	0.26	8.4	13.5

Trial	$Q_{R, i}$	$P_{a, i}$	T_i (^{O}C)	T_c (^{O}C)	T_h (^{O}C)	$Q_{R, c}$	T_4 (^{O}C)	T_5 (^{O}C)
3z	0.29	59.7	20.4	-4.6	39	0.26	8.5	13.5
3aa	0.29	59.7	20.4	-4.4	38.8	0.26	8.6	13.5
3bb	0.29	59.7	20.4	-4.5	38.6	0.26	8.7	13.5
3cc	0.29	59.7	20.4	-4.4	38	0.255	8.9	13.6
3dd	0.29	59.7	20.4	-4.5	37.7	0.255	9.1	13.6
3ee	0.29	59.7	20.4	-4.5	37.5	0.255	9.2	13.6
3ff	0.29	59.7	20.4	-4.4	37.3	0.25	9.3	13.7
3gg	0.29	59.7	20.4	-4.2	37.2	0.25	9.4	13.7
3hh	0.29	59.7	20.4	-4	36.9	0.245	9.4	13.8
3ii	0.29	59.7	20.4	-3.7	36.6	0.25	9.5	13.8
3jj	0.29	59.7	20.4	-3.7	36.9	0.245	9.5	13.8
3kk	0.29	59.7	20.4	-3.3	36.5	0.24	9.5	13.9
3ll	0.29	59.7	20.4	-3.1	35.3	0.23	9.6	14
3mm	0.29	59.7	20.4	-2.9	33.5	0.22	9.8	14
3nn	0.29	59.7	20.4	-2.9	30.9	0.2	10	14.2
3oo	0.29	59.7	20.4	-3.1	30.2	0.21	10.2	14.4
3pp	0.29	59.7	20.4	-2.9	30.1	0.195	10.4	14.4
3qq	0.29	59.7	20.4	-2.6	29	0.19	10.7	14.6
3rr	0.29	59.7	20.4	-2.6	28.8	0.19	10.9	14.7
3ss	0.29	59.7	20.4	-2.2	28.5	0.18	11.1	14.8
3tt	0.29	59.7	20.4	-2.2	28.5	0.19	11.3	13.9
3uu	0.29	59.7	20.4	-1.8	29.1	0.18	11.8	13
3vv	0.29	59.7	20.4	-2.1	28.9	0.19	11.7	15.1

6.7 REFERENCES

Abdulwahed, M., Nagy, Z.K., & Blanchard, R.. (2009). *Online laboratories embracing the reusable learning objects metaphor.* Twentieth Annual Conference of The Australian Association for Engineering Education. Adelaide, Australia.

Acroname Robotics. (2012). *Description of pulse-width modulation.* Acroname Robotics. Retrieved from http://www.acroname.com/robotics/info/concepts/pwm.html

AiRTX International. (2007). *Catalogue #29*. 3620 Wiehe Road, Cincinnati, OH, 45327: AiRTX International. Retrieved 10/24, 2011 from http://www.axxosales.com/Newsdocs/AiRTX_Catalog_29.pdf

AiRTX International (n.d.). *AiRTX stainless steel vortex tubes*. 6320 Wiehe Road, Cincinnati, OH, 45327: AiRTX International. Retrieved 10/24, 2011 from http://site.wtech-services.com/clients/wtechservices/Downloads/HeatingCooling612200924135PM2.pdf

Ang, K.H., Chong, G.C.Y., & Li, Y. (2005). PID control system analysis, design, and technology." *IEEE Transactions on Control Systems Technology, 13(4)*. 559-576.

Astrom, K.J, & Hagglund, T. (2004). Revisiting the Ziegler–Nichols step response method for PID control. *Journal of Process Control, 14.* 635-650. Retrieved on Mar. 12, 2012, from http:www.sciencedirect.com.

Astrom, K.J, & Hagglund, T. (2001). The future of PID control. *Control Engineering Practice, 9(11)*, 1163-1175. Retrieved on Mar. 13, 2012, from http:www.sciencedirect.com

Bequette, B.W. (2003). *Process control: modeling, design, and simulation.* Upper Saddle River, NJ: Prentice Hall.

Bequette, B.W., Aufderheide, B., Prasad, V., & Puerta, F. (2000). *A process control experiment designed for a studio course.* Unpublished manuscript. Retrieved 10/1, 2011 from http://128.113.2.9/dept/chem-eng/WWW/faculty/bequette/ramesh/publications/CEE.pdf

Bucknell University. (2012). *An Introduction to Signals*. Bucknell University [web]. Retrieved 11 Apr. 2012, from http://www.facstaff.bucknell.edu/mastascu/elessonshtml/Signal/Signal2.htm

Carnegie Mellon & University of Michigan. (1997). *Control tutorials for matlab - PID tutorial.* Retrieved 10/28, 2011, from http://www.engin.umich.edu/group/ctm/PID/PID.html#controller

Cengel, Y. & Cimbala, J. (2010) *Fluid mechanics – fundamentals and applications, 2nd ed*. New York, NY: McGraw-Hill, inc. 930.

Cooper, D.J. (2008). *Integral (Reset) Windup, Jacketing Logic and the Velocity PI Form*. Retrieved 20 Mar. 2012, from http://www.controlguru.com/2008/021008.html

Desborough, L. & Miller, R. (2002). Increasing customer value of industrial control performance monitoring—Honeywell's experience. *Proceedings from the Sixth International Conference on Chemical Process Control: AIChE Symposium Series 98(326)*.

Gao, C. M., Bosschaart, K. J., Zeegers, C. H., & de Waele, A. T. A. M. (2005). Experimental study on a simple ranque-hilsch vortex tube. *Cryogenics, 45*(1), 173-183.

Hilsch, R. (1947). The use of the expansion of gasses in a centrifugal field as cooling process. *The Review of Scientific Instruments, 18(2)*, 108-113.

Johnstone, A.H., & Al-Shuaili, A.. (2001). Learning in the laboratory; some thoughts from the literature. *University Chemistry Education, (5).*

Love, J. (2007). *Process Automation Handbook: A guide to Theory and Practice*. London, England: Springer-Verlag London.

Massa, John. (2005). *Pulse Width Modulation (PWM) Tutorial*. Datadog Systems [web]. Retrieved 10 Apr. 2012, from http://www.datadog.com/pwm_tutorial.pdf

Ming-da, M., & Xin-jian, Z. (2006). A simple auto-tuner in frequency domain. *Computers and Chemical Engineering, 30(4).* 581-586. Retrieved on Mar. 13, 2012, from http:www.sciencedirect.com

Motion Systems Design Staff. (2000, Oct.). MS-101: Pulse-width modulation. *Motion Systems Design.* Penton Publications, Cleveland, Ohio. Retrieved 5 Apr. 2012, from http://fab.cba.mit.edu/classes/MIT/961.04/topics/pwm.pdf

Muske, K. R.. (2003). "Simulation and Experiment in an Introductory Process Control Laboratory Experience*" Chemical Engineering Education, 37(4),* 306-315.

National Instruments. (2012). *NI LabVIEW control design and simulation module for windows* [web]. http://sine.ni.com/nips/cds/view/p/lang/en/nid/209850

National Instruments. (2010). *LabVIEW 2010 help* [web]. http://zone.ni.com/reference/en-XX/help/371361G-01/

O'Dwyer, A. (2006). *Handbook of PI and PID controller tuning rules, 2nd ed.* Covent Garden, London, UK: Imperial College Press.

OMEGA ENGINEERING, Inc (n.d., a). *CN9000A series autotune temperature controller* – (User Manual). Retrieved from http://www.omega.com/manuals/manualpdf/m1191.pdf

OMEGA ENGINEERING, Inc (n.d., b). *Temperature control: Tuning a PID (three mode) controller* (User Manual No. ERG05). Online: Omega.

Osei-Prempeh, G., & Silverstein, D. L.. (2010) "Making a Chemical Process Control Course an Inductive and Deductive Learning Experience," *Chemical Engineering Education, 44(2),* 119-126.

Parallax Inc. (2006). *Process control - student guide* Parallax Inc. Retrieved from http://www.parallax.com/dl/docs/prod/sic/Web-PC-v1.0.pdf

Scientific Instruments. (2007). *Electrical heating (PWM).* Scientific Instruments. Retrieved from http://www.imagesco.com/articles/nitinol/07.html

Seal, A.M. (1998). *Practical process control.* . Elsevier. Online version available at: http://www.knovel.com/web/portal/browse/display?_EXT_KNOVEL_DISPLAY_bookid=3442&VerticalID=0

Seborg, D.E., Edgar, T.F., Mellichamp, D.A., & Doyle, F.J. (2010). *Process dynamics and control.* Hoboken, NJ: John Wiley and Sons.

Technological Education Institute of Chalkis. (2001). *How pulse width modulation works: what is PWM and why is it useful?.* Technological Education Institute of Chalkis. Retrieved 6 Apr. 2012, from http://www.ee.teihal.gr/labs/electronics/web/downloads/What_is_PWM_and_why_is_it_useful.pdf

Wescott, T. (2000). PID without a PhD. *Embedded Systems Programming,* 86-108.

University of Michigan. (2009). *12. chemical process PID control - chem 466.* Wolf, P. (Director). (2009).[Video/DVD] Youtube: University of Michigan.

Visioli, A. (2006). *Pratical PID Control.* London, England: Springer-Verlag London.

Ziegler, J.G., & Nichols, N.B. (1942). Optimum settings for automatic controllers. *Transactions of the American Society of Mechanical Engineers (ASME)*. 759-768.

Zhong, J. (2006). *PID controller tuning: A short tutorial* (Presentation). West Lafayette, Indiana: Purdue University.