The Use of Music to Teach Chemical Engineering Concepts

An Interactive Qualifying Project completed in partial fulfillment of the Bachelor of Science degree at Worcester Polytechnic Institute, Worcester, MA

Submitted By:

Bryan Mendes

Brian Hecker

Yudi Wang

Yo Karita

Submitted To:

Professor Michael Timko

Professor Vincent Manzo

Abstract

Chemical engineering and the concepts that comprise this field involve many visual aspects including graphs, charts, and diagrams. This makes visual learners the ideal for succeeding in the chemical engineering field, therefore, those who learn a different way, for example, auditory learners, are at a distinct disadvantage. The purpose of this project was to advance a previous IQP, *Sonification of Chemical Engineering Data,* by providing an easier to use interface of their idea, namely, creating a way to represent and teach the chemical engineering concept of multiple steady states (MSS) through sound. Our project uses a combination of auditory and visual representations of the workings of a continuously stirred tank reactor (CSTR) and allows the user to modify a set of values through the use of PID controls. The software was converted from a combination of Simulink and Python to a single software, Unity, in order to increase the ease of use for the participants using the program. The program was tested by a variety of WPI students, many of whom were not in the chemical engineering field. This was done in an attempt to objectively receive data showing whether or not problems involving MSS could be solved through sound, even by those with little to no prior knowledge on the subject. The students were given a VARK test as a pre-survey to determine their preferred method of learning. Then after reading a short briefing on PID controls and MSS, they were asked to complete four problems using the visual and auditory components presented by our program on Unity. Finally, the students were presented with a post survey where they record their VARK scores, their answers to the four problems, and answers to a few other questions concerning the program itself. The data suggests that there is a negative relationship between VARK auditory score, and whether or not the student found the auditory portion of the program to be helpful. Furthermore, the data collected shows that most students found the program simple to use.

Acknowledgements

Our group would like to thank Professors Manzo and Timko for their continual aid and feedback over the duration of this IQP. We would also like to thank those who tested the program and provided us with feedback. Without the combined efforts of those mentioned above, this project could not have been completed.

Authorship

This IQP report was written by Bryan Mendes, Brian Hecker, Yudi Wang, and Yo Karita.

All of these individuals are responsible for the contents of each section of this report.

Table of Contents

Executive Summary

Title of project: The Use of Music to Teach Chemical Engineering Concepts

Authors: Bryan Mendes, Brian Hecker, Yudi Wang, Yo Karita

Advisors: Professor Vincent Manzo, Professor Michael Timko

Abstract

Chemical engineering and the concepts that comprise this field involve many visual aspects including graphs, charts, and diagrams. This makes visual learners the ideal for succeeding in the chemical engineering field, therefore, those who learn a different way, for example, auditory learners, are at a distinct disadvantage. The purpose of this project was to advance a previous IQP, *Sonification of Chemical Engineering Data,* by providing an easier to use interface of their idea, namely, creating a way to represent and teach the chemical engineering concept of multiple steady states (MSS) through sound. Our project uses a combination of auditory and visual representations of the workings of a continuously stirred tank reactor (CSTR) and allows the user to modify a set of values through the use of PID controls. The software was converted from a combination of Simulink and Python to a single software, Unity, in order to increase the ease of use for the participants using the program. The program was tested by a variety of WPI students, many of whom were not in the chemical engineering field. This was done in an attempt to objectively receive data showing whether or not problems involving MSS could be solved through sound, even by those with little to no prior knowledge on the subject. The students were given a VARK test as a pre-survey to determine their preferred method of learning. Then after reading a short briefing on PID controls and MSS, they were asked to complete four problems using the visual and auditory components presented by our program on Unity. Finally, the students were presented with a post survey where they record their VARK scores, their answers to the four problems, and answers to a few other questions concerning the program itself. The data suggests that there is a negative relationship between VARK auditory score, and whether or not the student found the auditory portion of the program to be helpful. Furthermore, the data collected shows that most students found the program simple to use.

Project Goals

The goal of this project was to create a simple, easy to use program which would function as a means to properly teach the concept of Multiple Steady States through the use of PID control. The project is to be a continuation of the work in *Sonification of Chemical Engineering Data* wherein we address the comments from the test group that voiced complaints over the complexity of their program, which was utilized using MATLAB, Max, and Python. Another goal of this project was to improve upon the musical aspect of the sonification of the chemical engineering data, thus allowing for a more conclusive result as to whether or not those who learn best from auditory cues can be adequately taught chemical engineering through sound and music. Our thought process, and how we went about achieving these goals, is presented in the paper below.

1 - Introduction

Worcester Polytechnic Institute is known for its advanced engineering background including its chemical engineering major. Through these engineering programs, an advanced comprehension of concepts and theories are provided and a hand on experience is provided in a laboratory setting. These mental and applicational skills are culminated into creating some of the world's leading engineers. Through four years of schooling, the most cutting edge science is combined with the newest math forms in order to create a person who can tackle some of the world's leading problems. Although this curriculum boasts the creation of some of the best engineers worldwide, the program is not entirely perfect. Large class sizes restrict the ways that teachers can teach their subject, so the primary forms of data presentation are usually of an auditory or visual variety, as most engineers tend to learn this way. This lack of diversity in teaching method leads to many students who learn in an auditory manner to struggle in the classroom and may potentially hinder their innate ability an engineer's, or even worse, damage their love of the subject.

Previously, an IQP by the name of "Sonification of Chemical Engineering Data" was created and put into motion to correct this lack of auditory learning. The project consisted of taking a complex chemical engineering concept, multiple steady states, and creating a way for auditory learners to be able to more easily learn the subject. In a chemical engineering class, more often than not, visual and auditory methods such as graphs and lectures are used to portray a certain subject. Unfortunately, this neglects those that learn most adeptly through auditory means, so this previous IQP used a program called MATLAB in order to create a way for these learners to thrive in the

classroom. Through survey questions, and a live test of their program, the previous group worked to determine a correlation between those who claim to be auditory learners and whether or not they found the live test helpful in retention of knowledge.

Based on this previous IQP's data, a new IQP was created in order to make the interface more user friendly. Many of the participants from "Sonification of Chemical Engineering Data" claimed that the MATLAB program was difficult to use, and that some of the tones present in the program were unpleasant if exposed to them for too long. This project works off of the previous project's program and aims to make an easy to use interface that is both fun and immersive for the user.

2 - Background

2.1 - Chemical Engineering Concepts

As engineering becomes a more prominent field in the world, the need for increased amounts of engineering colleges becomes apparent. One field in particular that has grown throughout the years is that of chemical engineering, with schools such as WPI and MIT having very strong programs. Chemical engineering involves large amounts of visualization and incorporates the language of chemistry and the building of schematics and devices into a single major. This being the case, many of the classes offered focus on a visual perspective of teaching. For example, being able to “see” the molecules you are working with makes understanding the chemistry much easier, and those students that can do this tend to have an advantage and higher success. This is yet another setback for those who learn in an auditory manner, as there is currently no applicable way to teach many chemical engineering subjects using sound. The sonification of temperatures and control parameters in a reactor as performed by the previous IQP is a directly applicable way to use sound to teach a difficult subject in chemical engineering. This project, being an extension of that one, also includes the subject of multiple steady states in a reactor.

2.1.1 - Multiple Steady States (MSS)

The reactor that is normally used to teach multiple steady states (MSS) in the classroom is known as a continuous stirred-tank reactor or CSTR. This type of reactor

assumes perfect or ideal mixing and due to it being performed at a ‘steady state’, the temperature, reaction rate, and the concentration are all independent of the variable of time. When the reactor is in a steady state with the qualities stated above, it can be assumed that the energy within the reactor remains constant. If this occurs, then the temperature within the reactor remains constant. The overall goal of a steady state problem is to find the point of energy balance of the system, which the system will be driven to. At this point, the energy entering, the net energy of the reaction, and the energy escaping either through cooling or leaving the system will be in equilibrium. This point varies as different conditions of the system are changed. Normally this is a trivial question for a chemical engineering student to answer, but the challenge of MSS is that there are more than one set of conditions that can cause the energy of the system to balance. Depending on the initial state of the reactor in question, the process can be driven to one of these multipole unique energy points.

The graph of a multiple steady state with these qualities contains two ‘lines’. The graph of the heat generated due to the chemical reaction resembles that of a logistic function, with two horizontal asymptotes, one upper bound and one lower bound, while the heat removed from the cooling jacket present is a straight line that crosses the other graph at most three times. Usually, there are three stable points to which the system can converge, an upper temperature, a lower temperature, and an unstable middle temperature. Regulation of the system can be used to achieve performance at the middle point, but any slight increase in temperature will cause the system to push towards the upper steady point, while any decrease in temperature from the middle point will cause a push towards the lower steady point. In this way, the system will not

converge to the middle point on its own; any temperature besides the precise middle point will converge either to the upper point or lower point.

2.1.2 - PID Control

Proportional-Integral-Derivative(PID) control is a control loop feedback system that is commonly used in many applications. PID control is used to manipulate a system to converge to a desired condition. In PID control, the control system measures the current state of the overall system and calculates the error from the desired condition using the formula stated below.

$$E(t) = C_{desired} - C_{output}$$

In a PID controlled system, the error is time dependent, meaning that the difference between the desired condition and the current condition will change over time. The control system then attempts to minimize this error by adjusting a control variable through the use of a differential equation. The error is used in three separate terms; the proportional term, the integral term, and the derivative term.

$$Q = Q_0[k_P E(t) + k_D \int_0^t E(t)dt + k_I \frac{dE(t)}{dt}]$$

$K_P E(t)$ is the proportional term. This term multiplies the current instantaneous error by K_P and has more influence on the output than the integral or derivative terms. $k_I \int_0^t E(t)dt$ is the integral term. By integrating the error from the start time until the current time, this term incorporates all the past values of the error into the output. $k_D \frac{dE(t)}{dt}$ is the derivative term. It finds the instantaneous slope of the error at the current time and multiplies it by K_D. This allows the equation to predict future values of the error and potentially dampen oscillations that are common with this type of control system.

K_P, K_I, and K_D are parameters for the PID control equation that are set by the designer of the control system. These parameters are optimized to control the system property with minimal overshoot, offset, and oscillations, which are problems that result from an improperly controlled system. There are several methods for optimizing these parameters, the most common of which are the *Ziegler-Nichols Tuning Method,* and the *Riggs Tuning Method.*

PID control can be to incorporated into the continuous stirred-tank reactor introduced in the Multiple Steady State section by using it to control the temperature of the cooling jacket. By changing the temperature of the cooling jacket, it is possible to control the intersection points of the heat gained and the heat removed functions. Using the middle steady state temperature as the target, PID control will try to minimize the difference between the current temperature and the middle steady state. This will cause the system to converge to the unstable steady state condition despite that it would normally never do that.

2.2 - The basics of Auditory Learning

The study of learning types, their distinctions and their effects on human behavior and overall mentality became a subject of interest among psychologists and sociologists in the 1970s. These researchers hoped to prove that not all minds are inherently alike and in doing so, promote different methods of teaching and communication that would appeal to a more diverse range of people who may differ in learning preference (Grinder & Bandler, 1976). Most materials provided in colleges are for visual learners, especially in science, technology, engineering and math (STEM) fields.

Conversations between professors and students and group discussions take an important role during academic studies. Also more and more professors record their lectures and let students just watch videos and take online courses. All these above are powerful examples of auditory learning, which is a learning style depends on sounds including speaking and hearing. Common methods for those auditory learners are reading a textbook out loud, asking questions in a lecture and listening to audio books. Those actions give them strong stimulus and learn contents more effectively than just reading a textbook and solving practice problems.

2.2.1 - Research and History of Auditory Learning

The first concept to gain widespread notoriety was neuro-linguistic programming (NLP), which suggested that a person's predisposed approach to communication (and by extension, their primary modality of learning) has a profound effect on their personal development. This concept claims that if a person approaches a new subject that is represented through only one modality to which they are not prone (i.e. if an auditory learner is only provided with the visual means to learn – mismatching representational systems) then they are significantly more likely to be diagnosed with a learning disorder, experience anxiety and depression, or suffer from a lack of confidence in their learning abilities (Dowlen, 1996). Observing historical context, it is now understood that the NLP approach would not have gained such notoriety had it not been introduced against the backdrop of the emerging self-esteem movement of the 1960s and 1970s. The proposed links to an increase in mental disorders (extending to problems such as allergies 14 and myopia) were unfounded, causing NLP to be discredited and labeled a pseudoscience (Bradley & Biedermann, 1985). However, the concept of different

representational systems and their characteristics spurred a larger interest in learning modalities, which psychologists sought to further research and define. One highly regarded model to expand upon NLP representational systems is Neil Fleming's VARK model, which has been commended as having rigidly defined learning styles and recognizing common traits among visual learners, auditory learners, reading writing preference learners, and kinesthetic (tactile) learners (Fleming, 2014). Though several different models have been designed, the intended application of most seems to revolve around the "meshing hypothesis," which suggests that instruction is most effective when provided in the format that matches the learner's preference. For example, the hypothesis suggests the most effective instruction for a visual learner would be a visual representation of the information being provided (Pashler, McDaniel, Rohrer & Bjork, 2008). However, in the 1980s, Grinder and Bandler rejected their previously proposed meshing hypothesis, stating that people are not restricted to one modality of learning or thinking, but rather use all senses when receiving and interpreting instruction, concluding that the most effective instruction for most people is given in a way that appeals to multiple senses. Therefore, auditory learning methods should be mixed with visual and reading-writing methods to promote the most profound level of understanding. One or more of these methods are often left out of an instructor's curriculum either for brevity or to alleviate some difficulties of instructing large groups at once (Bradley & Biedermann, 1985).

2.2.2 - Auditory Learning and Education

Auditory learning is a preference of learning tool through listening to spoken lessons and instructions, dialogue, and to a lesser extent mnemonics and rhythms.

People may to classified as auditory learners if they find the aforementioned learning methods more effective than visual or tactile methods. While college students who are recognized as auditory learners typically benefit substantially from actively listening in lectures and discussing lecture material with professors, teaching assistants and peers, lessons are not often formulated in a way that optimizes auditory learning.

Most lectures have auditory components such as instructors explaining a particular subject and displaying a narrated video, but it is increasingly less common for college courses to integrate class discussions with lectures, particularly in STEM fields. Barring in-class discussions, STEM courses rarely implement novel tools for auditory learners such as mnemonic devices or sonically represented data (such as audio files or interactive software), instead favoring graphs, tables and calculations. The vast majority of concepts in the STEM field, both in the classroom and in industry, are taught using methods that favor visual and reading/writing learning. This led educational learning researchers David Kolb and Ron Fry to hypothesize that auditory learners must adapt to less-intuitive learning styles to succeed in STEM fields, and that many auditory learners avoid STEM fields altogether due to their difficulty in learning the subject matter through the available models (Kolb, 1984).

2.3 - Gamification

In order to further the previous project of the “Sonification of Chemical Engineering Data” and to make the interface more user friendly for the target audience, it was necessary to ‘gamify’ the project. In order to do this, the previous IQP stated

above will be adapted to the gaming engine Unity. The type of gaming format that this project will take involves creating what is known as a ‘serious game’.

2.3.1 - Serious Games

A ‘serious game’ involves engaging the user in active problem solving through real-world situations. These games can be entertaining but their main purpose is the education factor as well as other purposes such as advertising as evident through the game America’s Army 3, which was used as a recruitment tool in 2009. Another good example of a serious game is the popular game Civilization V, which takes the player through the history of the human race as they use a particular race in an attempt to reach an “end game” whether it be world domination or destruction (to name a couple). The educational value of running through real-life scenarios such as war, rebellion, and growing a nation in a proper manner is immense. One of the most important aspect of serious games is that they don’t focus on one particular fact in an isolated manner. Instead, these games use whole experiences and scenarios to display the full ‘idea’ behind a particular subject, thus allowing the player to learn the interconnections of the idea in question rather than learning through memorization. There are a few key reasons why serious games act as a better learning experience than conventional teaching methods.

One such reason is the adaptability of the game itself. By creating different scenarios spanning a wide range of difficulties, the game ensures that each player is adequately challenged but not overly burdened by the information. As trends within the game are seen by the creators, they can change their game as they see fit to make a

better learning experience. This adaptability is not as prevalent in conventional teaching methods due to time constraints of a particular class, class size, or the overall teaching style of the professor. The game can also be used to repeat instructions and proceed at a pace unique to each individual, so students who retain information at a slower rate will not be left behind as compared to in the classroom.

Another reason using serious games is the game's ability to provide immersive and interesting scenarios. These scenarios can range from a fantasy environment, to performing mechanical maintenance on a satellite in space. There is essentially no ceiling to the types of scenarios that can be created. This leads to the last major benefit of serious games; the easy to create problems that the users must solve. One of the main ways for a student to learn especially in a technical school, is by problem solving. By implementing many different problem solving scenarios pertaining to the subject, the student can achieve a better grasp on just how far reaching the particular subject is. They may find connections to problems that they had never considered before, and this in turn not only helps them to learn the material, but raises their creativity as well. By creating engaging scenarios, the students perceive the learning less as work, and more as a fun experience with the added bonus of becoming better at their craft. In order to create a successful serious game, it is important to be able to create a situation in which the user forgets that they are learning classroom material and instead are completely immersed in the game. In order to understand how to create this situation a different type of game was researched; the ‘immersive game’.

2.3.2 - Immersive Games

An ‘immersive game’ is a game that provides challenge, control, and fantasy for the player as well as putting them into a sense of ‘flow’. Flow is defined as a state of optimal experience where the player is so engaged that self-consciousness disappears, sense of time is lost, and the task is done not for external rewards, but for the exhilaration of doing so. This can be achieved through a multitude of different ways, including creative environments, in depth or relatable characters, and an interesting plot that progresses well. Understanding the basics of an immersive game can make creating a better serious game much easier.

The main aspects of gameplay that captures the interest of a player can be split into a few key ideas. The most important of these ideas is to present a well defined goal. With a goal presented to the player, there is incentive to play other than just learning or furthering the plot. Once the goal has been set, the next objective of the game designer is provide concise feedback into how close the goal is to being achieved. Presenting checkpoints and intermediate mini-goals make the final goal seem attainable even if it takes a long time. With small rewards thrown in at the intermediate points, it keeps the player interested instead of just plugging away for hours trying to complete one objective. This is present in adventure rpg games in the form of side quests that make leveling up characters less tedious. This type of ‘small goal- reward’ system is also present in major serious games such as Foldit, where users manipulate chemical structures in order to make a final structure with best conformation and energy. Foldit provides tutorials to learn the finer points of the game as well as providing encouraging text and a progress bar to know how close a player is to completing a level. Finally, with the goals and rewards firmly laid out, the final major aspect of the

game should be to constantly adjust games challenges to be slightly above what the player is at currently to provide adequate difficulty. The levels and individual problems should not be so easy as to promote boredom but should also not be so difficult as to provide frustration to the player. A successful immersive-serious game ensures that the player is constantly in flow with as little disturbance to this ideal as possible.

2.3.3 - Game Engines

A game engine is a package of software that takes care of many of the basic elements of modern video games. Game engines often handle game physics, as well as graphical user interfaces and game input. A game engine makes it possible for the game creator to focus on the uniqueness of a game and not creating the interfacing between the software and the different types of hardware it might be running on. Game engines can be used to create software other than games as well. Any program that is designed to have a graphical interface or has to interface with input devices could benefit from a game engine. Some modern game engines also have a development environment for working with that specific engine.

When creating a piece of software, there are several game engines that could be considered. The two most popular engines used by independent developers are Unreal Engine and Unity. While both engines are wonderful game engines, each has its strengths and weaknesses. Unreal engine is able to create very complex games with high quality graphics. It is mostly used for making high quality 3D games. However, its capabilities also make it very complex to work with.

The Unity game engine is also a modern game engine used to create fairly complex games. Unity is often used in the mobile game market, as is it very flexible. Unity has built in cross-platform portability, which makes it great for reaching a wider market of gamers. Unity also has greater capabilities for creating 2D games and interfaces than Unreal Engine. Unreal Engine however has greater 3D capabilities. These two game engines also differ in the programming language that is used. Unreal Engine uses the C++ language. C++ is an object-oriented language that is derived from the C language. It is a well established language, though it is somewhat complicated to use. Unity can either be used with the C# language or the Javascript language. C# is viewed as the more stable language to use with Unity. C# is another object oriented language that is derived from C. However, it is a more modern language than C++ and is easier to learn and use. C# can be used as either a compiled language like C or as a scripting language. This makes C# a flexible and powerful language. For these reasons, Unity was the clear choice of game engine for our project.

3 - Methodology

3.1 – Introduction

Our team set out to improve upon the results of a previous project that studied the effectiveness of auditory teaching techniques. To do this, we condensed the assignment used by the previous group into a single piece of software. We also attempted to make the software more interesting and game like, in order to capture the user's attention. To compare our software with the results of the previous project, we designed an assignment similar to that of the previous project.

3.2 – Preliminary Survey

The designers of the previous project used a questionnaire created by VARK to ascertain preliminary information about the learning styles of their users. This survey uses answers to certain questions to give a person’s relative preferences toward visual, auditory, read/write, and kinesthetic learning. The results of this survey are given as a number for each style, where a higher number corresponds with a greater preference towards that style. For this project, this survey had several important uses. As in the previous project, the VARK survey allowed us to determine the learning tendencies of our pool of test subjects. This was used to determine whether or not the project appeals to the target audience. The preliminary survey also allowed for comparisons between to testing pool of this project and that of the previous project, which allowed us to more accurately compare the results of the two projects.

3.3 - Choosing a Game Engine

In order to simplify the use of the previous project, our first step was to choose the optimal environment and software package to use. It quickly became clear that game engines supported the audio, visual, and computing requirements needed to complete all segments of the project. Once game engines had been discussed as a general topic, a specific game engine needed to be chosen. While many game engines exist, the most popular and versatile are Unity Game Engine and Unreal Game Engine. After exploring both of these game engines, it was decided that Unity Game Engine was the best environment to develop our project. Unity's simplicity and excellent documentation make it good for developers, such as our team, who have never worked in a game engine before. Unity also has easy cross-platform compatibility, which makes it ideal for future expansion upon this project. Overall, it was clear that Unity Game Engine was the best platform with which to develop this project.

3.4 - Modeling the Chemical Engineering Problem in Unity

The main goal for this project was to condense the Simulink, python, and MAX/msp/jitter components of the previous project into a single simple to use program. For this, it was decided that the Unity game engine and development environment would be the most practical platform. The process of modeling this problem in Unity can be broken into four components; modeling the continuously stirred tank reactor, modeling PID control, and sonifying the data.

3.4.1 – Modeling the Continuously Stirred Tank Reactor

The first part of the system to be modeled was the continuously stirred tank reactor. This system can be broken into two parts; the heat gained by the reaction, the heat lost through the cooling jacket, and the overall change in temperature. Each of these parts can be modeled through the use of a mathematical equation.

Based on the previous project's Simulink model, the heat gained by the reaction was calculated by the following equation for a first-order exothermic reaction:

$$G(T) = \frac{-\Delta H_{rxn} \tau A e^{-E/RT}}{1 + \tau A e^{-E/RT}}$$

In this equation, the heat gained is a function of the temperature of the CSTR and a set of defined constants. Within the C-Sharp code of this project each of these constants is defined to the values shown in table one. This equation exhibits the desired behavior shown in Figure 1.

Constant	**Value**
Delta H	57739.2
Tau	0.06
A	4000000000
E	56902.4
R	8.31434

Table 1: Heat Gained Constant Values

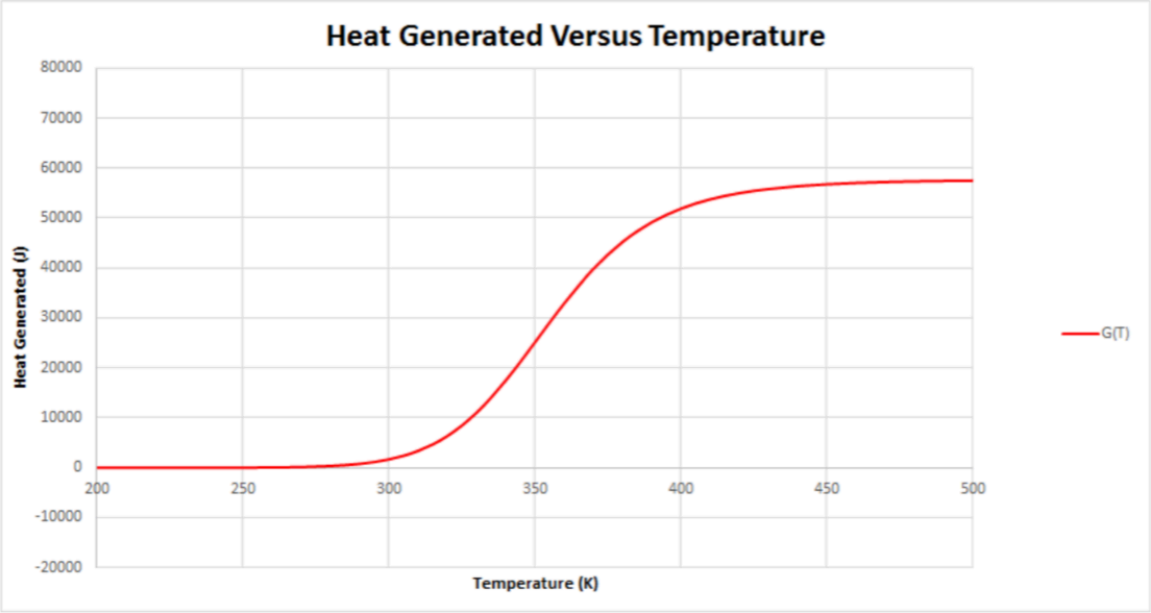

Figure 1: Heat Gained vs. Temperature

To implement this behavior within the C-Sharp project, heat gained is calculated based on the temperature using the update() function which is called in each frame. The temperature of the CSTR is initialized to 370K at the start up of the program. During each frame, the equation

shown above is used to calculate the heat gained. This value is then used later to update the state of the CSTR model.

The second part of this model is the heat lost through the cooling jacket. The cooling jacket removes heat from the CSTR so that the heat generated from the reaction does not continuously increase the temperature of the CSTR. The amount of heat removed was calculated by the following equation from the previous project's Simulink model.

$$R(T) = C_{P0}(1+k)(T - T_C)$$

Where: $k = \frac{UA}{C_{P0}F_{A0}}$ and $T_C = \frac{T_0 + kT_\alpha}{1+k}$

This equation is dependent on the values of several constants, as well as the value of the CSTR temperature T, and the value of the cooling jacket temperature T_α. The values of the constants are shown in table 2.

Constant	Value
UA	3400
C_{P0}	300
F_{A0}	30
k	0.378
T_0	288.15

Table 2: Constant Values for Heat Removed

If the cooling jacket temperature is held constant the equation is linear with respect to the temperature of the CSTR. When combined with the behavior of the heat gained from the reaction, this gives us the multiple steady states at any point where the heat gained is equal to

the heat removed. This is shown in Figure 2.

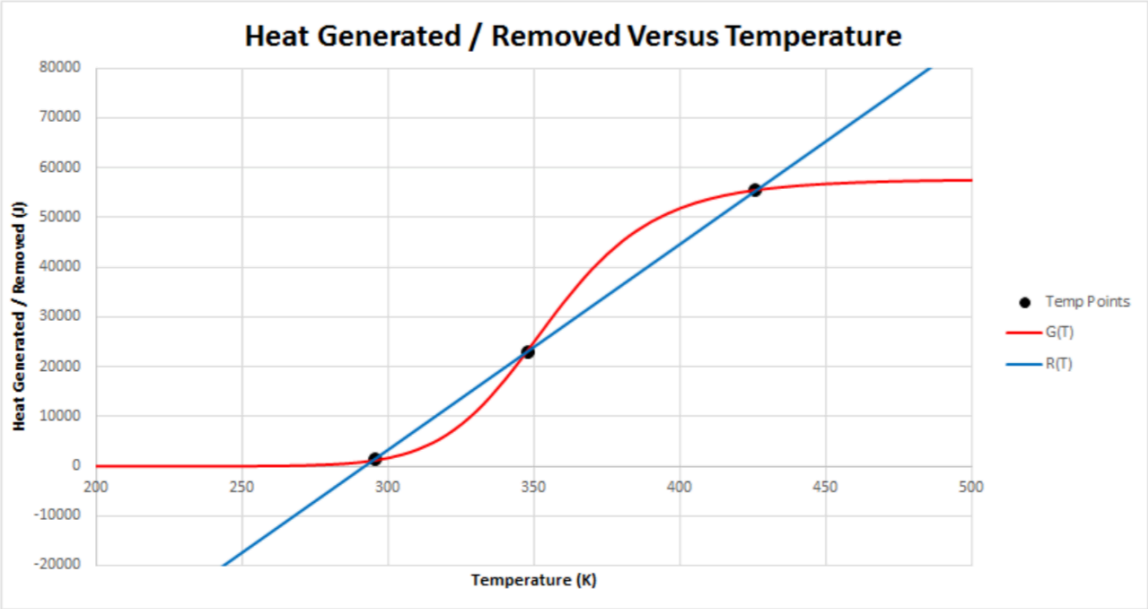

Figure 2: Heat Removed as compared to Heat Generated

Changing the value of the cooling jacket temperature has the effect of shifting the linear graph of the heat removed up or down depending on how much the temperature has changed. This allows for the steady state values to be changed. This behavior is shown in Figure 3.

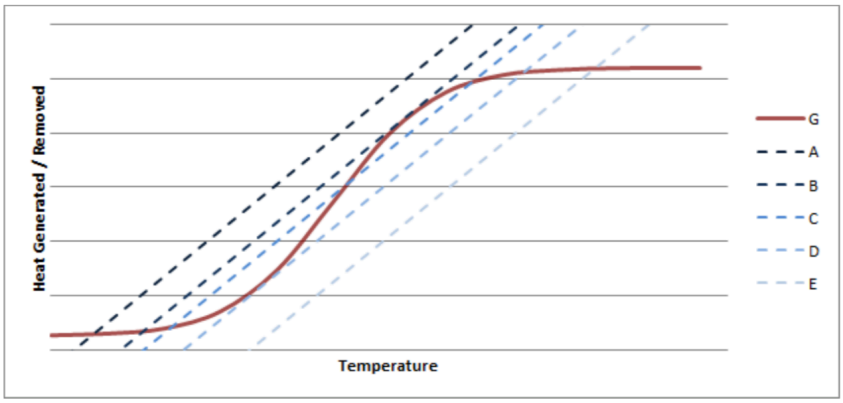

Figure 3: Behavior of Changing Cooling Jacket Temperature

The heat removed is modeled in the C-sharp project in a similar way to the heat gained. The jacket temperature is initialized to 288.15K when the program starts. During each frame, the

value of the heat removed is calculated using the temperature dependent equation for heat removed. This value is then used to update the state of the CSTR system.

The final step in modeling the CSTR system is to update the temperature based on the heat gained and the heat removed. The heat gained by the reaction and the heat removed by the cooling jacket have a direct effect on the overall temperature of the system. This effect is controlled by the basic thermodynamic equation:

$$\Delta H = m * C * \Delta T$$

Where ΔH is the change in heat, m is the mass, C is the specific heat capacity, and ΔT is the change in temperature. In the instance, the equation needs to be reversed in order to solve for change in temperature. Values for the mass and specific heat were chosen based on values used for the previous project. The value for mass was chosen to be 30, and the value for specific heat was chosen to be 300.

At the end of each update cycle, the change in temperature would be calculated by using this equation. The overall change in heat would be calculated by subtracting the heat removed by the cooling jacket from the heat gained by the reaction. If the heat removed was greater than the heat gained, this value would be negative and therefore the change in temperature would also be negative. The calculated change in temperature would then be added to the current temperature of the CSTR to update the state of the system for the next update cycle. In the call to update, the cooling jacket temperature is also updated based on the PID control method addressed in the next section. The final code for modeling the CSTR is shown in appendix C.

3.4.2 – Modeling PID Control

As in the previous project, a PID control system was used to converge the system to the middle, unstable, steady state condition. The PID control system uses the equation:

$$Q = K_P E(t) + K_I \int_0^t E(t)dt + K_D \frac{dE(t)}{dt}$$

to minimize the error value between the current input and a target value. In the equation, K_P, K_I, and K_D are user specified coefficients and E(t) is the function representing the error value at each moment in time. To calculate this error, a target value is first needed. The target value for this CSTR system is the middle steady state condition. This steady state condition is defined as the temperature at which the heat generated by the reaction is equal to the heat removed by the cooling jacket. Using the equations from previous sections:

$$G(T) = \frac{-\Delta H_{rxn}\tau A e^{-E/RT}}{1 + \tau A e^{-E/RT}} = C_{P0}(1+k)(T - T_C) = R(T)$$

However, this equation cannot be used because this condition is also true at the high and low steady states. This equation can be used to solve for a set of middle steady state points. With these points, it is possible to use linear regression to solve for an equation, which will approximate the middle steady state temperature based on the cooling jacket temperature. This equation is based on the cooling jacket temperature because it is the jacket temperature which controls the locations in which the heat generated and heat removed functions intersect. The equation used for this project is shown below.

$$T_{target} = 466.1 - 0.394 * T_{Jacket}$$

Once the target temperature is defined, the PID control system subtracts the current temperature from the target temperature to get the current error value. As this is a discrete time system and error values only exist for each update cycle, the integral of the error is replaced with a sum of all of the errors before it and the derivative term is replaced with the difference between the current and previous error values. This program stores the previous error value and the sum of all previous error values as global variables.

The final part of the PID control system is the user input K values. These values control how much effect each term of the PID control equation has. It is often desired that the K input

values be tuned to reduce the time in which it takes for the error to reach zero. In this project, three input fields in the User Interface are used to edit the K values. These input fields will accept only numeric floating-point values, and each calls a specific function when the user is finished editing it. The function for each of these input fields serves the same function for each of the specific k values. When the function is called, the function updates a global variable that signals to the rest of the program that that specific k value exists, It then parses through the string that was given from the input field and converts it into a float to store it as a global. The final thing done by this function is to reset the CSTR and cooling jacket temperature values to their initial conditions. This means that whenever the K values are changed, the program restarts and shows the full error behavior.

The PID control function is used to control the temperature of the cooling jacket. During each call of the update method, the PID control function is called and the output is used as a heat value to change the temperature of the cooling jacket. Because the output of the PID function is the heat to be added or subtracted from the cooling jacket, to calculate the temperature change, the equation $\Delta T = \frac{\Delta H}{m*C}$ must be used again. Since the value T_c used in the heat removed equation is dependent on the value of the jacket temperature, it also needs to be recalculated.

3.4.3 – Sonifying the Data

The final part of the project to be implemented in Unity was to sonify the data produced by the continuously stirred tank reactor system. In the previous project, this was done through the use of the Max visual programming language.

In that project the error data was represented by the sounds output by the max program in a specific pattern. As the error increased in the positive direction, a trumpet would play higher and higher pitches. The trumpet playing higher pitches corresponded with a greater positive error

value. As the error increased in the negative direction, the pitch of the trumpet would decrease to lower notes. When the error value is equal to zero, the trumpet plays a middle C note. In addition to the pitch of the trumpet representing the error value, a set of other instruments would play a short song. At large error values in either the positive or negative direction, the song would be inaudible. As the error began to approach zero, the volume and number of instruments would increase until the entirety of the song would play at full volume when the error value was equal to zero. In this way, the project accurately represented the data through sound in an attempt to help auditory learners understand the concepts being taught.

In this project, the Unity program attempts to sonify the error data in a similar way. The Unity program uses a trumpet playing different pitches to represent the current error value. As the error value increases in the positive direction, the trumpet will play higher notes and as the error increases in the negative direction, the trumpet will play lower notes. The scale in this project, however, is centered on the G-note above middle C. The set of instruments playing the short song is also implemented in a slightly different fashion. In this project, instead of being introduced gradually, the set of instruments are introduced at the same time as the error value approaches zero. This allows the user to clearly know when the error value has reached zero. When the music from the set of instruments come in, instead of playing the central note, the trumpet stops playing. This is implemented in this way due to difficulties with the Unity audio support and API. For any future work, it is suggested that this be changed.

To implement the audio portion of this project, several instances of the Unity Audio Source object were used. Each audio source was controlled by its own script and played the sound for a different instrument. The most complicated audio logic was the audio source for the trumpet. The logic for the trumpet was implemented using a series of if statements and a set of audio samples for each different trumpet pitch. During each call to the update() function within the trumpet control script, the series of if statements figures out what range the error value is

currently in. Once an if statement is found to be true, the playoneshot() function is called in order to play the corresponding pitch from the trumpet.

The other audio sources that control the other instruments are controlled by a very simple script. The scripts for each of the other instruments share the same simple code. On the startup of the program, all of the instruments start playing with the volume of the audio source set to 0. This synchronizes the all the instruments so the song is played correctly. Each time the update method is called, it checks whether or not all of the K-values for the PID control have been initialized. This makes sure that the music does not play before the user wants to start the program or when the user has stopped the program. If the K-values have been initialize, it will then check if the error value to see if it is in a small range close to zero. If the error value is, the volume of each audio source will be increased allowing all of the instruments to play. If the error value is not in that range, the volume will once again be set to zero.

3.5 – Creating the Assignment

To complete this project, an assignment was created in order to get user feedback on the software that was created. The assignment was designed to gage how useful the software was and to compare the results of this project to those of the previous one. Another goal of this assignment was also to determine what improvements should be made during the next iteration of the project. In the previous project, the pool of testers was chosen from a chemical engineering class to ensure the candidates had some background and understanding of the concepts. Because our project focused on the usefulness and simplicity of the software create, it was decided that our testing pool would stray away from this, and testing candidates were chosen at random. In order to compare the testing pools from the two projects, each candidate was asked to fill out the survey by VARK that documents the candidate's preferred learning styles.

The assignment begins by having the candidates read a passage of background information on both multiple steady states and PID control. Once this was completed the candidates were then set a series of tasks to complete using the software. These tasks are designed to have the candidate explore their understanding of PID control as well as explore the limits of the software. A copy of the assignment is shown in appendix A.

Once each candidate had completed the assignment they were asked to complete a survey using Google forms. This survey asks them to record their VARK survey outcomes, and the outcomes of the assignment tasks. It then asks a series of simple questions about their experience using the software. This survey is designed to help the team understand the usefulness and limitations of the software. It is also designed to help show how this project could be expanded upon and what future work can be done. The survey and the candidate's responses are shown in appendix B.

4 - Results and Discussion

This project was conducted to determine if PID controls and MSS could be taught properly and efficiently through auditory means. In addition to this, the project was created to develop a more user friendly program with which to teach the students in question, as compared to the multi pronged program used in *Sonification of Chemical Engineering Data*. With the voluntary input of 8 students, the system was tested through the use of a pre survey, four questions involving PID controls and MSS, and then a post survey. Our group plotted whether or not the student found the auditory cues to be helpful versus their auditory score on the VARK pre survey. This data is shown in the graph below.

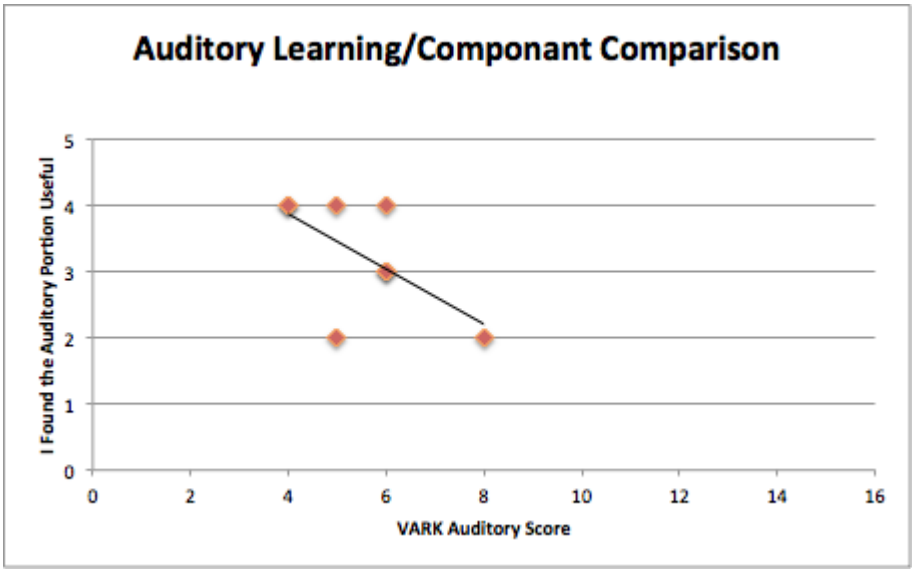

Figure 4: this figure shows the relationship between VARK auditory scores (x axis) and whether or not the student found the auditory portion of the program helpful in solving the questions (y axis). Both points (4,4) and (6,3) are representative by two students.

The figure above above is indicative of a negative correlation between the students VARK auditory score, and whether or not they found the auditory portion of the

program helpful. Although this is the opposite of what the project was intended to do, this could be due to a multitude of factors including too small of a testing size. A larger testing size could yield a more reliable correlations, because although the graph shows a negative correlations, the majority of participants found the auditory portion to be helpful compared to those who didn't. Furthermore, the reliability of the data could increase with the introduction of a more stable environment in which to do the assignment; mainly, a place with little to no external noise. The discrepancy in values is shown in the graph below which depicts the student versus how helpful they thought the auditory portion of the assignment was.

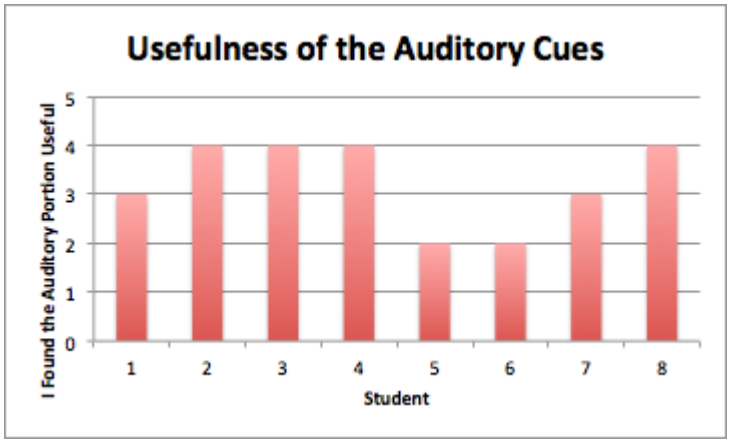

Figure 5: This graph represents how helpful each student thought the visual cues in the program were when solving the assignment.

Along with testing to see if the auditory element was useful to the student, one of our main focuses was in creating a program that was easy to use. To this end, we proceeded with creating the system solely on Unity, unlike our predecessors, who used MATLAB, Max, and Python. By creating a single, fluid system, we hoped to remedy the issues concerning ease of use that were brought up in *Sonification of Chemical*

Engineering Data. The graph below depicts what each of the 8 volunteers thought about our programs ease of use.

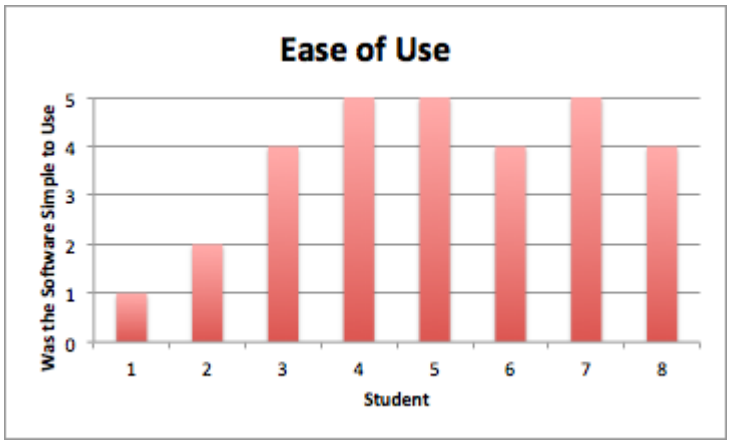

Figure 6: This graph depicts the student's response in the post survey to the question, “Was the software simple to use?” The scale ranged from 1 (confusing), to 5 (simple).

The graph reveals that many of the students gave high marks to the simplicity of the program. Although two students rated the program below average in terms of simplicity, 75% rated above the average of three. This could be due to the tasks being limited to a single program like we anticipated, or due to the overall simplicity of the assignment given. Although the lack of student volunteers could make the data less reliable, if this trend were to hold for a group of twenty or more, then a more affirmative decision on the program’s overall efficiency could be made.

Finally, we wanted to determine if the program accurately depicted how PID controls worked compared to whether or not the student had any prior knowledge concerning this type of controls. In the post survey, we provided two questions to help quantify this, *“Did you have an understanding of PID control before the assignment?”* and *“This software helped me understand PID control”*. The graph below depicts the responses to these two questions.

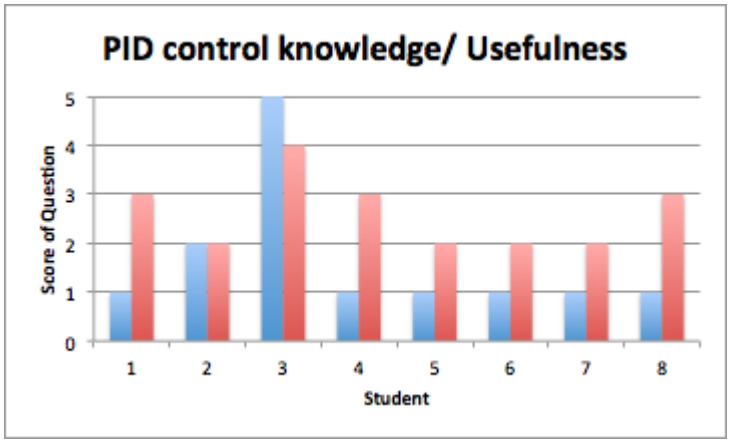

Figure 7. This graph compares the students' responses to the two questions *"Did you have an understanding of PID control before the assignment?"* (blue lines), and *"This software helped me understand PID control"* (red lines). In this case the value 1 meant no understanding/no and a 5 represented full understanding/yes.

The data in the graph above indicates that most people do not have a knowledge of PID control before testing this program. Even though this is the case, most people tended to have either a neutral or a slightly negative response as to whether or not the software helped them to understand PID controls. This could be due to a few extraneous factors such as a small survey size or the primary focus of the assignment being the auditory component

5 - Conclusions and Future Work

Based on the survey results, we were able to conclude that our software was helpful to students who have interests on auditory learning. Most of students who completed the survey were not auditory learners, but after using our software, everyone thought they had better understanding on PID control. We also learnt that, most students preferred visual learning, read/write learning and kinesthetic learning to auditory learning based on their VARK preliminary scores and there stated personal preferences in the survey. As a conclusion, extending auditory learning materials into more courses to present it to more students is very important. We strongly recommend presenting auditory learning as an additional option to students.

Despite the successful results of our project, the software has a lot to be improved in the future to enhance utility and the user experience. There are some students thought the instructions were very confusing, the others also thought the instructions were not clearly enough. In order to attract more students to complete the assignment and to use the software, the instructions should be simpler so that they are easy to understand to everyone. In addition, some students thought the software was hard to use, especially those who had no understanding with PID control before. So the software should be easier to implement.

Our project mostly just set the groundwork for condensing the project to Unity software, which is user-friendlier and easier. In order to make the program more interesting and useful, the future project can focus on updating the user interface to make it look nicer and easier to read, and update the auditory portion to make that work better for auditory learners. Furthermore, there are still some bugs and problems within our project to be

fixed. There is potential to make expansion on what we have done in this project in the future, both in terms of types and scope of sonification and the various engineering disciplines that can benefit from auditory learning styles.

References

Mahalingam, Krishna., Ollerhead, Andrew., Vardner, Jonathan(2015). *Sonification of Chemical Engineering Data*. Retrieved from https://www.wpi.edu/Pubs/E-project/Available/E-project-032515-172441/

Fogler, H. Scott (2006). Elements of Chemical Reaction Engineering Fourth Edition. Boston, Ma: Prentice Hall.

Marlin, Thomas E.(1995) *Process Control: Designing Processes and Control Systems for Dynamic Perfomance.* New York, NY: McGraw Hill

Fleming, Neil (2014). *VARK: A Guide to Learning Styles.* Retrieved from: http://vark-learn.com/

Unity Manual (2016) Retrieved from http://docs.unity3d.com/Manual/index.html

Appendix

Appendix A

Assignment:

Before starting this assignment, please follow the link below and take the VARK survey about learning styles. Please record your results.

http://vark-learn.com/the-vark-questionnaire/

For this assignment, you will be asked to read the background information on multiple steady states and PID control and complete a short set of assignments using visual and auditory learning tools. Finally, you will be asked to fill out a short survey about the assignment.

Background

Background Info on Multiple Steady States:

The idea of multiple steady states or MSS is a pivotal concept to chemical engineering. MSS traditionally takes place in what is known as a continuous stirred tank reactor or CSTR and involves a balance of energy entering and leaving the tank under multiple varying conditions (multiple steady state equilibrium). In this type of reaction, three possible equilibrium points exist, one upper, one lower, and one middle, with the middle point being the ideal but most unstable. Achieving the ideal conditions required for equilibrium in a CSTR is extremely difficult, as any deviation in the ratio of heat removed to heat put in can cause the system to either push towards the upper or lower equilibrium points. The goal of the assignment presented to you is to achieve the middle state in the shortest amount of time using our program, which involves not only a visual graph, but auditory stimulus in the form of instruments and at the optimal point, a symphony. Before you engage in the assignment, read the information on PID controls below.

Background on PID Control:

Proportional-Integral-Derivative (PID) control is a control loop feedback system that is commonly used in many applications. PID control is used to manipulate a system to converge to a desired condition. In PID control, the control system measures the current state of the overall system and calculates the error from the desired condition using the formula stated below.

$$E(t) = C_{desired} - C_{output}$$

In a PID controlled system, the error is time dependent, meaning that the difference between the desired condition and the current condition will change over time. The control system then attempts to minimize this error by adjusting a control variable through the use

of a differential equation. The error is used in three separate terms; the proportional term, the integral term, and the derivative term.

$$□(□) = □_0[□_□□(□) + □_□ \int_0^□ □(□)□□ + □_□ \frac{□□(□)}{□□}]$$

$K_P E(t)$ is the proportional term. This term multiplies the current instantaneous error by K_P and has more influence on the output than the integral or derivative terms. $□_□ \int_0^□ □(□)□□$ is the integral term. By integrating the error from the start time until the current time, this term incorporates all the past values of the error into the output. $□_□ \frac{□□(□)}{□□}$ is the derivative term. It finds the instantaneous slope of the error at the current time and multiplies it by K_D. This allows the equation to predict future values of the error and potentially dampen oscillations that are common with this type of control system.

K_P, K_I, and K_D are parameters for the PID control equation that are set by the designer of the control system. These parameters are optimized to control the system property with minimal overshoot, offset, and oscillations, which are problems that result from an improperly controlled system. There are several methods for optimizing these parameters, the most common of which are the *Ziegler-Nichols Tuning Method,* and the *Riggs Tuning Method.*

In the Continuously Stirred Tank Reactor introduced in the previous section, PID control can be used to control the temperature of the cooling jacket. This allows the jacket to remove the correct amount of heat to keep the reactor at an unstable steady state. The PID control system in this assignment has been modeled in this way, but the focus of the assignment is on PID control and therefore only the error values have been plotted.

Software Assignment

Program Information

In this set of assignments, you are tasked to input different values for each of the PID control parameters and note the effects of each on the behavior of the error. The program interface is shown in figure 1.

Figure 1: Program Interface

The program consists of four elements; the K value inputs, the error vs. time graph, the Stop/Reset button, and the audio element. The K value inputs are editable text boxes that allow the user to change the K values used in the PID control of the program. These boxes will take will take only integers or floating point numbers. Once all three of these boxes have been edited, the program will use the input values and begin to graph the error output over time. It also starts the audio. If any of the inputs are changes, the program will reset to its initial conditions and continue graphing. The stop/reset button will freeze the graph in place, and stop the audio. Each of the input value boxes needs to be edited before the graph and audio will start again. This will also reset the program to its initial conditions. The audio is also representative of the error values. For most error values the audio is a trumpet playing different pitches. When the error is positive, the trumpet will play high pitches, with very high pitches corresponding to large positive error values. When the error is negative, the trumpet will play low pitches with very low sounds corresponding to large negative error values. As the error value approaches zero, the trumpet will be replaced with a set of instruments playing a piece of music.

Assignment

You will be presented with 4 problems. The goal of each of these problems is to minimize the time it takes for the error value to approach zero. To do this, you need to find the optimal values for Kp, Ki, and Kd.

1) Set Kp = 75 and Ki = 0.1. Please attempt to find an optimal value for Kd.
2) Set Kp = 90 and Kd = 150. Please attempt to find and optimal Ki. (Hint: Try values 0<Ki<=1)
3) Set Ki = 0.015 and Kd = 300. Please attempt to find an optimal Kp.
4) Explore all three values and try to optimize all.

As a final assignment, test the software in any way you can think of and look for any errors you can produce. Once you have completed all parts of this assignment, please complete the survey below.

http://goo.gl/forms/9nCQxNdfhj

Thank you for participating!

Appendix B

Student Responses

Student 1

Record your Vark Scores:
Visual: 4
Aural: 6
Read/Write: 9
Kinesthetic: 9

Record your problem 1 Kd value
7.5

Record your problem 2 Ki value
0.5

Record your problem 3 Kp value
100

Record all three values from problem 4
10, 05. 100

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software
At 10000 it created a time that would be infinite to finish

Are there any suggestions you have to improve the software?
Nope

Additional Comments

Student 2

Record your Vark Scores:
Visual: 3
Aural: 5
Read/Write: 7
Kinesthetic: 10

Record your problem 1 Kd value
1

Record your problem 2 Ki value
0.1

Record your problem 3 Kp value
100

Record all three values from problem 4
100, 001, .01

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software
High numbers make program break

Are there any suggestions you have to improve the software?
Nope

Additional Comments

Student 3

Record your Vark Scores:
Visual: 4
Aural: 4
Read/Write: 4
Kinesthetic: 4

Record your problem 1 Kd value
500

Record your problem 2 Ki value
0.1

Record your problem 3 Kp value
1000

Record all three values from problem 4
110, 0.1, 20

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software
Not enough resolution on the graph. High Kp values cause it to break. Reset button seems to cause it to break

Are there any suggestions you have to improve the software?
Add a vertical line on the graph showing when each "reset" occurs

Additional Comments

Student 4

Record your Vark Scores:
Visual: 6
Aural: 4
Read/Write: 6
Kinesthetic: 0

Record your problem 1 Kd value
100

Record your problem 2 Ki value
0.01

Record your problem 3 Kp value
75

Record all three values from problem 4
100, 0, 0

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software

Are there any suggestions you have to improve the software?
Hard to know if I'm using the program correctly. It would be better if you have some system messages which tell what is going on.

Additional Comments

Student 5

Record your Vark Scores:
Visual: 2
Aural: 5

Read/Write: 7
Kinesthetic: 5

Record your problem 1 Kd value
66

Record your problem 2 Ki value
0.2

Record your problem 3 Kp value
38

Record all three values from problem 4
86, 0.6, 400

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software

Are there any suggestions you have to improve the software?
Showing a list of values tried might be helpful.

Additional Comments
The trumpet sound was too mechanical and harsh.

Student 6

Record your Vark Scores:

Visual: 7

Aural: 8

Read/Write: 6

Kinesthetic: 8

Record your problem 1 Kd value
100

Record your problem 2 Ki value
0.1

Record your problem 3 Kp value
100

Record all three values from problem 4
500, 0.5, 500

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write

Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software
Some times the software doesn't make sound. (when error = 0)

Are there any suggestions you have to improve the software?
Adding a function which hides the graph would help auditory learners concentrate on the sound.

Additional Comments

Student 7

Record your Vark Scores:
Visual: 5
Aural: 6
Read/Write: 7
Kinesthetic: 7

Record your problem 1 Kd value
5

Record your problem 2 Ki value
0.1

Record your problem 3 Kp value
90

Record all three values from problem 4
80, 001, 500

Were the instructions clear?
Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control

1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?

1 2 3 4 5

Please Describe Any Errors you Encountered With the software

Are there any suggestions you have to improve the software?

Additional Comments

Student 8

Record your Vark Scores:
Visual: 2
Aural: 6
Read/Write: 5
Kinesthetic: 3

Record your problem 1 Kd value
100

Record your problem 2 Ki value
0.01

Record your problem 3 Kp value
100

Record all three values from problem 4
10, .01, 200

Were the instructions clear?

Very Confusing 1 2 3 4 5 Very Clear

Was the software simple to use?
Very Confusing 1 2 3 4 5 Very Simple

Did you have an understanding of PID control before this assignment?
No Understanding 1 2 3 4 5 Full Understanding

This software helped me understand PID control
1 2 3 4 5

What Kind of Learner do you consider yourself?

Visual
Auditory
Read/Write
Kinesthetic
Other :

Was the auditory portion of this assignment helpful?
1 2 3 4 5

Please Describe Any Errors you Encountered With the software

Are there any suggestions you have to improve the software?

Additional Comments

Appendix C

C-Sharp Code:

CSTRModelScript

```
public class CSTRModelScript : MonoBehaviour {

    public KValueManagerScript PIDControlSystem; //Object that stores K-Values and calculates PID control

    //Set of Booleans to define whether or not the user has input K-Values.
    //These values are set to false when the stop/reset button is pressed
    private bool KpExists = false;
    private bool KiExists = false;
    private bool KdExists = false;

    private float CSTR_Temp;
    private float jacket_Temp;
    private float target_Temp;
    private float reaction_Heat;
    private float heat_Removed;
    private float Tc;

    static float dH = 57739.2f;
    static float tau = 0.06f;
    static long A = 4000000000;
    static float E = 56902.4f;
    static float R = 8.31434f;

    static float K = 3400f / (300f * 30f);
    static int Cp = 300;
    static float m = 30f;

    static float jacket_Mass = 0.01f;
    static int jacket_C = 41800;



    // Use this for initialization
    void Start () {
        jacket_Temp = 288.15f;
        CSTR_Temp = 370f;
        Tc = (288.15f+K*jacket_Temp)/(1+K);
```

```
        target_Temp = 466.1f - 0.394f * jacket_Temp;

    }

    // Update is called once per frame
    void Update () {
        if (KpExists && KiExists && KdExists) {
            //update reaction_Heat based on CSTR_Temp
            reaction_Heat = (dH * tau * A * Mathf.Exp (-
E / (R * CSTR_Temp))) / (1 + tau * A * Mathf.Exp (-E / (R * CSTR_Temp)));
            //update heat_Removed based on jacket_Temp
            heat_Removed = Cp * (1 + K) * (CSTR_Temp - Tc);

            //update target_Temp using jacket_Temp
            target_Temp = 466.1f - 0.394f * jacket_Temp;

            //update jacket_Temp using PID control
            jacket_Temp += PIDControlSystem.PIDControl (CSTR_Temp,target_Temp,Time.deltaTime) / (ja
cket_C * jacket_Mass);
            Tc = (288.15f + K * jacket_Temp) / (1 + K);

            //update CSTR_Temp based on reaction_Heat-jacket_Temp
            CSTR_Temp += (reaction_Heat - heat_Removed) / (m * Cp);
        }

    }

    public void setKpValue(string Kp){
        PIDControlSystem.KpStringToFloat (Kp);
        KpExists = true;
        jacket_Temp = 288.15f;
        CSTR_Temp = 370f;
    }

    public void setKiValue(string Ki){
        PIDControlSystem.KiStringToFloat (Ki);
        KiExists = true;
        jacket_Temp = 288.15f;
        CSTR_Temp = 370f;
    }

    public void setKdValue(string Kd){
        PIDControlSystem.KdStringToFloat (Kd);
        KdExists = true;
```

```
        jacket_Temp = 288.15f;
        CSTR_Temp = 370f;
    }

    public float getError(){
        return target_Temp - CSTR_Temp;
    }

    public bool kValuesExist(){
        if (KpExists && KiExists && KdExists) {
            return true;
        } else {
            return false;
        }
    }

    public void resetKValues(){
        KpExists = false;
        KiExists = false;
        KdExists = false;

    }
}


KValueManagerScript
public class KValueManagerScript: MonoBehaviour {

    float Kp = 0;
    float Ki = 0;
    float Kd = 0;
    float totalError = 0;
    float previousError;

    public void KpStringToFloat(string stringKp){
        float result;
        float.TryParse (stringKp, out result);
        if (Mathf.Abs (result) <= 2150) {
            Kp = result;
        } else {
            Kp = 2150;
            MonoBehaviour.print ("Kp Value too high. Kp set to 2150");
        }
    }
```

```
    public void KiStringToFloat(string stringKi){
        float result;
        float.TryParse (stringKi, out result);
        if (Mathf.Abs (result) <= 2150) {
            Ki = result;
        } else {
            Ki = 2150;
            MonoBehaviour.print ("Ki Value too high. Kp set to 2150");
        }
    }

    public void KdStringToFloat(string stringKd){
        float result;
        float.TryParse (stringKd, out result);
        if (Mathf.Abs (result) <= 2150) {
            Kd = result;
        } else {
            Kd = 2150;
            MonoBehaviour.print ("Kd Value too high. Kp set to 2150");
        }
    }

    public float PIDControl(float current_Temp, float target_Temp, float timeFrame){
        float Qout;
        float error = target_Temp - current_Temp; //Calculate Error
        totalError += error * timeFrame; //Scale to timeframe
        Qout = ((Kp*error)+(Ki*totalError)+(Kd*(error-previousError))); //Calculate PID Control Output
        previousError = error; //Store error for use in next iteration
        return Qout;
    }
}
```

Graph

```
public class graph : MonoBehaviour {
    private GameObject lineGroup;
    public GameObject panel1;
    public GameObject panel2;
    public Canvas myCanvas;
    private float accTime = 0.0f;
    private float currentPos = 0.0f;
    public CSTRModelScript cstr;
```

```
int n = 0;
int i = 0;
List<float> errors =new List<float>();

//private float accTime = 0.0f;
//private float currentArg = 0.0f;

void DrawLine(List<Vector2> my2DVec, int startPos, int cl){
   List<Vector3> myPoint = new List<Vector3>();
   for (int idx = 0; idx < 2; idx++) {
      myPoint.Add(new Vector3(my2DVec[startPos+idx].x, my2DVec[startPos+idx].y, 0.0f));
   }
   GameObject newLine = new GameObject ("Line" + startPos.ToString());
   LineRenderer lRend = newLine.AddComponent<LineRenderer> ();
   lRend.material = new Material(Shader.Find("Particles/Additive"));
   if (cl == 0)
      lRend.SetColors (Color.gray, Color.gray);
   else lRend.SetColors (Color.magenta, Color.magenta);
   lRend.SetVertexCount(2);
   lRend.SetWidth (0.05f, 0.05f);
   Vector3 startVec = myPoint[0];
   Vector3 endVec   = myPoint[1];
   lRend.SetPosition (0, startVec);
   lRend.SetPosition (1, endVec);

   newLine.transform.parent = lineGroup.transform;
}
void drawGraph(List<Vector2> my2DVec, GameObject panel, int cl) {
   lineGroup = new GameObject ("LineGroup");

   for (int idx=0; idx < my2DVec.Count - 1; idx++) {
      DrawLine (my2DVec, /* startPos=*/idx, cl);
   }

   lineGroup.transform.parent = panel.transform; // to belong to panel
}

void clearGraph(GameObject panel) {
   foreach (Transform line in panel.transform) {
      if (line.gameObject.name.Contains("LineGroup")) {
         Destroy(line.gameObject);
      }
   }
}
```

```
    void addPointNormalized(List<Vector2> my2DVec, GameObject panel, Vector2 point)
    {
        // point: normalized point data [-1.0, 1.0] for each of x, y

        RectTransform panelRect = panel.GetComponent<RectTransform> ();
        float width = panelRect.rect.width;
        float height = panelRect.rect.height;

        RectTransform canvasRect = myCanvas.GetComponent<RectTransform> ();

        Vector2 pointPos;

        // Bottom Left
        pointPos = panel.transform.position;
        pointPos.x += point.x * width * 0.5f * canvasRect.localScale.x;
        pointPos.y += point.y * height * 0.5f * canvasRect.localScale.y;
        my2DVec.Add (pointPos);
    }
    void drawBox(List<Vector2> my2DVec, GameObject panel)
    {
        addPointNormalized (my2DVec, panel, new Vector2 (-1.0f, -1.0f));
        addPointNormalized (my2DVec, panel, new Vector2 (-1.0f, 1.0f));
        addPointNormalized (my2DVec, panel, new Vector2 (1.0f, 1.0f));
        addPointNormalized (my2DVec, panel, new Vector2 (1.0f, -1.0f));
        addPointNormalized (my2DVec, panel, new Vector2 (-1.0f, -1.0f));

        drawGraph (my2DVec, panel,0);
    }

    void drawGraph1(List<Vector2> my2DVec, GameObject panel, float currentPos, List<float> errors, in
t cl)
    {
        float step = 0.05f;
        float x = 0.0f;
        //x += currentPos;
        int k =i;

        while (x < 2.0f) {

            //y = -1/(x+currentPos);

            addPointNormalized(my2DVec, panel, new Vector2(x-1.0f, errors[k]));
            x += step;
```

```
            k++;


        }
        i++;
        drawGraph (my2DVec, panel, cl);
    }
    void drawGraph2(List<Vector2> my2DVec, GameObject panel)
    {
        float step = 0.01f;
        float x = 0.0f;
        float y = 0.0f;
        //x += currentPos;

        while (x < 2.0f) {

            y = x*x;

            addPointNormalized(my2DVec, panel, new Vector2(x-1.0f, y));
            x += step;


        }
        drawGraph (my2DVec, panel,0);
    }

    // Use this for initialization
    void Start () {

        for (int c = 0; c < 41; c++) {
            errors.Add (0.0f);
        }
        //List<Vector2> my2DPoint = new List<Vector2> ();
        panel1 = GameObject.FindWithTag ("Player");
        panel2 = GameObject.FindWithTag ("Panel2");
        //drawGraph1 (my2DPoint, panel1, currentPos, errors);
        //drawGraph2 (my2DPoint, panel2);
        /*
        List<Vector2> my2DPoint = new List<Vector2> ();
        clearGraph (panel1);
        //drawBox (my2DPoint, panel1);
        drawGraph1 (my2DPoint, panel1,0);*/
    }
```

```
    // Update is called once per frame
    void Update () {
        if (cstr.kValuesExist ()) {
        accTime += Time.deltaTime;
        if (accTime < 0.3f) {
            return;
        }
        List<Vector2> my2DPoint = new List<Vector2> ();
        accTime = 0.0f;

            //errors.Add (currentPos*currentPos);
        errors.Add (cstr.getError()/10);
        //System.Console.WriteLine ("{0}¥n", cstr.getError ());
        clearGraph(panel1);

            if (n < 1) {
                drawGraph1 (my2DPoint, panel2, currentPos, errors, 0);
                i--;
                n = 1;
            }

            drawGraph1 (my2DPoint, panel1, currentPos, errors, 1);
            currentPos += 0.05f;

        }

    }
}
```

Trumpet Audio Control

```
    private AudioSource source;

    public AudioClip lowE;
    public AudioClip lowF;
    public AudioClip lowG;
    public AudioClip middleA;
    public AudioClip middleB;
    public AudioClip middleC;
    public AudioClip middleD;
    public AudioClip middleE;
    public AudioClip middleF;
```

```
public AudioClip middleG;
public AudioClip highA;
public AudioClip highB;
public AudioClip highC;
public AudioClip highD;
public AudioClip highE;
public AudioClip highF;
public AudioClip highG;
public AudioClip highestA;
public AudioClip highestB;

public CSTRModelScript cstr;

// Use this for initialization
void Start () {
    source = gameObject.GetComponent<AudioSource> ();
}

// Update is called once per frame
void Update () {
    //source.Stop ();
    if (cstr.kValuesExist()) {
        source.volume = 1.0f;
        if (cstr.getError () < -14f) {
            source.PlayOneShot (lowE, 0.75f);
        } else if (cstr.getError () < -12f) {
            source.PlayOneShot (lowF, 0.75f);
        } else if (cstr.getError () < -10f) {
            source.PlayOneShot (lowG, 0.75f);
        } else if (cstr.getError () < -8f) {
            source.PlayOneShot (middleA, 0.75f);
        } else if (cstr.getError () < -6f) {
            source.PlayOneShot (middleB, 0.75f);
        } else if (cstr.getError () < -4f) {
            source.PlayOneShot (middleC, 0.75f);
        } else if (cstr.getError () < -2f) {
            source.PlayOneShot (middleD, 0.75f);
        } else if (cstr.getError () < -0.1f) {
            source.PlayOneShot (middleE, 0.75f);
        } else if (cstr.getError ()  < 0.1f) {
            source.Stop ();
        } else if (cstr.getError () < 2f) {
            source.PlayOneShot (highA, 0.75f);
        } else if (cstr.getError () < 4f) {
```

```
            source.PlayOneShot (highB, 0.75f);
        } else if (cstr.getError () < 6f) {
            source.PlayOneShot (highC, 0.75f);
        } else if (cstr.getError () < 8f) {
            source.PlayOneShot (highD, 0.75f);
        } else if (cstr.getError () < 10f) {
            source.PlayOneShot (highE, 0.75f);
        } else if (cstr.getError () < 12f) {
            source.PlayOneShot (highF, 0.75f);
        } else if (cstr.getError () < 14f) {
            source.PlayOneShot (highG, 0.75f);
        } else if (cstr.getError () < 16f) {
            source.PlayOneShot (highestA, 0.75f);
        } else if (cstr.getError () > 16f) {
            source.PlayOneShot (highestB, 0.75f);
        }
    } else {
        source.volume = 0f;
    }
  }
}
```