204 Script:

The video presentation should be embedded into its own individual thread on Piazza under the folder for 'Project Submission'.

Your video should not be longer than 10 minutes (ideally ~8 min). Please upload your video onto any video sharing site such as Youtube. Do not upload your video onto Piazza, there is a limit on storage for the entire course on Piazza which mean in all likelihood it will run out of storage before everyone have submitted their video.

The presentation should be a standalone report of your group's work, that is it should contain all relevant information you think is necessary for the audience to (i) understand your results, (ii) be confident that your results are trustworthy and valid, and (iii) understand the implication of your results. You may also attached in the same post any material that you think are redundant to be presented in your presentation but it will further strengthen your main argument/discussion, or if you have any other materials that you think will strengthen your discussion section. In other words, you may attach an Appendix section (in text form or in video form, or in graphical form ,etc.) along with your videos.

There are no format restrictions on how you present your results. This means that you can do a typical presentation that you videotaped, or do a screencast with a voice over, do a cartoon/animation, etc.

Tip for Success: Do not underestimate the amount of planning that is necessary to condense an entire project into 10 minutes, so plan accordingly.

https://editor.animatron.com

***Purple == Animation**

ACTUAL SCRIPT:

- ## Ice Breaker: 30s

  ** FADE INTO ANIMATION OF A GUY SITTING/PONDERING*

  Customer: Why does this happen!? No matter how hard I try, I can never get this ball to balance on a shifting stand! Oh how I wish I had a ball balancer!

  Guy 1 & 2: We have the product just for you sir! (SOUND AS IF YOU ARE FAR AWAY?)

  **CUSTOMER LOOKS SHOOK, SOME ANIMATION DEPICTING THAT*

  Customer: What the...

  **TRANSITION INTO ANIMATION OF TWO SUITED UP DOODS STANDING THERE*

  Guy 1: We heard your plea and we are here to present the ball balancer 5000 mark I.

  Guy 2: Which uses an innovative technique know as a PID controller to balance the ball!

- ## Intro: 45s - 1 min

  **TRANSITION TO CUSTOMER TALKING*

  Customer: Ok, that's great and all you weirdos but what exactly is a PID controller and why should I even care about it? I just wanted a ball…

  **IMAGE (OR SLIDESHOW) OF GENERAL PID STUFF, SUCH AS FIGURE 1 OF THE REPORT, FOR EXAMPLE*

  Guy 1: A PID controller is a representation of a closed-loop system. This systematic controller works by constantly calculating the error value as the difference between a desired setpoint and a measured variable. A correction based on the proportional, integral and derivative of the error is then applied to actuate the system to the desired value. A PID controller is a control feedback mechanism which is used in industry for many different applications , for instance a furnace temperature control.

  Customer: Or an innovative, disruptive ball balancer.

  Guy 2: Exactly!

- ## PID Algorithm and its relation to Numerical Methods: 2 mins

  **FACE OF CUSTOMER LOOKING CONFUSED*

  NOTE: Regarding the *SHOW EQUATION* parts down below, show the equation as you are talking about it, then transition to the next (any way we want) *SHOW EQUATION*

  Customer: But how exactly does this PID algorithm work?

Guy 2: The PID algorithm starts with determining the current error value. The error is calculated as the difference between a variable's current value and its desired value, which we will call the setpoint **SHOW EQUATION**. The setpoint can be defined as any value that you want the variable to reach and stay at, in our case the position of rest for the ball. After the error has been calculated the derivative and integral of the error can be determined. The derivative can be calculated by using the current and past error values and determining the slope between the two points **SHOW EQUATION**. The integral can be calculated as the sum of all the past error values multiplied the time difference between each point **SHOW EQUATION**. Once all three of these values are determined they can be used to get the correctional output value for the systems actuator. The output is simply the equation here **SHOW EQUATION**. The three K values in the equation are the different gain values, these values are determined experimentally and are very important in ensuring that your system gives the correct output.

**FACE OF CUSTOMER LOOKING ENLIGHTENED OR INTERESTED*

Customer: Very cool! how do each of the three values impact the correction for the error?

**ONE SUITED UP DOOD TALKING*

Guy 1: Great question, well the error value has the largest impact on the output value and decreases the amount of time it would take to remove the error, but if used alone would leave the system in steady state error. So in order to remove this the integral portion. However, this value continues to grow as long as error exists in the system. As such, the integral portion removes the steady state error but could result in overshooting the setpoint or having the variable larger than the setpoint. In order to correct appropriately for this the derivative is added this counteracts the overshooting, resulting in the output value converging on the setpoint and staying there.

**FACE OF CUSTOMER LOOKING ENLIGHTENED OR INTERESTED*

Customer: Once the system gets the value to its setpoint how does it make sure that it stays there?

**ONE SUITED UP DOOD TALKING*

Guy 1: Well for this implementation of a PID controller we used a method called Fixed Point iteration. This method continuously re-evaluates the same equation with new inputs every time. In other words the system will output a value from the PID algorithm and this output will move the ball to reduce the error. Now the next time the equation is going to be evaluated it will use a new error value which was determined by the last output value, and compute a new output value. And this process continues until the ball has reached the setpoint. Now although the ball has reached its setpoint the equation will still be continuously re-evaluated to see if an external force has moved the ball, and if so it will begin the re-balancing process again.

**FACE OF CUSTOMER SURPRISED*
Customer: Wow, this process seems too simple, is that really all?
Guy 2: Yes, this is all the logic. That is why the PID controller is a perfect solution to this problem. It does not require any complex math equations or tons of sensors. Also the logic can be used to solve similar problems for different systems. The only thing that needs to be changed are the three gain values since they are system dependent.

- ## Ball Balancer: 1-2 min

Customer: So how exactly does this system work?
Guy 2: The system begins with a known setpoint, the place you want the ball to stay at. The ball can then be placed at any random position on the platform. The sonar sensor will determine the relative position of the ball. This positional value allows the error for the location of the ball to be calculated along with the integral, and derivative. Then as mentioned before these values will be multiplied by their respective gain values and then summed to give the output value of the system. This output correlates as a PWM signal to the servo motor which adjusts the platform, thus moving the ball. This iterative process continuously runs even when the ball has reached the setpoint.
Customer: This is good information but could you be more specific about what the Arduino is doing?
Guy 1: Sure, the code has two main functions and three helper functions. The two main functions are the *setup* and *loop*. The setup function is run once when the board has been loaded with the compiled code. Inside of this function are initializations and pin setup. The *loop* function is continuously called as long as the board has power. This function contains the PID controller logic. The logic can be seen in this flow diagram. **SHOW FLOW DIAGRAM** The logic uses the three helper functions which are *getSonarValue*, *getBallLocation* and *calculate*. The *getSonarValue* function simply sends a signal to different pins on the sonar sensor and calculates the duration that it takes for the sound wave to travel to the object and bounce back. This function returns the travel time in milliseconds. The *getBallLocation* uses the *getSonarValue* function and converts the time to a distance in centimeters and then ensures that the distance is within a reasonable range of values and returns the location of the ball in centimeters. The final function *calculate* takes in the current location of the ball in centimeters and computes the error, integral of the error, derivative of the error, which are all used to determine the output of the PID algorithm. The function then makes sure that the output is within appropriate bounds and returns the output.
Customer: Thank you that was quite detailed.

- **PID Apparatus Limitations: 1 min**

  Customer: This all seems great but there must be some issues with the system.
  Guy 2: There are some minor limitations to the system. The sonar sensor is not very accurate. It cannot accurately detect the ball when it is further than 20cm from it. This decreases the efficiency of the system and forces a strict range on the setpoint. When we created the system we made sure that the actuator would never be saturated.
  Customer: How can the actuator become saturated?
  Guy 1: This is when the actuator, in our case the servo, is not large enough to cause a large control effect as request by the controller. This causes the integrated error to continuously grow. A large integrated error will result in incorrect output values from the PID controller.
  Customer: Good thing you guys have that covered.
- **Conclusion/Recommendations 30 seconds**

  Guy 1: Now if you think that this product is great, you will really enjoy our next model, which has many improvements on the current one. It has the Parallax Ping as the sonar sensor which has a much greater range of detection. So you would have a larger range for the setpoint. Also the new system has a more powerful servo motor. This allows for faster and more precise movements, which will decrease the amount of time that it takes to balance the ball. The microcontroller has been replaced with a beaglebone which will be used for higher numerical accuracy and faster calculation time. Lastly, we modeled the system in Matlab and found the most optimal gain values for the system. This will maximize the efficiency of the entire system. Sadly this system will only be coming out in 2020.

- **Results from Test Cases: 1-2 min**

  Customer: Woah woah, slow down there sirs! I mean this great and all but uhm… How do I trust that your product actually works? Do you have any data or graphs to show me that it is actually a feasible system and algorithm?
  Guy 2: Of course dear customer. So with a PID controller, there is a lot of tuning required to obtain the desired outcome. In our case, we had to tune the gain values, the kp, ki and kd values, to a perfect value.
  Customer: Ok… why does that even matter?
  Guy 2: Well, the gain values affect the behaviour of our system. This table depicts how each gain value affects the system. **SHOW TABLE**
  Guy 1: So we adjusted the gain values based on these parameters. It required a lot of trial and error but eventually our system worked!
  Customer: Ok, cool but you still haven't proved to me that it works

Guy 2: Yes you are correct, apologies. As we mentioned previously, our current system is limited by the servo and the sensor and as such, we picked a setpoint value of 6 cm and a servo output value range limited from 0 - 33 duty cycles. We have data from a recent test run just for you, depicting that our ball balancer can, indeed, balance.

ERROR X TIME GRAPH

Guy 2: This graph depicts the error of the system over time in a test run, where the error is defined as the difference between the setpoint and the current location of the ball (with respect to the sonar sensor). When the error reaches zero, it signifies that the ball has reached the setpoint, signifying that it has been balanced.

SONAR OUTPUT X TIME GRAPH

Guy 1: The sonar sensor readings vary from 3 to 20 centimeters. The group decided to make the maximum distance from the sonar sensor 20 centimeters due to the unreliability of the sonar sensor for values exceeding 20 centimeters. This graph then, is simply the location of the ball with respect to the sonar sensor. We chose a setpoint of 6 cm, meaning that when the ball reaches 6 cm, it is balanced.

SERVO PWM X TIME GRAPH

Guy 2: Finally, this graph shows the pwm value that the servo will read. This is the means of which actual balancing, where the servo is connected to an arm that moves the ball up or down to reach a setpoint.

FADE TO ONE DOOD TALKING

Guy 2: This proves our PID algorithm and consequently, the ball can balance because all the expected values appear for the chosen setpoint and all three graphs converge at the same time value.

- ## Show PID in progress: 1 min

Customer: WOW, I learnt a lot today but I still haven't seen your product in action. Can I?

Guy 1: Off course! Here is a demonstration video of how the system works.

SHOW VIDEO OF THE REAL BALL BALANCER

Guy 2: So after seeing that how much are you willing to purchase this for?

Customer: As much as I can, this project is waterloo engineering in its finest! Now, get out of my room!