

Design a Temperature controller for Directed Light

Project: Jonathan Zhong, Ryan Zhang

Well Pool Temperature Control

Introduction
The objective of this project is to design a temperature controller for a laser diode (LD) that can maintain a constant temperature of 25°C. The LD is a sensitive component that requires precise temperature control to operate correctly. The system is modeled as a second-order system with the following state-space representation:

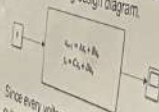
$$\dot{x}(k+1) = Ax(k) + Bu(k)$$

$$y(k) = Cx(k) + Du(k)$$

where $x(k)$ is the state vector, $y(k)$ is the output temperature, and $u(k)$ is the input driving voltage of laser power.

Control Parameters
The dynamic model is identified using second order state-space model with the form

Control Process
The control system design is based on a feedforward control loop. The first step is to generate a voltage-temperature lookup table using the following design diagram:



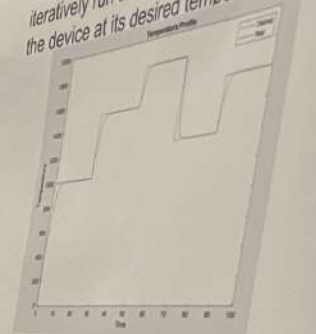
Since every voltage has its unique temperature output, when we get a temperature output, we can inversely find the voltage input.

For example, when we have 1.42V, the temperature output is 2191.2K, vice versa.

To start with temperature controller, we first need to set the temperature we desire into the control model. The initial voltage state is set as 0. The figure below is the control diagram we build.

Conclusion
DLD has a bright and wide application scenarios in the future. With feedforward control system, the DLD will self-correct and have less flaws and defects on the produced parts. This poster shows a possible method of achieving it.

Mentor: Kezi Li



The above figure shows how temperature desired to change and in real. We can see there will always be delay to change from one temperature to another. Code added below as reference.

```

% Parameters of the system
A = [0.0001 0.0001; 0.0001 0.0001];
B = [0.0001; 0.0001];
C = [0.0001 0.0001];
D = [0.0001];

% Reference temperature
T_ref = 25;

% Initial conditions
x0 = [0; 0];

% Control loop
for k = 1:1000
    % Calculate the error
    error = T_ref - y(k);

    % Calculate the control signal
    u(k) = error;

    % Update the state
    x(k+1) = A*x(k) + B*u(k);

    % Calculate the output
    y(k) = C*x(k) + D*u(k);
    
```

