ĐẠI HỌC CÔNG NGHỆ - ĐẠI HỌC QUỐC GIA HÀ NỘI

BÁO CÁO MÔN HỌC

KỸ THUẬT ĐIỀU KHIỂN NÂNG CAO

XÂY DỰNG BỘ ĐIỀU KHIỂN PID ĐIỀU KHIỂN ĐỘNG CƠ

Học viên: **Phạm Tiến Thành**

Giảng viên hướng dẫn: **TS. Phạm Minh Triển**

Hà Nội, 2019

Lời nói đầu

Bộ môn Kỹ thuật điều khiển nâng cao dưới sự hướng đẫn của TS. Phạm Minh Triển cùng lớp cao học khóa K24 và K25 tuy diễn ra trong thời gian ngắn và thời lượng lên lớp cũng không được nhiều, nhưng vẫn mang lại cho lớp cũng như riêng bản thân tôi nhiều kiến thức quý báu về lý thuyết cũng như cuộc sống. Không những thế việc thực hành bài tập mang lại cho lớp một không khí vui vẻ và cởi mở. Nhân đây tôi xin cảm ơn thầy vì thầy đã không chỉ mang đến một môn học mà còn mang đến cả một không khí rất mới rất tươi trẻ. Xin cảm ơn thầy.

Bài báo cáo về chủ đề "Xây dựng bộ điều khiển PID, điều khiển tốc độ động cơ", báo cáo này đươc chia theo thứ tự các giai đoạn học viên tiến hành tìm hiểu và nghiên cứu:

Giai đoạn 1: Tìm hiểu lý thuyết về PID

Giai đoạn 2: Thiết kế bộ điều khiển PID

Giai đoạn 3: Xây dựng ứng dụng điều khiển từ thiết bị android

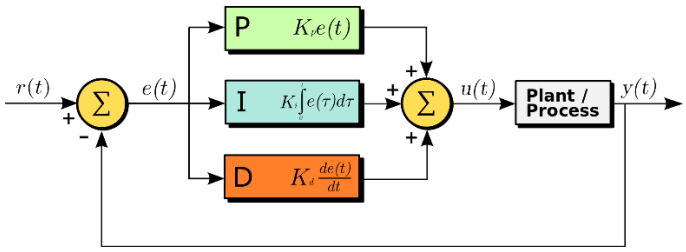
Báo cáo còn nhiều thiếu sót do thời gian thực hành còn nhiều eo hẹp. Mong thầy và các bạn cùng đóng góp để hoàn thiện hơn.

Trong bài có tham khảo một số tài liệu tại Wikipedia. Toàn bộ source code của bộ môn được upload lên Github tại địa chỉ https://github.com/thanh-pt/auto-tune-pid.

Xin cảm ơn!

GIAI ĐOẠN 1

TÌM HIỂU LÝ THUYẾT VỀ PID



Mô hình bộ điều khiển PID

Một bộ điều khiển vi tích phân tỉ lệ (bộ điều khiển PID- Proportional Integral Derivative) là một cơ chế phản hồi vòng điều khiển (bộ điều khiển) tổng quát được sử dụng rộng rãi trong các hệ thống điều khiển công nghiệp – bộ điều khiển PID là bộ điều khiển được sử dụng nhiều nhất trong các bộ điều khiển phản hồi. Bộ điều khiển PID sẽ tính toán giá trị "sai số" là hiệu số giữa giá trị đo thông số biến đổi và giá trị đặt mong muốn. Bộ điều khiển sẽ thực hiện giảm tối đa sai số bằng cách điều chỉnh giá trị điều khiển đầu vào. Trong trường hợp không có kiến thức cơ bản (mô hình toán học) về hệ thống điều khiển thì bộ điều khiển PID là sẽ bộ điều khiển tốt nhất. Tuy nhiên, để đạt được kết quả tốt nhất, các thông số PID sử dụng trong tính toán phải điều chỉnh theo tính chất của hệ thống-trong khi kiểu điều khiển là giống nhau, các thông số phải phụ thuộc vào đặc thù của hệ thống.

Giải thuật tính toán bộ điều khiển PID bao gồm 3 thông số riêng biệt, do đó đôi khi nó còn được gọi là điều khiển ba khâu: các giá trị tỉ lệ, tích phân và đạo hàm, viết tắt là P, I, và D. Giá trị tỉ lệ xác định tác động của sai số hiện tại, giá trị tích phân xác định tác động của tổng các sai số quá khứ, và giá trị vi phân xác định tác động của tốc độ biến đổi sai số. Tổng chập của ba tác động này dùng để điều chỉnh quá trình thông qua một phần tử điều khiển như vị trí của van điều khiển hay bộ nguồn của phần tử gia nhiệt. Nhờ vậy, những giá trị này có thể làm sáng tỏ về quan hệ thời gian: P phụ thuộc vào sai số hiện tại, I phụ thuộc vào tích lũy các sai số quá khứ, và D dự đoán các sai số tương lai, dựa vào tốc độ thay đổi hiện tại.

Bằng cách điều chỉnh 3 hằng số trong giải thuật của bộ điều khiển PID, bộ điều khiển có thể dùng trong những thiết kế có yêu cầu đặc biệt. Đáp ứng của bộ điều khiển có thể được mô tả dưới dạng độ nhạy sai số của bộ điều khiển, giá trị mà bộ điều khiển vọt lố điểm đặt và giá trị dao động của hệ thống. Lưu ý là công dụng của giải thuật PID trong điều khiển không đảm bảo tính tối ưu hoặc ổn định cho hệ thống.

Vài ứng dụng có thể yêu cầu chỉ sử dụng một hoặc hai khâu tùy theo hệ thống. Điều này đạt được bằng cách thiết đặt đội lợi của các đầu ra không mong muốn về 0. Một bộ điều khiển PID sẽ được gọi là bộ điều khiển PI, PD, P hoặc I nếu vắng mặt các tác động bị khuyết. Bộ điều khiển PI khá phổ biến, do đáp ứng vi phân khá nhạy đối với các nhiễu đo lường, trái lại nếu thiếu giá trị tích phân có thể khiến hệ thống không đạt được giá trị mong muốn.

GIAI ĐOẠN 2

THIẾT KẾ BỘ ĐIỀU KHIỂN PID TRÊN ARDUINO

1. Xây dựng bộ điều khiển

Ta có công thức của bộ điều khiển PID như sau:

$$\mathrm{u(t)} = \mathrm{MV(t)} = K_p e(t) + K_i \int_0^t e(\tau)\,d\tau + K_d \frac{d}{dt} e(t)$$

Từ đó ta xây dựng được hàm cho mạch Arduino như sau:

```
void updateMeasuredValue(float measured_value){
  m_error = m_setPoint - measured_value;
  m_integral = m_integral + m_error * m_dt;
  m_derivative = (m_error - m_previous_error) / m_dt;

  m_output = m_Kp * m_error + m_Ki * m_integral + m_Kd * m_derivative;

  m_previous_error = m_error;
}
```

Mỗi khi giá trị đo lường được, ta sẽ gọi đến hàm này để cập nhật giá trị điều khiển.

Tuy đơn giản nhưng rõ ràng với mỗi hệ thống khác nhau thì tín hiệu điều khiển đầu ra cũng khác nhau và cách thức điều khiển cũng khác nhau. Nên với bài toán này cũng như một số kiến thức đã học về hướng đối tương nên tôi đã xây dựng bài toán này theo kiểu hướng đối tượng như sau.

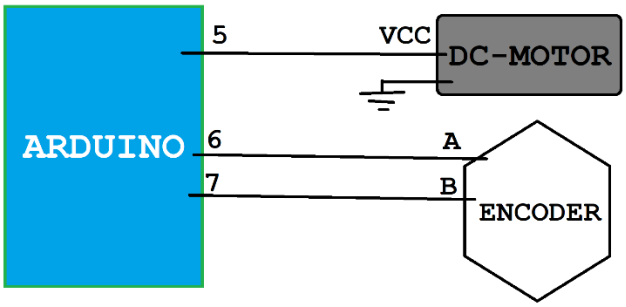
Trong class cha ta có hàm sau:

```
void update(float measured_value){
  updateMeasuredValue(measured_value);
  convert(m_output);
  control();
};
```

Trong đó hai hàm "*convert*" và "*control*" là các hàm thuần ảo và khi mình thiết kế cho các hệ thống riêng biệt mình sẽ viết hàm này tương ứng. Ví dụ như trong ứng dụng này hai hàm được viết lại như sau:

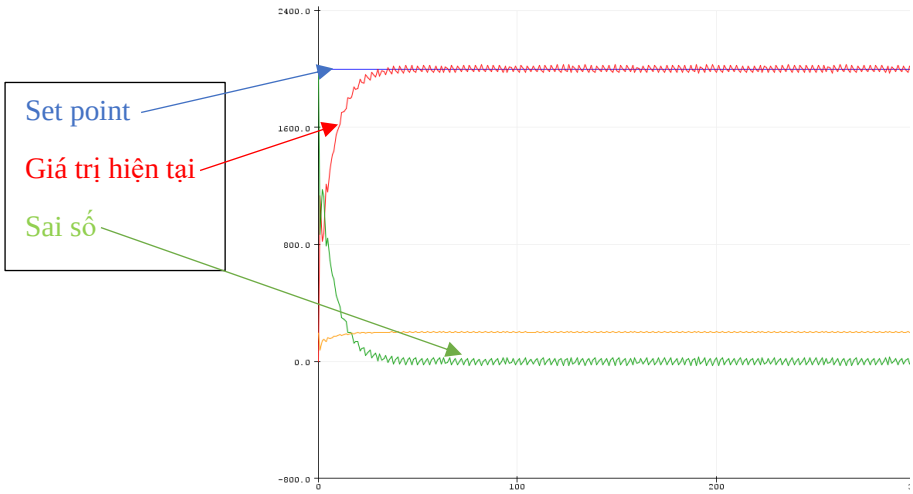
```
void convert(float& output){
  output = round(output);
}
void control(){
  int a = round(getOutput());
  analogWrite(m_control_pin, a);
}
```

2. Lắp đặt phần cứng và thiết lập hệ số PID



Mô hình phần cứng

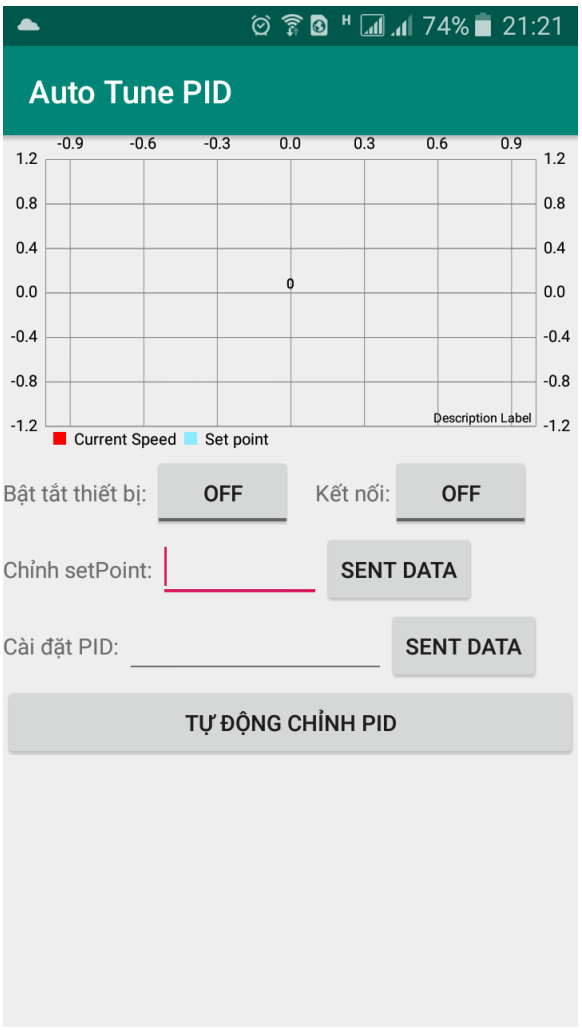
Áp dụng theo phương pháp Ziegler–Nichols. Thực nghiệm nhiều lần ta có kết quả sau:



Ứng với PID tương ứng là: 0.045, 0.12 và 0

GIAI ĐOẠN 3

XÂY DỰNG ỨNG DỤNG ĐIỀU KHIỂN TỪ THIẾT BỊ ANDROID



Giao diện ứng dụng

Ứng dụng được xây dựng bằng phần mềm Android Studio với ngôn ngữ java. Ứng dụng kết nối với hệ thống động cơ và Arduino qua đường truyền Bluetooth sử dụng module HC05 và có các chức năng chính như sau:

1. Bật tắt thiết bị
2. Cài đặt setpoint cho hệ thống
3. Cài đặt PID bằng tay
4. Vẽ biểu đồ trạng thái setpoint và vận tốc tức thời
5. Tự động điều chỉnh PID (chưa hoàn thành)

Cách thức truyền nhận dữ liệu/ gói tin. Đây là một phần tương đối quan trọng trong việc truyền nhận dữ liệu. Vì gói tin mà đóng gói không đúng cũng như không có sự hiểu và thông suốt giữa phần gửi và phần nhận sẽ không đạt được mục đích mong muốn. Với một ứng dụng nhỏ chỉ với số ít lệnh tôi đã xây dựng gói tin với cấu trúc nhưu sau:

Gói tin = "**cmd**" + "**parameter**"

Trong đó "**cmd**" là mã lệnh và "**parameter**" là các biến số truyền đi ứng với mỗi mã lệnh

KẾT LUẬN

Đi kèm với báo cáo này là một sản phẩm hoàn thiện có thể hoạt động được tương thích với ứng dụng đã thiết kế. Sản phẩm mang tính chất đóng gói cẩn thận. Tuy chưa được hoàn thiện 100% về chức năng nhưng đã phần nào đạt chỉ tiêu đề ra ban đầu.