Implementação de um controle realimentado de baixo custo em um trocador de calor de casco-tubo

Implementation of low-cost feedback control in a shell-and-tube heat exchanger

Raul Guilherme Dias[1], Felipi Luiz de Assunção Bezerra[2]

RESUMO

O controle de processos industriais é feito por meio da utilização de controladores lógicos programáveis (CLP's). Esses controladores têm um elevado custo de implementação, o que torna inviável sua implementação em sistemas mais simples de baixo custo. Como alternativa, surge a possibilidade de utilização do Arduino para implantação de um controle PID de baixo custo. O objetivo deste trabalho é o desenvolvimento de um controle de temperatura de saída do casco manipulando a rotação de uma bomba centrífuga e, consequentemente, a vazão de fluido no casco. A metodologia empregada consistiu em utilizar um módulo educacional de trocador de calor casco e tubos em combinação com os componentes instalados no Arduino. Após a análise do funcionamento do trocador de calor, foi constatado que a utilização de um controle PID com Arduino é possível, visto que dentro da faixa de operação da bomba, o controle conseguiu manter o processo próximo ao valor de referência (setpoint). É possível aumentar a faixa de operação do controle utilizando uma bomba de maior capacidade, reduzindo os pontos de perda de carga do sistema ou, como melhoria, utilizando um método de auto sintonia.
PALAVRAS-CHAVE: Arduino, controle PID; trocador de calor.

ABSTRACT

The control of industrial processes is carried out through the use of programmable logic controllers (PLCs). These controllers have a high implementation cost, making it impractical to implement them in simpler low-cost systems. As an alternative, the possibility of using Arduino for the implementation of low-cost PID control arises. The aim of this work is to develop temperature control of the outlet of the shell by manipulating the rotation of a centrifugal pump and, consequently, the fluid flow in the shell. The methodology employed consisted of using an educational module of a shell-and-tube heat exchanger in combination with the components installed on the Arduino. After analyzing the operation of the heat exchanger, it was found that using PID control with Arduino is feasible since, within the operating range of the pump, the control was able to maintain the process close to the setpoint value. It is possible to increase the control's operating range by using a pump with higher capacity, reducing system pressure drop points, or as an improvement, using an auto-tuning method.
KEYWORDS: Arduino, PID control; heat exchanger.

INTRODUÇÃO

O processo de transferência de calor entre dois fluídos em diferentes temperaturas e separados por uma parede sólida ocorre em muitas aplicações na engenharia. O dispositivo utilizado para implementação desta troca é chamado trocador de calor, dentre os diferentes tipos, o mais utilizado é o trocador casco e tubos (BARGMAN; LAVINE, 2017).

Para a operação de processos térmicos com segurança e confiabilidade, como em um trocador casco e tubo, a aplicação de um controle *feedback* é amplamente utilizada. Dentre os controles feedback utilizados industrialmente, o controle Proporcional-Integral-

[1] Discente do Curso de Bacharelado em Engenharia Química. Universidade Tecnológica Federal do Paraná, Londrina, Paraná, Brasil. E-mail: rauldias@alunos.utfpr.edu.br. ID Lattes: 2872922582158678.
[2] Docente do Departamento de Engenharia Química. Universidade Tecnológica Federal do Paraná, Londrina, Paraná, Brasil. E-mail: felipibezerra@utfpr.edu.br. ID Lattes: 8529742705597350.

Derivativo (PID) é o mais comum (ROHANI, 2017). Neste tipo de controlador, três termos de controle são aplicados para obter, de acordo com a entrada, uma saída que busque atingir o valor desejado (*setpoint*). Sendo o termo "Proporcional" um valor proporcional ao valor da diferença entre o *setpoint* e a variável controlada (erro), o termo "Integral" levando em conta valores passados do erro e o termo "Derivativo" uma estimativa para o comportamento futuro do erro (SEBORG *et al.*, 2017; KHERKHAR *et al.*, 2022).

Usualmente a implementação do controlador é feita em um controlador de lógica programável (*CLP*), cujo preço de aquisição e instalação encontra-se em torno dos milhares de reais (ROHANI, 2017). Por esse motivo, faz-se necessário obter alternativas para que seja possível controlar equipamentos com valores mais acessíveis, especialmente para fins didáticos. Uma alternativa é o uso de microcontroladores de prototipagem como Arduino, Raspberry Pi, BeagleBone Black, ESP32, STM32 e PIC (JAMIESON; HERDTNER, 2015).

O presente trabalho utiliza o Arduino, plataforma física de computação de código aberto baseada em uma placa simples de entradas e saídas (*Input* e *Output*, *I/O*) e um ambiente de desenvolvimento que implementa a linguagem *Processing* (BANZI, 2009). A plataforma é versátil e pode ser utilizada para diversas aplicações.

No Arduino as saídas analógicas podem fornecer uma tensão variável entre zero e cinco volts, desta forma, para utilizar-se de sinal proveniente do Arduino para equipamentos com maior tensão, uma das alternativas é a utilização de uma fonte de energia externa e um driver de modulação por largura de pulsos (*Pulse With Modulation*, PWM, na sigla em inglês), isso faz com que a energia fornecida pela fonte externa seja modulada, fazendo a tensão variar entre zero e a tensão máxima da fonte externa. Esta configuração foi necessária para realizar o presente trabalho, esquematizado na Figura 1.

Para implementar o controle PID em um Arduino, é possível utilizar diversas bibliotecas disponíveis gratuitamente, como a *PID Library*, a *PID_v1* ou a *PID_AutoTune*. Essas bibliotecas permitem que o usuário defina os parâmetros do controlador PID, como o valor do *setpoint*, a taxa de amostragem, o ganho proporcional, o ganho integral e o ganho derivativo.

Para utilizar bibliotecas PID no Arduino, é necessário primeiro instalar a biblioteca escolhida de acordo com o objetivo desejado. Em seguida, é preciso criar um objeto PID e definir os parâmetros necessários, como o valor do *setpoint* e as constantes de ganho. É possível então adicionar o objeto PID ao loop do Arduino e calcular a saída do controlador. Uma vez que o controlador PID está funcionando, é possível ajustar seus parâmetros para obter um desempenho ideal. Isso pode ser feito por meio de experimentação, utilizando algoritmos de auto sintonia externos, como o método Ziegler-Nichols, ou ainda por bibliotecas que realizam a sintonia automaticamente por meio de algoritmos de auto sintonia.

Desta forma, o objetivo do presente trabalho é verificar a possibilidade de controle da temperatura de saída do casco de um trocador casco e tubos pela alteração da tensão de uma bomba de água acoplada a um driver PWM, responsável pela vazão de fluido no casco, com o uso de um controlador de baixo custo desenvolvido em Arduino.

METODOLOGIA

O desenvolvimento do trabalho se deu por meio da utilização de um módulo educacional de trocador de calor casco e tubos da empresa ECO Educacional® em consonância com os componentes instalados no Arduino.

ARDUINO

A Tabela 1 apresenta os componentes eletrônicos utilizados para o controle de temperatura e seus valores unitários médios em abril de 2023, exceto os cabos de ligação utilizados, com um valor total de R$ 237,30.

Tabela 1 - Lista dos componentes eletrônicos utilizados no circuito com Arduino

Símbolo	Quantidade (unidades)	Componente	Valor unitário (R$)
U1	1	Arduino Uno R3	70,50
M1	1	Minibomba de água 12V RS385	27,50
D1	1	Driver controlador PWM para motor DC (IRF520 MOSFET)	14,00
P1	1	Protoboard pequena	11,90
F1	1	Fonte de alimentação 5A (12V)	22,00
R1, R2, R3 e R4	4	Resistores de 1 kΩ	0,10
T1, T2, T3 e T4	4	Sensores de temperatura DS18B20	14,90
V1	1	Sensor de vazão de água YF-S201	31,40
		Total	**237,30**

Fonte: Elaborado pelos autores (2023).

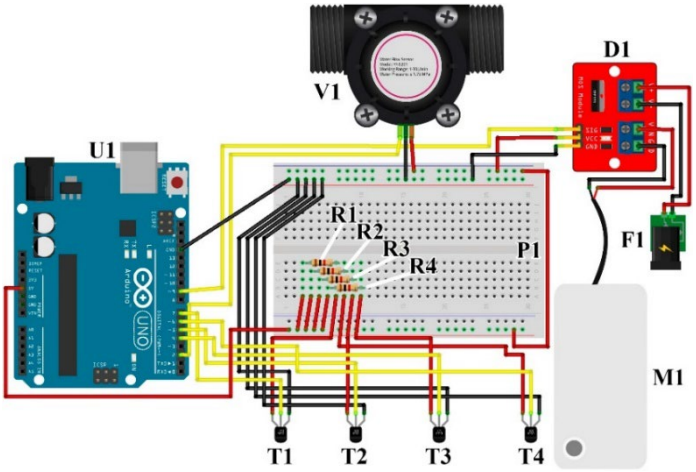
Os sensores de temperatura T1 e T2 foram instalados na entrada e saída do casco, respectivamente, enquanto os sensores T3 e T4 foram instalados na entrada e saída dos tubos. Todos os sensores de temperatura contavam com um resistor R1, R2, R3 ou R4, respectivamente.

O sensor de vazão de água V1 está localizado na saída dos tubos e, portanto, realiza a medição da vazão de água da rede.

A bomba de água M1 (ligada a fonte de alimentação F1) envia água para o casco, e é controlada pelo PWM conectado ao Arduino, de acordo com a medição T2, por meio de controle PID.

Utilizando os componentes listados na Tabela 1, foi possível representar em um diagrama as ligações realizadas, como apresentado na Figura 1, no software Fritzing®.

Figura 1 - Diagrama do circuito elétrico dos componentes conectados ao Arduino.



Fonte: Elaborado pelos autores (2023).

Para a operação do sistema:

- Verificou-se que o controle era reverso, uma vez que com o aumento da vazão na bomba (saída maior do sinal PWM, variável manipulada) a temperatura (T2 - variável controlada) do sistema diminuía e vice-versa;
- A tensão mínima de operação para a bomba (M1) foi verificada como sendo de 5,6 V, sendo a saída mínima correspondente fixada na programação do Arduino no algoritmo PID, como um limite mínimo de operação da ação de controle;
- O *setpoint* deveria se encontrar dentro dos limites de operação da bomba (M1), dessa forma, primeiramente obteve-se a temperatura de saída do casco (T2) nas voltagens máxima (12,0 V) e mínima (5,6 V) da bomba, para as vazões de água da rede no máximo e no mínimo;
- A saída do controle PID deveria se encontrar dentro dos limites de operação da bomba (M1).
- O método de sintonia utilizado foi experimental, por tentativa e erro.

O *setpoint* para a temperatura T2 foi de 34,5 °C.

MÓDULO DE TROCADOR DE CALOR CASCO E TUBOS

O módulo conta, originalmente, com uma entrada de água e duas saídas. Foi necessário, para que a vazão de entrada do casco fosse controlada e a vazão de entrada dos fluidos monitorada, que as entradas fossem separadas. Dessa forma, o módulo contava com:

1. Entrada de água ligada à rede de abastecimento da universidade, que passava pelo aquecedor a gás e posteriormente passava pelos tubos do trocador de calor;
2. Entrada de água ligada à minibomba M1 que retirava a água de um reservatório e bombeava a água para o casco conforme voltagem de saída do Arduino.

Os sensores de temperatura que o módulo originalmente contava foram desconectados para a conexão de quatro sensores de temperatura do tipo sonda (T1, T2, T3 e T4), que foram monitorados pelo Arduino.

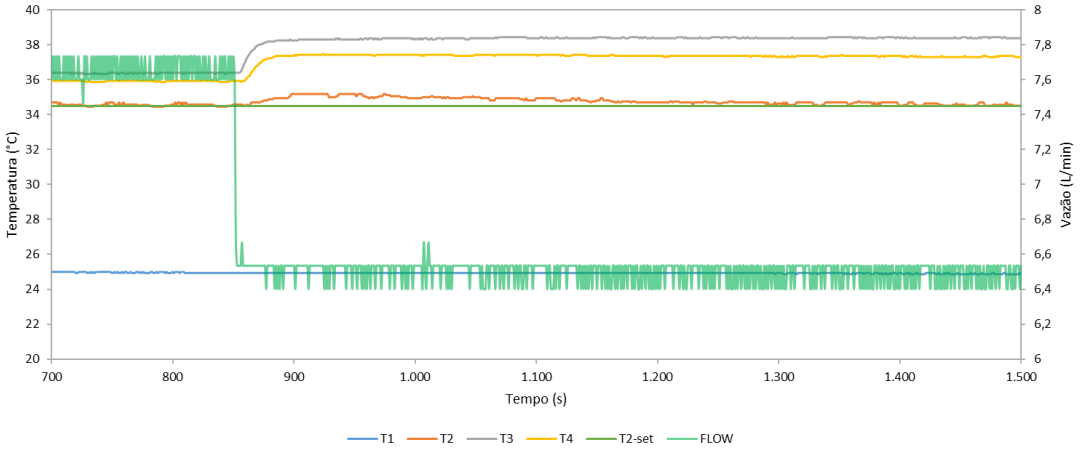
Ligando a chave geral do módulo, a bomba de vácuo para isolamento do sistema foi ligada. Efetuou-se a abertura e fechamento das válvulas de bloqueio definindo o arranjo dos fluxos para contracorrente. Seguido, respectivamente, pela abertura da torneira de água da rede, as válvulas da linha de gás e a válvula de regulagem de água quente. O aquecedor de gás liga-se automaticamente quando uma determinada vazão de água passa por ele. Regulou-se a vazão de água quente, nos tubos, pelo valor apresentado pelo sensor de vazão V1, sendo mantida em aproximadamente 7,68 L/min, com oscilações inerentes à rede compartilhada.

Foram tomadas as devidas precauções para que a temperatura não ultrapassasse 65°C, regulando a vazão de gás do aquecedor no mínimo. Após determinado tempo, foi aplicado um degrau, diminuindo a vazão de água da rede para aproximadamente 6,48 L/min.

RESULTADOS E DISCUSSÃO

Com o monitoramento das temperaturas de entrada e saída do casco (T1 e T2, respectivamente) e dos tubos (T3 e T4, respectivamente) juntamente com a vazão dos tubos (FLOW), em função do tempo, foi possível obter a Figura 2, que mostra o comportamento da temperatura com a aplicação do degrau na vazão dos tubos.

Figura 2 - Relação das temperaturas de entrada e saída do casco e dos tubos e vazão dos tubos em função do tempo



Fonte: Elaborado pelos autores (2023).

É possível notar que com a aplicação do degrau negativo na vazão dos tubos, a temperatura de entrada dos tubos (T3) aumenta de 36,38 °C para 38,31 °C, já que o calor fornecido continua constante. A temperatura de saída do casco (T4) também aumenta de 35,94 °C para 37,44 °C, já que a troca térmica realizada não busca mantê-la constante pelo controle PID. A temperatura de entrada do casco (T1) se mantém constante em aproximadamente 24,94 °C, já a temperatura de saída do casco (T2) flutua de maneira irregular por conta da atuação do controle PID, porém, é possível verificar que a temperatura busca se manter no *setpoint* (T2-set) de 34,5 °C.

A vazão da água da rede possui uma variação bastante alta e com bastante ruído, porém, é possível verificar uma média do valor dentro do esperado em 7,68 L/min e, após aplicação do degrau, 6,48 L/min.

Verifica-se ainda que o tempo de resposta do sistema é bastante lento, isso se deve, principalmente, a dois fatores: o primeiro deles é o tempo morto inerente do processo devido à baixa vazão da bomba de água (M1), que atinge um máximo de 2 L/min e, portanto, não consegue trabalhar com um controle que possua uma faixa de trabalho ampla, fato que poderia ter sido resolvido utilizando-se uma bomba de maior potência ou ainda duas bombas em paralelo, o que aumentaria a vazão. O segundo fator é a sintonia ter sido realizada apenas de maneira experimental, o que poderia ter sido solucionado pela utilização de um método de auto sintonia dentro do próprio Arduino.

CONCLUSÃO

Portanto, foi possível realizar uma rotina de controle de baixo custo para o trocador de calor casco e tubo de forma efetiva utilizando Arduino. Tal resultado se mostra positivo ao se observar a necessidade de utilização de controle em projetos de pesquisa, trazendo uma possibilidade eficiente e de baixo custo, além disso, a aplicação desse projeto pode

trazer ganhos consideráveis na academia, visto que os discentes podem obter uma visualização do funcionamento de um controle PID sem que seja necessário para isso a aquisição de um CLP, por exemplo.

Um melhor resultado poderia ser obtido com uma bomba de maior capacidade, com uma bomba dimensionada adequadamente para funcionar no intervalo de vazão de operação, redução dos pontos de perda de carga do sistema ou utilização de um método de sintonia que não fosse o método de tentativa e erro, experimental. Outra sugestão é a utilização de uma válvula proporcional após a bomba, utilizando a abertura da válvula como variável manipulada no lugar da rotação da bomba, tendo como desvantagem o custo, geralmente, elevado desse tipo de válvula. Para futuros trabalhos, é proposto que seja verificada a possibilidade de simulação do processo, para comparação com os resultados obtidos experimentalmente, além da obtenção e avaliação do tempo morto do sistema.

Agradecimentos

Agradecimentos à Universidade Tecnológica Federal do Paraná por propiciar ambiente adequado para a realização do trabalho.

Disponibilidade de código

O repositório no GitHub com o código utilizado no trabalho pode ser encontrado em: https://github.com/vrauldias/arduino_control.

Conflito de interesse

Não há conflito de interesse.

REFERÊNCIAS

BANZI, M. **Getting Started with Arduino**. [S. l.]: O'Reilly Media, Inc., 2009.

BARGMAN, T. L.; LAVINE, A. S. **Fundamental of Heat and Mass Transfer**. 8. ed. [S. l.]: John Wiley & Sons, Inc., 2017. 1046 p.

JAMIESON, P.; HERDTNER, J. More missing the Boat: arduino, raspberry pi, and small prototyping boards and engineering education needs them. **2015 IEEE Frontiers In Education Conference (FIE)**, El Paso, TX, USA, p. 1-6, out. 2015. Disponível em: http://dx.doi.org/10.1109/fie.2015.7344259. Acesso em: 17 abr. 2023.

KHERKHAR, A. *et al*. Thermal investigation of a thermoelectric cooler based on Arduino and PID control approach. **Case Studies In Thermal Engineering**, [S.l.], v. 36, p. 102249, ago. 2022. Disponível em: http://dx.doi.org/10.1016/j.csite.2022.102249. Acesso em: 17 abr. 2023.

ROHANI, S. S. (ed.). **Coulson and Richardson's Chemical Engineering**: Volume 3B: Process Control. 4. ed. [S. l.]: Elsevier Ltd., 2017. 630 p.

SEBORG, D. E. *et al*. **Process Dynamics and Control**. 4. ed. [S. l.]: John Wiley & Sons, Inc., 2017. 515 p.