CONGRESSO BRASILEIRO
DE ENGENHARIA QUÍMICA EM
INICIAÇÃO CIENTÍFICA
21-24 Julho de 2019
Uberlândia/MG

DESENVOLVIMENTO DE KITS EXPERIMENTAIS DE BAIXO CUSTO UTILIZANDO PLATAFORMA LIVRE PARA O ENSINO E PESQUISA DE CONTROLE DE PROCESSOS

B. C. M. HENRIQUE[1], M. A. VILELA[2], L. C. M. HENRIQUE[3] e H. M. HENRIQUE[2]

[1] Universidade Federal de Uberlândia, Faculdade de Engenharia Elétrica
[2] Universidade Federal de Uberlândia, Faculdade de Engenharia Química
[3] Universidade Federal de Uberlândia, Faculdade de Engenharia Mecânica
E-mail para contato: humberto@ufu.br

RESUMO – O presente trabalho apresenta a implementação de dois *kits* experimentais para laboratórios de controle e automação utilizando *hardware* e *softwares* livres e de baixo custo. A plataforma *Visual Studio Code* (livre) foi utilizada para desenvolver os códigos fontes e interfaces dos *kits* e a plataforma de prototipagem eletrônica *open source* Arduino (livre) foi utilizada para desenvolver o *hardware* das montagens experimentais. Além de abordar as questões teóricas e conceituais do Arduino, o presente trabalho apresenta descritivos das montagens com foco no desenvolvimento de controladores reais. Os resultados demonstraram que a plataforma desenvolvida se mostrou adequada para o uso em *kits* experimentais permitindo ao usuário confrontar suas ideias e expectativas com as evidências experimentais. Mostrou-se ainda que a instrumentação é de fácil configuração, com baixo nível de ruído e custo baixo, o que torna viável a construção de *kits* desta natureza para fins didáticos e de pesquisa.

1. INTRODUÇÃO

O ensino e pesquisa de controle de processos na academia está baseada em demasia em pacotes de simulação disponíveis e em laboratórios virtuais, ambos com sua posição insubstituível no processo de ensino e pesquisa. Mas, infelizmente, os experimentos com malhas de controle são muitas vezes limitados a esses domínios virtuais e os alunos não têm o *feedback* do mundo físico e real sobre o impacto dos algoritmos de controle e seus parâmetros. As razões para isso residem no fato do elevado custo do *hardware* e *software* necessários para implementar algoritmos de controle fisicamente e em tempo real. Este trabalho tem como objetivo desenvolver uma plataforma experimental de baixo custo, direta e surpreendentemente poderosa para a implementação de algoritmos de controle em tempo real. A plataforma consiste em placas Arduino e um microcomputador de baixo custo rodando sob plataforma Windows™ para interfacear com os sensores de baixo custo compatíveis com a plataforma Arduino e com a planta experimental real. A placa Arduino é usada para interação com o mundo físico através de suas entradas e saídas. Os algoritmos de controle foram implementados usando o *software open-source Visual Studio Code*, que permitem que alunos desenvolvam seus algoritmos de controle sem custo adicional. O ensino e a pesquisa em controle e automação em cursos de graduação e pós-graduação no Brasil são muito caros por envolver o uso de *hardware* e *softwares* proprietários de elevado custo. Por causa disso,

cursos de controle e automação são relegados à abordagem teórica, deixando o aluno sem a vivência prática dessas tecnologias. O trabalho tem por objetivo desenvolver *kits* experimentais de baixo custo para uso em laboratórios de automação e controle de processos para que possam ser usados em atividades de ensino e pesquisa nas disciplinas associadas ao tema.

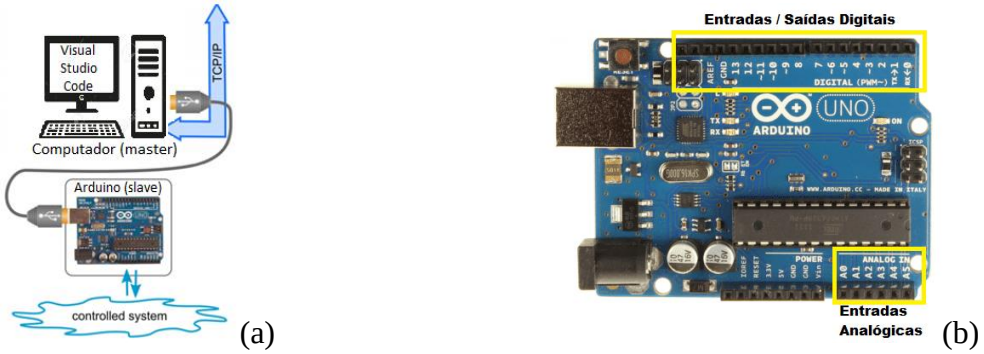
2. A PLATAFORMA DESENVOLVIDA

O microcomputador no qual a plataforma foi baseada forma par perfeito para a educação e pesquisa de controle e automação, combinando o poder computacional da plataforma *VSCode* e a capacidade de entrada-saída das placas Arduino. Parte integrante da plataforma apresentada é o *software* que transforma essas placas em um controlador industrial programável, pois permite que os alunos/usuários criem algoritmos de controle usando as mesmas ferramentas de *software* e conceitos de fluxo de trabalho que são usados no desenvolvimento de algoritmos de controladores industriais. A plataforma de controle desenvolvida consistiu de placa microcontroladora Arduino UNO R3, um microcomputador Dell 8ª Geração Intel Core i5 8GB RAM 500 GB com Windows 10 e o pacote *Visual Studio Code* que é um editor de código-fonte desenvolvido pela Microsoft para Windows, Linux e macOS. Os componentes individuais são discutidos nas próximas seções.

2.1. A placa Arduino

O Arduino (disponível em: <https://code.visualstudio.com/>, 2019) é uma plataforma de prototipagem eletrônica de código aberto bem conhecida que consiste de uma unidade de controle central de muitas aplicações de controle incorporadas, desde roupas interativas até robótica e impressoras 3D. No presente trabalho a placa foi programada para atuar como escravo (*slave*) e suas entradas e saídas foram usadas para interação com o mundo físico (Figura 1a). Neste trabalho usou-se as placas Arduino UNO R3 para a unidade de controle de vazão e para a unidade de controle de nível. Esta placa traz em seu núcleo funções para leitura de sinais analógicos através da utilização do conversor analógico digital. A placa Arduino UNO possui pinos de entrada e saídas digitais, assim como pinos de entradas e saídas analógicas. Essa placa possui 14 pinos que podem ser usados como entradas ou saídas digitais (Figura 1b). Alguns desse pinos podem ser usados como saídas PWM de 8 bit. A placa possui ainda 6 entradas com resolução de 10 bits. O *clock* do conversor AD do Atmega328 permite a aquisição de 15400 amostras por segundo.

Figura 1 – A plataforma de ensino e pesquisa Arduino/*Visual Studio Code*. (a). Esquema de ligação. (b). Detalhes da Arduino UNO R3.



2.2. O *Visual Studio Code* (*VSCode*)

O *VSCode* (disponível em: < https://code.visualstudio.com/>, 2019) é um programa gratuito de código aberto desenvolvido pela Microsoft com suporte para Windows, macOS e Linux. É altamente customizável, permitindo que os usuários mudem o tema do editor, as teclas de atalho, entre outros aspectos. Ele é mais vantajoso em relação à IDE do Arduino principalmente nos aspectos gráfico e de depuração de erros (*debugging*). O *VSCode* possui uma extensão chamada PlatformIO IDE que é um sistema de código aberto para desenvolvimento em IoT (*Internet of Things*). É escrito em Python e não necessita de nenhuma biblioteca ou ferramenta adicional. Adicionalmente, o editor possui suporte à sintaxe de diversas linguagens como Python, Ruby, C, C^{++}, Visual Basic, C Sharp e F Sharp.

2.3 Controle em tempo real

A última etapa da criação de um sistema de controle em tempo real foi a implantação do algoritmo de controle na plataforma de destino. Os algoritmos de controle tipo PID (Seborg, 2004) foram implementados em linguagem C^{++} dentro do *VSCode* com saídas gráficas e exportação dos dados referentes às variáveis controladas, manipuladas e *setpoints*. O protocolo de comunicação usado entre o *master* e o *slave* foi o padrão serial utilizando um cabo USB em modo assíncrono com *baud rate* de 115200 bps, 8 bits de dados sem paridade e 1 bit de parada. O programa já implementando a estratégia de controle foi então compilado pelo compilador C^{++} do *VSCode* e foi carregado na memória *flash* do microcontrolador Arduino via cabo USB.

3. AS PLANTAS EXPERIMENTAIS

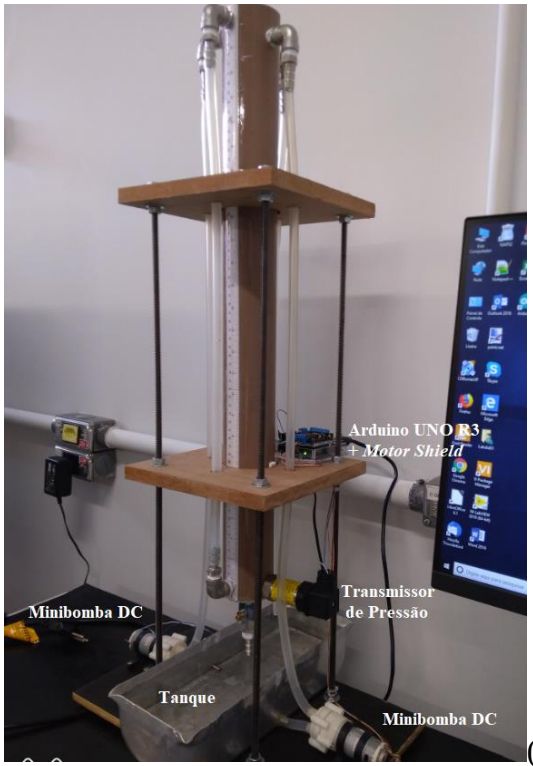
Duas plantas experimentais de bancada foram construídas e instrumentadas com sensores de baixo custo. A primeira planta é de controle de vazão e a segunda de nível. As plantas são detalhadas a seguir.

3.1 Sistema de controle de vazão

O primeiro sistema experimental construído consta de um pequeno tanque de um litro de capacidade, de onde é bombeado água por um sensor de vazão de efeito Hall (modelo YF-S402 de 0,5-5,0 L/min). Este sensor gera como saída um sinal de tensão na forma de uma onda quadrada de 0 a 5V com frequência variável em função da vazão medida. Este sinal, que é um sinal digital, é então adquirido por um dos canais digitais da Arduino UNO e o período da onda é medido usando-se a função *pulseIn()* de biblioteca do *VSCode*. Esse sensor foi previamente calibrado usando uma balança e cronômetro. Duas minibombas centrífugas (modelo RS-385 DC 12V e 0-2 L/min) em paralelo são utilizadas para impulsionar o líquido por um sensor de vazão. A vazão das bombas é manipulada alterando a tensão nos terminais da bomba de 0 a 12 V utilizando de um *motor shield* para Arduino (modelo L293D Driver Ponte H) que controla até 4 motores DC. Este *shield* recebe sinal PWM da placa Arduino. Dois pinos PWM do Arduino UNO são, então, usados para manipular a vazão das bombas utilizando a função de biblioteca *analogWrite()* do *VSCode*. O líquido bombeado fica em circuito fechado evitando o desperdício de água durante o funcionamento da unidade experimental. A Figura 2a mostra uma foto da unidade.

Figura 2 – Montagens experimentais com detalhes da instrumentação. (a). Controle de vazão. (b). Controle de nível.

(a)



(b)

3.2 Sistema de controle de nível

O segundo sistema é um tanque cilíndrico de PVC de 65 cm de altura e 50,8 mm de diâmetro, que é alimentado no topo por duas correntes de líquido provenientes de duas minibombas. A vazão das bombas é manipulada alterando a tensão nos terminais das mesmas de 0 a 12 V utilizando um *motor shield* (modelo L293D Driver Ponte H). Uma válvula manual tipo esfera de ¼" foi instalada no fundo do tanque. Na base do tanque foi instalado um transmissor de pressão diferencial (modelo JF302 piezoresistivo 10kPa/0.1bar gauge). O sinal de saída deste transmissor é um sinal de 4-20 mA que é enviado para um conversor corrente/tensão (4-20ma para 0-5v), uma vez que a placa Arduino UNO somente recebe sinal analógico de tensão na faixa de 0-5V. Assim, é possível adquirir um sinal analógico em um dos pinos de entrada do Arduino UNO que é proporcional ao nível de líquido dentro do tanque. A Figura 2b mostra uma foto da unidade.

4. RESULTADOS

4.1. Identificação dos sistemas

Nas montagens construídas as entradas do sistema dinâmico são sinais com resolução de 8 bits (0-255) que geram sinais PWM com *duty cicle* de 0 a 100% em pinos específicos para saídas PWM na placa Arduino UNO R3. Os sinais PWM são, então, convertidos pelo *motor shield* em sinais analógicos de tensão de 0 a 12VDC que alimentam as minibombas DC. Dessa forma as vazões das minibombas são manipuladas. As saídas são a vazão medida pelo sensor de efeito Hall na primeira montagem e o nível medido pelo transmissor de pressão

diferencial na segunda montagem, ambos devida e previamente calibrados. Para obtenção das funções de transferência dos sistemas uma corrida em malha aberta foi executada variando-se os sinais enviados aos pinos PWM entre os valores de 90 a 255, correspondendo às vazões mínima e máxima obtidas nas minibombas. Os dados dessas corridas foram aqui omitidos por falta de espaço. Com os dados entrada *versus* saída de cada planta em mãos, funções de transferência tipo *first order plus time delay* (FOPDT) foram numericamente ajustadas aos dados (Equações (1) e (2)):

$$Q'(s) = \frac{K}{\tau s + 1} U'(s) \quad (1)$$

(K = 8,57 $cm^3 min^{-1}$ e τ = 2,92 s)

$$H'(s) = \frac{K_1}{\tau_1 s + 1} e^{-\theta_1 s} U_1(s) + \frac{K_2}{\tau_2 s + 1} e^{-\theta_2 s} U_2(s) \quad (2)$$

(K_1 = 0,378 cm; τ_1 = 44,76 s; K_2 = 0,414 cm, τ_2 = 47,10 s e $\theta_1 = \theta_2$ = 10 s)

4.2. Resposta em malha fechada

De posse das funções de transferências ajustou-se controladores tipo PI para ambas as plantas usando o método IMC (Rivera *et. al.*, 1986) para a planta de vazão e o método ITAE *Load* e *Setpoint* (Seborg *et. al.*, 2004) para a planta de nível. A Tabela 1 mostra os parâmetros ajustados para os controladores. Para testar o desempenho real do controlador de vazão foi introduzida uma sequência de modificações no *setpoint* da vazão. Os resultados são mostrados na Figura 3a, onde observa-se a performance bastante adequada do controlador PI, sem introduzir oscilações e sem *offset* na variável controlada. Já a Figura 3b revela a performance de um controlador P, utilizando o mesmo K_C do controlador PI, porém, com τ_I infinito. Observa-se com clareza a presença de *offset* conforme prevê a teoria. Observa-se também dessas Figuras que as ações de controle (variável manipulada) sofreram grandes variações no momento da introdução dos degraus no *setpoint*. Essas grandes variações podem ser suavizadas ajustando o controlador de maneira mais conservadora. Contudo, haverá uma degradação na performance da malha fechada.

Figura 3 – Desempenho experimental da malha de controle de vazão frente a mudanças no *setpoint*. (a). Controlador PI. (b). Controlador P.

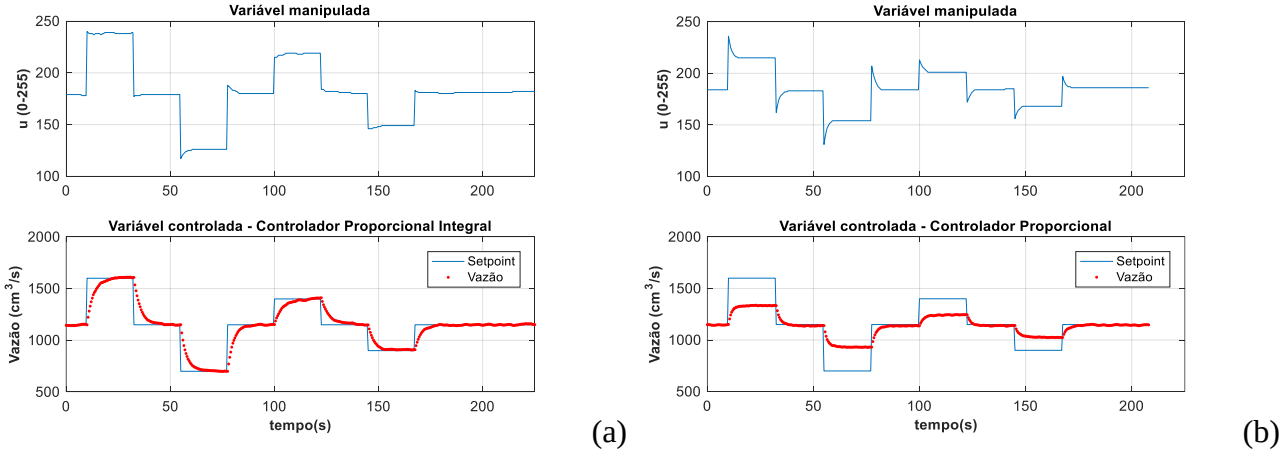
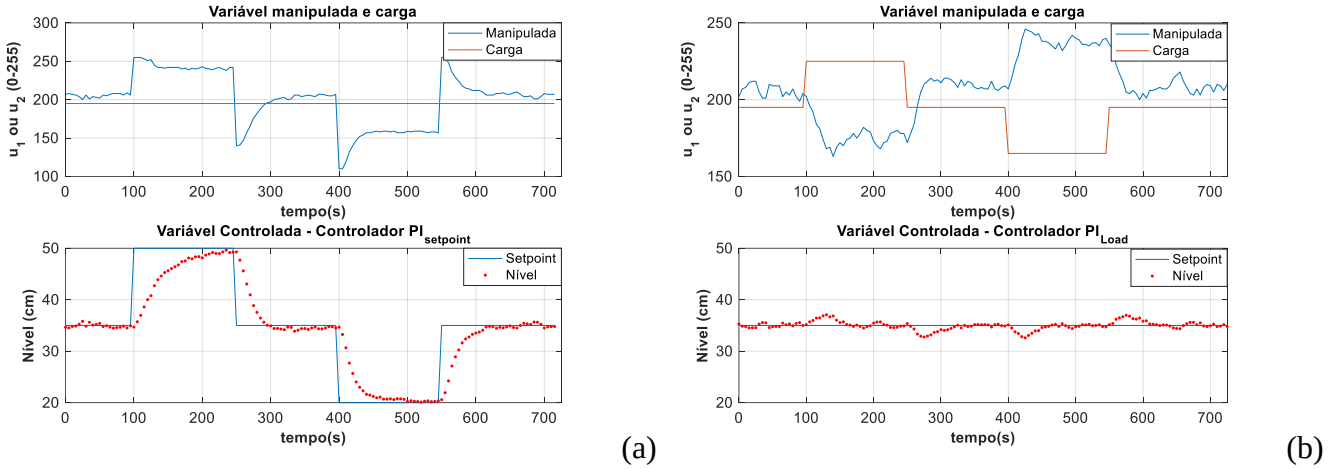


Figura 4 – Desempenho experimental da malha de controle de nível frente a mudanças no: (a). *Setpoint* e controlador $PI_{Setpoint}$. (b). Carga e controlador PI_{Load}.



(a) (b)

Tabela 1 – Parâmetros de sintonia dos controladores PI desenvolvidos.

Problema	Método	K_C	τ_I
Vazão (*setpoint*)	IMC	0,117 cm^{-3}.min	2,92 s
Nível (*setpoint*)	ITAE *setpoint*	6,124 cm^{-1}	45,07 s
Nível (*Load*)	ITAE *Load*	9,428 cm^{-1}	24,36 s

Neste caso também foi introduzida uma sequência de modificações no *setpoint*. Os resultados são mostrados na Figura 4a, onde observa-se a performance bastante adequada do controlador $PI_{Setpoint}$, sem introduzir oscilações e sem *offset* na variável controlada. Já a Figura 4b mostra os resultados do controlador PI_{Load} frente a perturbações na carga. Observa-se que o controlador conseguiu rejeitar as perturbações na carga sem degradar a performance da variável controlada, isto é, sem oscilações e sem *offset*.

4.3. Custo da Instrumentação das Unidades Experimentais

A Tabela 2 e 3 traz os custos associados com a instrumentação das unidades. Chama muito a atenção o baixo custo dos instrumentos e da placa Arduino UNO.

Tabela 2 – Custo da instrumentação da unidade de controle de vazão.

Item	**Descrição**	**Quant.**	**Custo (Us$)***	**Total**
Minibomba	Modelo RS-385 DC 9 a 15V e 0-2 L/min	02	11,29	22,58
Sensor Hall	Medidor de vazão modelo YF S402 de efeito Hall e 0,5-5,0 L/min	01	9,96	9,96
Ponte H 293D	Motor Shield Arduino L293D Driver Ponte H	01	1,49	1,49
Arduino UNO R3	Placa com microcontrolador Atmega328, 14 entradas/saídas digitais (6 são saídas PWM), 6 entradas analógicas.	01	6,50	6,5
VSCode	Programa *open-source* com suporte para Windows, macOS e Linux.	01	0,0	0,0

*. *Custo FOB, não inclui frete até o destino.* **Total =Us$40,53**

Tabela 3 – Custo da instrumentação da unidade de controle de nível.

Item	Descrição	Quant.	Custo (Us$)*	Total
Minibomba	Modelo RS-385 DC 9 a 15V e 0-2 L/min	02	11,29	22,58
Sensor pressão diferencial	Modelo JF302 de 4-20mA, 12-36 V de 10kpa/0.1bar gauge e aço inoxidável 316.	01	32,00	32,00
Conversor 4-20mA	Modulo Conversor 4-20ma para 0-5V.	01	12,86	12,86
Ponte H 293D	Motor Shield Arduino L293D Driver H	01	1,49	1,49
Arduino UNO R3	Placa com microcontrolador Atmega328, 14 entradas/saídas digitais (6 são saídas PWM), 6 entradas analógicas.	01	6,50	6,5
VSCode	Programa *open-source* com suporte para Windows, macOS e Linux.	01	0,0	0,0

*. *Custo FOB, não inclui frete até o destino.* **Total =Us$75,43**

5. CONCLUSÃO

Os resultados demonstraram que a plataforma Arduino juntamente com o *software VSCode* se mostram eficientes para o uso em *kits* experimentais de controle de processos. A instrumentação é de fácil configuração, com baixo nível de ruído e custo baixo, o que torna viável a construção de *kits* desta natureza para fins didáticos e de pesquisa em ambiente universitário.

Do ponto de vista pedagógico, a vantagem dessa abordagem de ensino e pesquisa é que o aluno confronta as suas ideias e expectativas com as evidências experimentais, tornando-se gradualmente competente no processo de coordenação das teorias de controle e automação com as evidências experimentais que produz. Assim, essa plataforma de ensino e pesquisa renova os papéis dos alunos e dos professores envolvidos com o ensino e a pesquisa de controle e automação de processos.

6. REFERÊNCIAS

_____, Microsoft Visual Studio Code, 2019. Disponível em: < https://code.visualstudio.com/>. Acesso em 15 de abril de 2019.

_____, Arduino, 2019. Disponível em: <https://www.arduino.cc/>. Acesso em 15 de abril de 2019.

Rivera, D. E.; Morari, M.; Skogestad, S. Internal model control: PID controller design. *Industrial & Engineering Chemistry Process Design and Development*,1986 *25* (1), p. 252-265

Seborg, D. E.; Edgar, T. F.; Mellichamp, D. A. *Process Dynamics and Control*, Wiley, New York, 2004.