Uso do Arduino com Matlab em Laboratório de Controle

O Arduino é utilizado nas aulas conforme ilustrado na figura 1. A comunicação entre o Matlab e o Arduino é feito via porta USB emulando uma porta serial RS232. O Matlab envia comandos que são interpretados por um programa em execução no Arduíno. Ele executa os pedidos e retorna ao Matlab as respostas.

Com este ambiente, pode-se aplicar um sinal médio no circuito RC e medir a tensão sobre o capacitor ou aplicar uma tensão PWM no motor CC e medir sua velocidade.

Os parâmetros passados para o Arduíno nos comandos são:

P1: função desejada:

0 = aplicar sinal U_0, para definir um ponto de operação

1 = aplicar sinal degrau U_0 em malha aberta e coletar a resposta,

2 = aplicar uma referência Ref e coletar a resposta em malha fechada com um controlador especificado

P2: define uso do circuito RC(1) ou Motor(0)

P3: Tempo de amostragem Ts (em ms)

P4: Tempo durante o qual será coletado o sinal (inteiro, em segundos)

P5: Referência ou U0 aplicados (número inteiro)

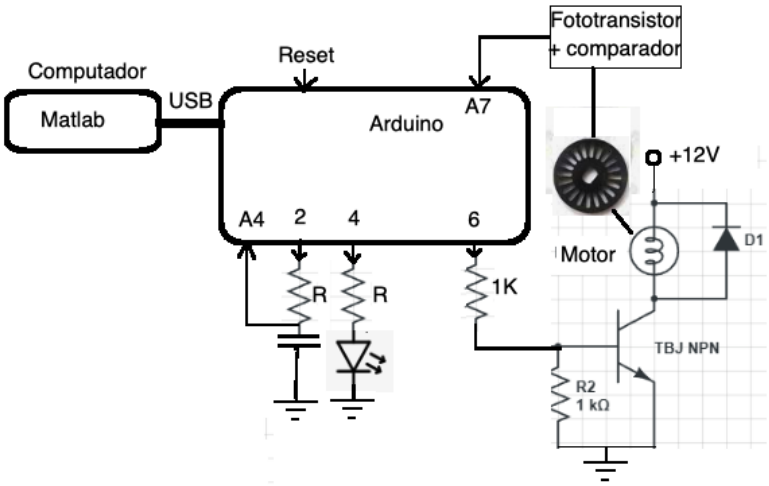
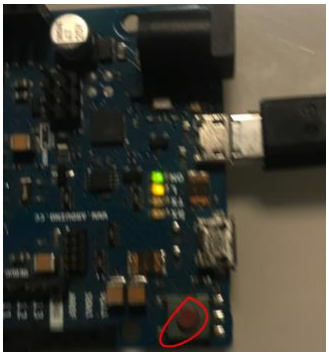


Figura 1. Interação Matlab, Arduino, plantas



O botão de reset no Arduíno pode ser usado quando houver mensagem de erro na comunicação usando a porta serial.

A malha de controle correspondente à Figura é vista na figura 2. O sinal aplicado pelo controlador é sempre um sinal PWM, que varia de 0-255 (8 bits), sendo convertido em tensão média no circuito RC ou no motor CC.

Circuito RC:

O sinal PWM varia de 0-255 sendo convertido em uma tensão média de 0-5V, que é a tensão de alimentação do Arduíno. Esta tensão média que aparece no capacitor é medida por um conversor A/D de 10 bits, logo, varia de 0-1023. Em malha fechada, o objetivo é obter uma tensão desejada Ref sobre o capacitor usando um controlador.

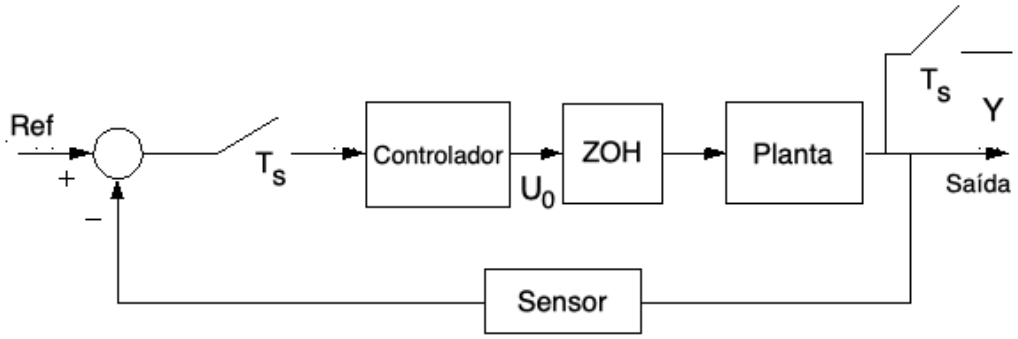


Figura 2. Malhas de controle

Na figura 3 se observa a resposta de primeira ordem coletada durante 3 segundos resultante da aplicação de um degrau de amplitude U_0=100 (PWM). O valor máximo em regime foi quase 400, lembrando que ele varia de 0 a 1023.

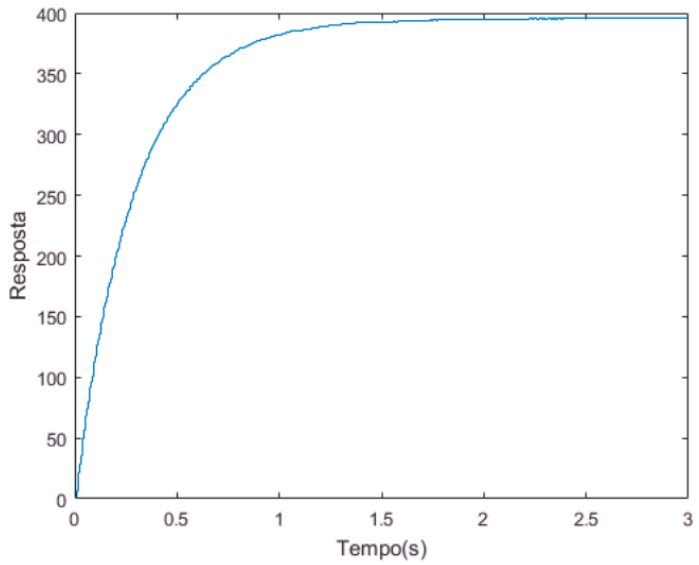


Figura 3. Exemplo de coleta de dados do circuito RC para resposta ao degrau U_0.

Motor CC:

O sinal PWM do Arduíno é aplicado a um transistor de potência alimentado em 12V. Assim, o motor recebe uma tensão média que varia de 0 a 12V quando o sinal PWM varia de 0-255. A medição de velocidade é feita usando um disco com 12 ranhuras, gerando portanto, 12 pulsos por rotação. Para medir a velocidade, mede-se o tempo entre pulsos. Em 6000rpm, tem-se 100rps. Considerando os 12 pulsos por rotação, tem-se 1200 pulsos por rotação, o que dá um pulso a cada 0.83ms. Na figura 4 superior se observa o tempo necessário para medir um pulso em diferentes velocidades. Para 500rpm, mais que 10ms são necessários para medir um pulso, indicando que um tempo de amostragem de 10ms não seria adequado nesta velocidade.

O timer do Arduíno mede com resolução de micro segundos. Para medir a resolução em RPM, varia-se 1us para ver a variação do valor de RPM discretizado, o que é mostrado na Figura 4 inferior. Observa-se que até 3000RPM a resolução mínima é em torno de 1rpm, chegando a 7 para as velocidades maiores. Portanto, em velocidades maiores as variações na medida de velocidade tendem a piorar o desempenho de uma malha de controle.

Na figura 5 observa-se a resposta do motor a uma sequência de valores PWM aplicados, de 80, 100 e 120, atingindo em regime as velocidades de 3.140, 4060 e 4750 RPM, respectivamente.

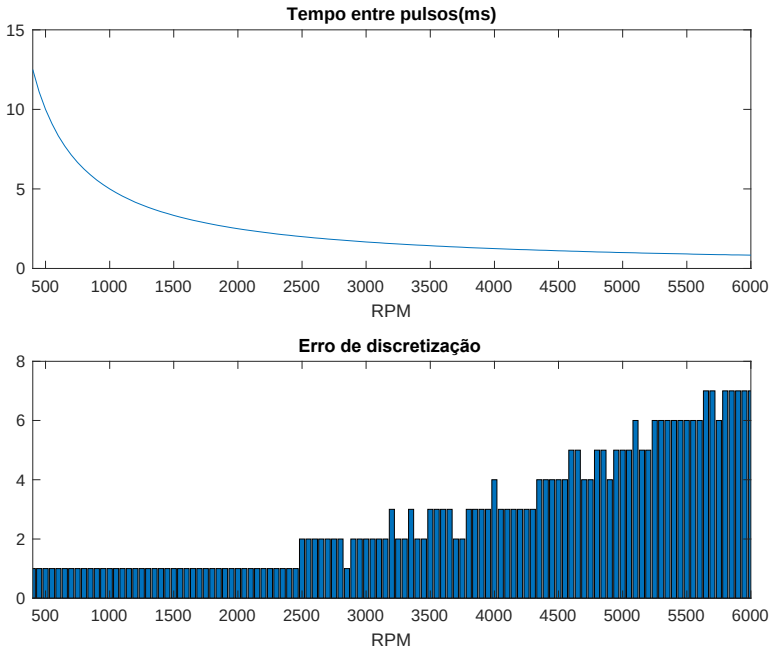


Figura 4. Medição de velocidade usando o encoder

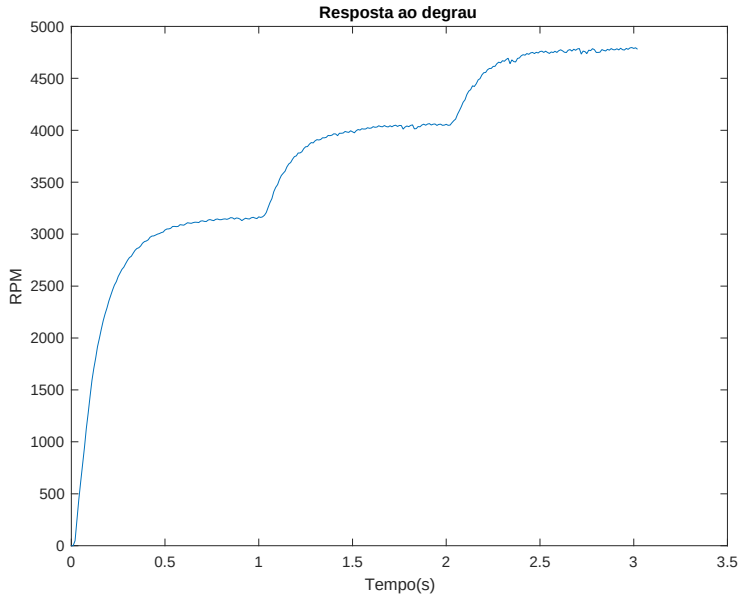


Figura 5. Respostas ao degrau do motor, para U0={80,100,120}

Controle via controlador Proporcional mais Integral (PI)

Este controlador é dado por $C(s) = K_p + \frac{K_I}{s}$ na forma paralela e $C(s) = K_p + \frac{K_p K_I}{s}$ na forma série. Como os métodos de sintonia de controladores consideram que o controlador PI está na forma série, ele foi assim implementado no Arduíno.

Caso seja projetado na forma paralela, deve-se dividir K_I por K_p antes de fornecer o ganho à função do Matlab. Devido a restrições da implementação, os ganhos K_I e K_p são multiplicados por 100 e tornados inteiros antes de serem enviados ao Arduíno, que faz o tratamento adequado antes de implementar o controle.

A implementação discreta do controlador PI é feita pela equação

$$u(k) = u(k-1) + K_p e(k) + (T_s K_I - K_p) e(k-1)$$

sendo T_s o tempo de amostragem e $e(k) = Ref - y(k)$.

Os valores de $u(k)$ são limitados a [0,255] devido ao PWM utilizado.