Projeto de Guindaste em Miniatura Automatizado

Artur Mello
Graduação em Engenharia de Sistemas
Universidade Federal de Minas Gerais
artursbm@ufmg.br

Ítalo Andrade
Graduação em Engenharia de Sistemas
Universidade Federal de Minas Gerais
italoaugustoandrade@ufmg.br

Rúbia Guerra
Graduação em Engenharia de Sistemas
Universidade Federal de Minas Gerais
rubia-rg@ufmg.br

***Abstract*—O objetivo deste trabalho é apresentar as definições de projeto de um guindaste em miniatura, feito com palitos de madeira, automatizado por meio de um Arduino. Os conceitos gerais das componentes de hardware e software do projeto, o sistema de controle e o protocolo de comunicação serial são apresentados.**

***Index Terms*—Guindaste em miniatura, Arduino, mapa conceitual, interface de usuário, sistema de controle.**

I. Introdução

Guindastes são estruturas complexas que incluem, geralmente, mastro principal, lança e contra-lança, base, estruturas de balanceamento, sistemas mecânicos e sistemas de automação.

O projeto de um guindaste em miniatura envolve diferentes componentes e fases de planejamento. Neste trabalho, serão apresentadas descrições sobre cada subsistema envolvido no projeto de um guindaste automatizado, que pode ser dividido em: interface de usuário (*GUI*), estrutura física e o sistema de controle realizado por *Arduino*. Serão abordadas as características de cada subsistema, tratando do projeto e da implementação, como foram definidas as entradas e saídas e como é feita a integração entre essas partes.

II. Materiais e Métodos

O trabalho proposto envolvia a gestão de várias etapas de um projeto de sistema complexo. Uma vez que essas etapas envolviam especificidades, fez-se necessário um acompanhamento do progresso do projeto. Assim, foi adotada a plataforma *Trello*, onde listas de tarefas foram adicionadas, e cada tarefa era assinalada para um ou mais integrantes do grupo a cada semana de trabalho.

Para o desenho inicial do projeto, foram realizados mapas mentais e conceituais do produto como um todo, utilizando o software CMaps. Para criar uma visualização inicial do projeto, foi utilizado o software SketchUp, que permite a criação de um desenho 3D da estrutura do guindaste.

O projeto e construção do guindaste foram feitos seguindo as determinações iniciais de requisitos funcionais e não funcionais, conforme descrito abaixo:

A. Requisitos

1) Materiais de Estrutura: O guindaste utilizou como material base de construção palitos de picolé, materiais de plástico para fazer o enrolamento do barbante (utilizado como corda de içamento) e também as bases de rotação dos motores para a movimentos da lança e da corda, e também abraçadeiras de nylon para melhorarem a fixação das partes mecânicas do guindaste.

2) Funcionalidades: O guindaste deveria se movimentar em ambas as direções, e também deveria içar a corda da lança. Ambas as ações são executadas pelos motores DC de 3,6V com caixa de redução de eixo duplo. Foi utilizado um Driver para motor DC do tipo ponte H, para fixar o sentido de corrente e nível de tensão para os motores. Para alimentar o sistema eletrônico, foi utilizada uma fonte de alimentação de 9V.

Para o sistema de controle, foi utilizado um Arduino Uno, com a implementação do algoritmo de controle sendo feita na linguagem C. Este sistema interagia com sensores tipo encoder, que, a partir da medição de deslocamento realizada com discos de encoder.

Todo o circuito eletrônico foi desenhado em um diagrama de circuitos utilizando a ferramenta Fritzing, e o funcionamento do algoritmo de controle do Arduino foi simulado no Tinkercad. Para validar os modelos do sistema de controle, foi utilizado o Matlab para simulações e aquisições de gráficos resultantes.

O produto final teve como objetivo integrar, também, a interação com o usuário. Foi proposta a criação de uma interface gráfica utilizando a *framework* Qt, que tem como base a linguagem de programação C++, além de serem utilizados padrões de projeto de orientação a objetos.

O projeto contou com um protocolo de comunicação entre o sistema mecânico de movimentação do guindaste e a interface gráfica. Este foi implementado também utilizando a linguagem C. Um diagrama de estados deste protocolo foi criado e desenhado utilizando a ferramenta Fizzim [1].

III. Descrição do Projeto

O guindaste em miniatura é um sistema complexo, composto de diversos sub-sistemas inter-dependentes em comunicação. Assim, o primeiro passo para sua construção é o mapeamento e o entendimento das partes envolvidas. Para tanto, foi criado o mapa de conceitos (com auxílio da ferramenta *cmapTools* [2]) da Figura 1, buscando captar, em alto nível, cada componente do guindaste e seu respectivo funcionamento.

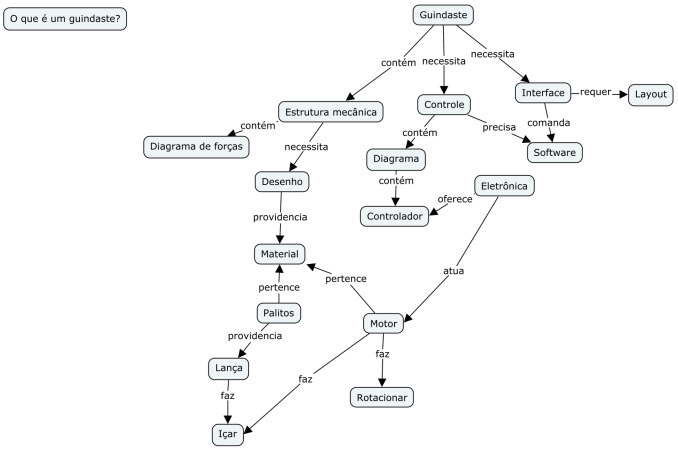

Fig. 1. Mapa conceitual geral de um guindaste

A seguir, serão discutidas as definições e implementações de cada sub-sistema mapeado: estrutura, interface de usuário, sistema de controle, hardware envolvido e protocolo de comunicação entre interface e o microcontrolador.

A. Projeto Estrutural

Um dos primeiros passos no desenvolvimento do guindaste foi o design da estrutura. O projeto deve considerar as restrições de matéria-prima para construção, de componentes, a disposição das sub-estruturas - lança, contra-lança, base e haste - e sustentação da estrutura. Para melhor visualização, conjunto de conceitos e processos para realização dessa etapa foi esquematizado no mapa mental conforme mostra a Figura 2.

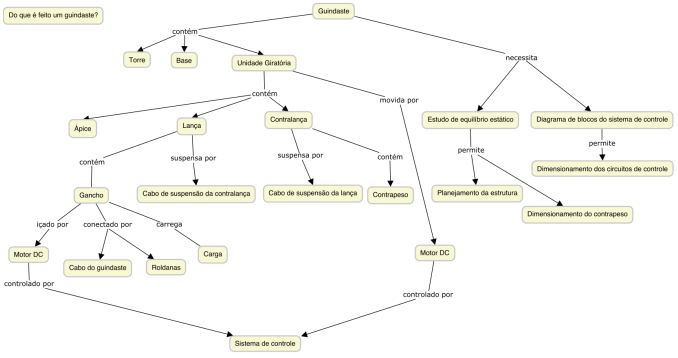

Fig. 2. Mapa conceitual sobre a parte estrutural de um guindaste

A primeira questão considerada foi a carga máxima suportada pelo guindaste, permitindo definir restrições para a sustentação da estrutura. A partir do diagrama mostrado na figura 3, foi realizado um estudo preliminar e simplificado do equilíbrio estático estrutural.

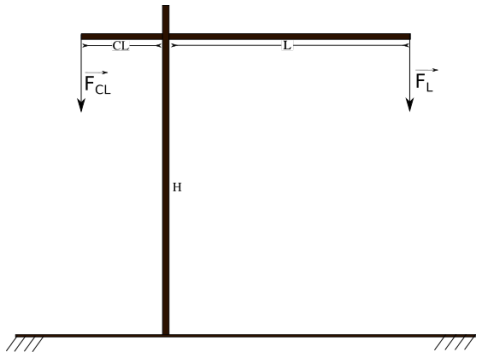

Fig. 3. Diagrama do guindaste

Objetivando torque resultante nulo na estrutura do guindaste,

$$\sum \vec{\tau}_j = 0 \tag{1}$$

e considerando que o torque no próprio eixo da haste é nulo, tem-se que:

$$\vec{\tau}_{CL} + \vec{\tau}_L = 0 \tag{2}$$

Dado que

$$\tau = r \times F$$

:

$$\vec{r}_{CL} \times \vec{F}_{CL} + \vec{r}_L \times \vec{F}_L = 0 \tag{3}$$

Considerando que a força aplicada na lança e na contra-lança devem-se, respectivamente, à força peso exercida pela presença de carga na lança e na contra-lança, tem-se:

$$\vec{r}_{CL} \times m_{CL}\vec{g} + \vec{r}_L \times m_L\vec{g} = 0 \tag{4}$$

$$\|r_L\| = \frac{m_L}{m_{CL}} \|r_{CL}\| \tag{5}$$

A partir da relação 5 e definidas as dimensões nominais da estrutura:

- Lança: 30cm
- Contra-lança: 10cm
- Haste: 40cm

considerando que a única carga da contra-lança será o motor utilizado para içamento ($m_{CL} \approx 20g$) e que não haverá outro contrapeso, o guindaste pode içar até $m_L \approx 60g$, mantendo-se em equilíbrio estático.

Enfim, o desenho inicial da estrutura foi feito com auxílio da ferramenta *SketchUp* [3]. O resultado é como o mostrado na Figura 4.

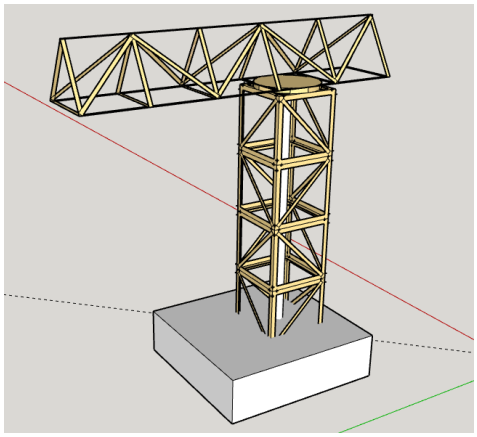

Fig. 4. Projeto 3D da estrutura do guindaste

Optou-se por utilizar treliças na construção da haste e da lança pela ampla utilização deste tipo de estrutura, que apresenta-se como uma solução estrutural simples, prática e econômica para muitas situações de engenharia, especialmente em projeto de passagens superiores, guindastes, pontes e coberturas. O tipo de treliça escolhido consiste em uma variação da treliça de Warren [4].

B. Hardware

O sistema foi interligado através do esquemático representado na Figura 5. A entrada externa do Arduíno e a ponte H foram alimentadas com a fonte 9V, enquanto a porta serial do computador forneceu 5V para os *encoders*.

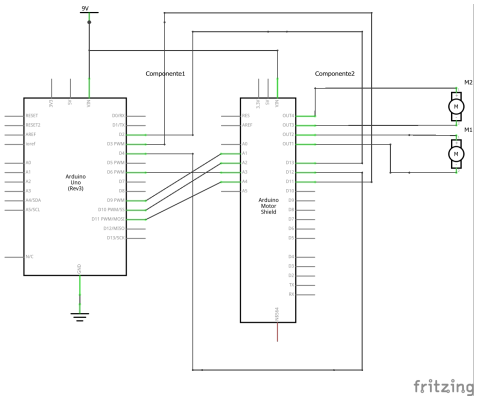

Fig. 5. Diagrama circuito

As saídas 9 e 10, 6 e 11 forneceram o sinal *PWM (Pulse Width Modulation)*, ondas quadradas de largura modular, para o controle e são conectadas nas entradas da ponte H no intuito de realizarem a variação de velocidade dos motores.

Os *encoders* fornecem pulsos ao terem sua luz interrompida e, acrescido o disco ao rotacionar, conseguimos um sinal periódico de onda quadrada na entrada com frequência proporcional a rotação do motor. Eles foram conectados nas entradas 2 e 3 do Arduíno, que possuem funções de interrupção do ciclo de processamento, a fim de não se perder a medição do movimento.

C. Software

O software do projeto foi desenvolvido em duas plataformas, Qt [5] e Arduino [6], podendo assim ser dividido em software gráfico e software de controle.

Interface Gráfica de Usuário (GUI)

O *framework* utilizado para implementar a interface de interação com o usuário foi o Qt, que é baseado na linguagem orientada a objetos C++. Inicialmente, a interface havia sido projetada para ser implementada em Python utilizando a biblioteca *Pygame* [7], mas decisões de projeto posteriores, alinhadas com o pensamento de uma melhor integração *software-hardware*, foi escolhida a ferramenta Qt para o projeto.

O software implementado é orientado a objetos, e possui uma classe principal, onde os componentes tipo *Widget* estão inseridos. Enquanto isso, uma classe de controle, *CraneController*, implementa a lógica por trás do programa. Como o Qt é uma ferramenta de projeto de interfaces orientadas a eventos, a classe de controle é responsável por emitir sinais a partir dos eventos dos widgets.

Dessa forma, se o comando de ”Enviar” um valor é selecionado, o Sinal de clique ativa o método de seleção de valor do controle, que emite o sinal de envio para o sistema de comunicação com o Arduino.

O desenho final da interface está demonstrado na figura abaixo. Foram utilizados Widgets tipo *slider*, pensados de forma a serem mais intuitivos para as respectivas funções - rotação e içamento da lança do guindaste.

Fig. 6. Desenho final da interface gráfica feita com Qt

Como é possível perceber, a interface obedece aos padrões de informação do sistema de comunicação implementado.

Na esquerda, o *dial* controla a rotação do guindaste. Cada comando pode determinar 18^o ou 36^o de rotação da lança, tanto para o sentido horário como anti-horário. Ao enviar o comando 0 para o Arduino, o sistema reconhece a posição atual da lança como a referência inicial para novas interações de rotação - como um *reset* no referencial de posicionamento.

Na direita, é possível ver o *slider* vertical, que indica o controle de altura do cabo do guindaste. Mais uma vez, há a conformidade da interface com o protocolo de comunicação implementado, onde cada nível selecionado representa uma rotação completa do disco do encoder. É possível selecionar uma ou duas voltas para o motor, ou, em termos da interface, 20 ou 40 pontos de rotação, para cima ou para baixo.

A comunicação da interface foi configurada dentro do Qt utilizando a biblioteca padrão de comunicação serial, QtSerialPort. Os padrões de conexão foram inseridos, e foram implementados dois métodos na classe principal do projeto, um para leitura e um para escrita serial na porta USB onde conectava-se o Arduino.

Sistema de Controle

O deslocamento angular desejado em um motor pode ser apontado pela relação da posição de inicio θ e a posição desejada θ_d durante um tempo t . Sendo θ_e = θ - θ_d temos:

$$\lim_{t\to\infty} \theta(t) = \theta_d \implies \lim_{t\to\infty} \theta_e(t) = 0 \tag{6}$$

Através da velocidade, podemos escrever:

$$\dot{\theta_e} = \dot{\theta} - \dot{\theta_d} \tag{7}$$

$$\dot{\theta} = -\lambda\theta_e + \theta_d \tag{8}$$

Excluindo o *feedfoward*($\dot{\theta_d}$) do modelo, temos:

$$\dot{\theta_e} = -\lambda\theta_e \implies \dot{\theta} + \lambda\theta_e = 0 \tag{9}$$

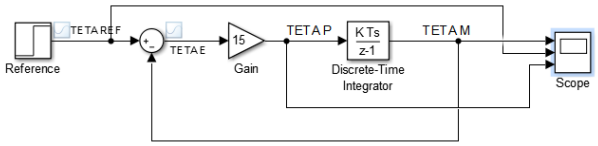

Fig. 7. Diagrama de controle do motor

A equação diferencial 9 resulta no diagrama da **??** com θ a posicao da grua e $\dot{\theta}$ a velocidade do motor. Uma vez recebido da interface o valor de referencia, um degrau de mesmo delta é gerado em θ_e . O erro θ_e é a diferenca entre o valor desejado e θ_m, valor medido, assim como identifica-se o controlador proporcional e o modelo da planta.

O algoritmo implementado itera sob o erro a fim de gerar as ações de controle proporcional para chegar na posição de referência, considerando um sistema de primeira ordem. A função $angulo()$ retorna a posição integrada pelos pulsos do encoder, soma erros positivos e subtrai para erros negativos, e a função $PWM(positivo, negativo)$ uma abstração das funções *analogWrite* do Arduino permitindo a rotação em sentidos diferentes.

Algorithm 1 Algoritmo de controle

```
while |e| ≤ emin do
    vel = (ref − pos) ∗ Kp
    if (vel ≥> 0) then
        PWM = (vel, 0)
    else
        PWM = (0, vel)
    end if
end while
```

D. Protocolo de Comunicação

Inicialmente, a comunicação entre o sistema físico, utilizando o Arduino, e a interface do usuário foi idealizada seguindo a biblioteca "Robust Arduino Serial" [8]. O protocolo estabelecido consistia em duas etapas principais: conexão - para garantir a sincronia da comunicação - e a comunicação, por meio de variáveis *enum*, que indicavam uma ação a ser executada e seus respectivos parâmetros.

Contudo, a utilização da "Robust Arduino Serial" foi impossibilitada por apresentar dependências a outras bibliotecas instáveis ou indisponíveis nos sistemas operacionais Windows e macOS, disponíveis ao grupo. Assim, optou-se por implementar um protocolo mais simples, seguindo a máquina de estados apresentada na Figura 8.

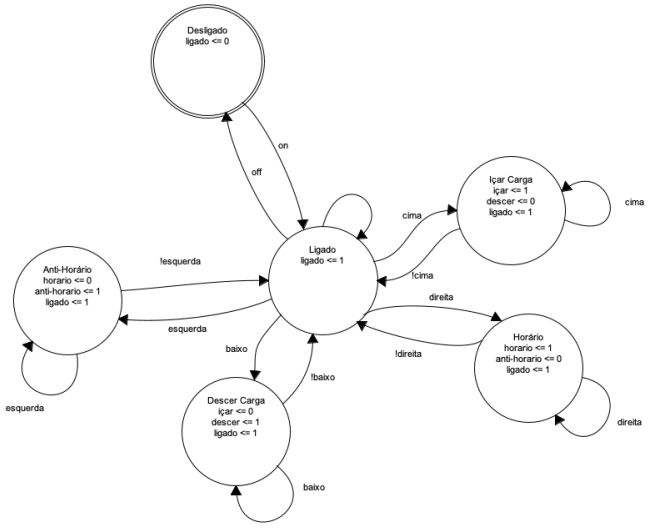

Fig. 8. Máquina de estados do guindaste

Como descrito anteriormente, o usuário tem a opção de enviar os ângulos ($[-36, -18, 0, 18, 36]$) para a rotação ou o número de rotações do disco *encoder* ($[-40, -20, 0, 20, 40]$) para o içamento do guindaste. No âmbito da rotação, tem-se:

TABLE I
VALORES POSSÍVEIS PARA O COMANDO DE ROTAÇÃO

Opção	Ação	Caractere Enviado	Decodificação
-36º	36º anti-horário	0	48
-18º	18º anti-horário	1	49
0	Posição = referência	2	50
18º	18º horário	3	51
36º	36º horário	4	52

Conforme a Tabela I, tem-se o seguinte fluxo:

1) O usuário escolhe qual ângulo, em relação à posição atual, deseja que o guindaste se desloque;
2) A interface codifica a seleção em um caractere representando um número entre 0 e 4;
3) O Arduino faz a leitura da saída da interface e decodifica o caracter recebido conforme os valores da tabela ASCII;
4) A ação de controle desejada é executada sobre o guindaste.

TABLE II
VALORES POSSÍVEIS PARA O COMANDO DE IÇAMENTO

Opção	Ação	Caractere Enviado	Decodificação
-40	2 rotações subida	5	53
-20	1 rotação subida	6	54
0	Posição = referência	7	55
20	1 rotação descida	8	56
40	2 rotações descida	9	57

Conforme a Tabela II, tem-se o seguinte fluxo:

1) O usuário escolhe qual o número de rotações da bobina de içamento, em relação à posição atual, deseja que o guindaste realize. O valor visualizado pelo usuário corresponde ao número de ranhuras do disco encoder que serão lidas;
2) A interface codifica a seleção em um caractere representando um número entre 5 e 9;
3) O Arduino faz a leitura da saída da interface e decodifica o caracter recebido conforme os valores da tabela ASCII;
4) A ação de controle desejada é executada sobre o guindaste.

Assim, observa-se que os sinais de mudança de estado indicados na Figura 8 são descritos como:

- baixo: 8 ou 9 (codificado) - 56 ou 57 (decodificado);
- cima: 5 ou 6 (codificado) - 53 ou 54 (decodificado);
- esquerda: 0 ou 1 (codificado) - 48 ou 49 (decodificado);
- direita: 3 ou 4 (codificado) - 51 ou 52 (decodificado);

IV. RESULTADOS EXPERIMENTAIS

Os resultados experimentais obtidos nesse trabalho mostra a interferência do sistema de controle no hardware e a montagem mecânica e eletrônica do sistema.

A ação de controle pode ser visto na Figura 9, assim como sua atuação de controle na Figura 10. Nota-se um comportamento característico de um sistema de primeira ordem como esperado na equação 9.

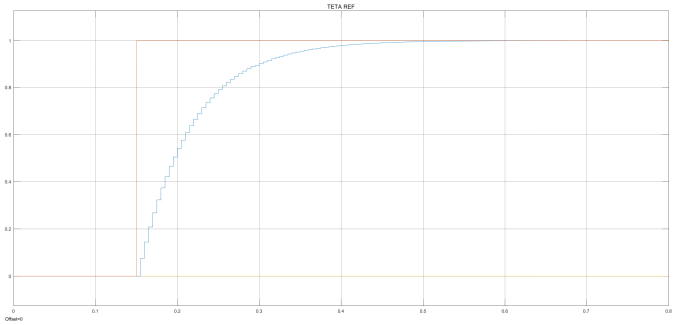

Fig. 9. Resposta do sistema a um degrau unitário na referência θ_r.

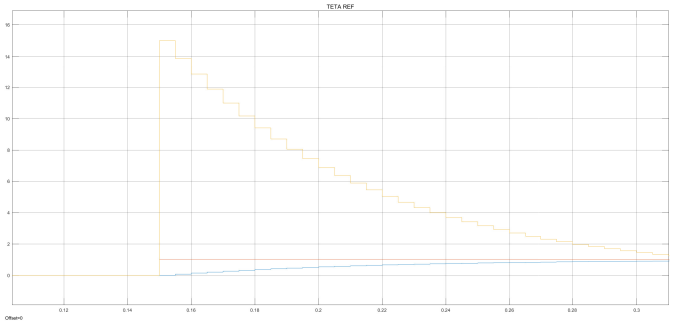

Fig. 10. Atuação do controlador na velocidade $\dot{\theta}$

O produto final do projeto foi ilustrado nas Figura 11. Observa-se que a treliça tipo Warren definida foi levemente modificada, por simplicidade de construção, mantendo-se o efeito estrutural desejado. Nota-se, também a modificação na geometria da contra-lança para melhor afixar o motor de içamento.

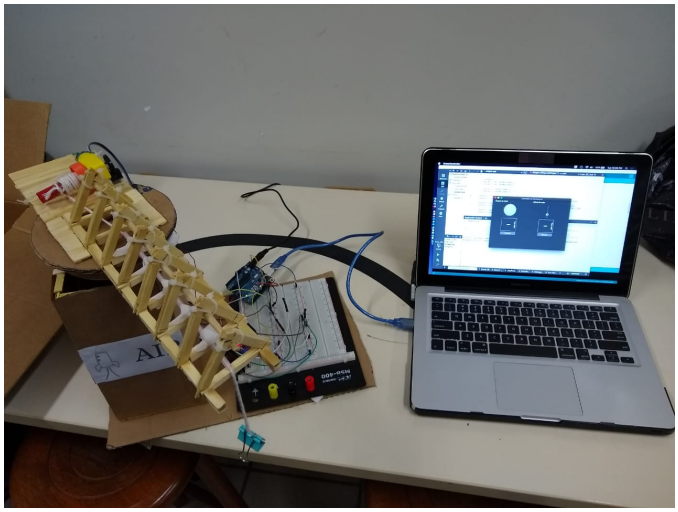

Fig. 11. Sistema integrado

O circuito implementado utilizando a ponte H, o motor, os sensores encoder e o Arduino pode ser visto na Figura 12.

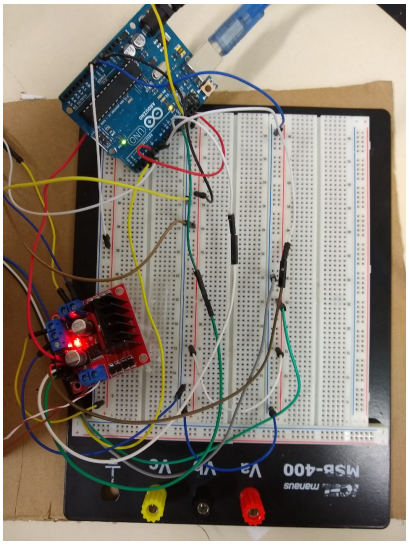

Fig. 12. Circuito montado

V. Discussão e Conclusões

Neste trabalho, foi possível realizar um projeto de um sistema complexo, com várias definições a serem realizadas antes da execução, e com algumas dificuldades de integrações naturais de serem percebidas nesse tipo de produto.

Inicialmente, a ideia de uma estrutura controlada a partir da velocidade de rotação do motor parecia a melhor decisão de projeto, uma vez que representava a visão do grupo enquanto controle de movimento do guindaste. Posteriormente, com os experimentos e com novas perspectivas adquiridas no progresso do projeto, foram necessários novos cálculos para adequar os requisitos à realidade do produto.

O comportamento de um sistema de primeira ordem nos facilitou quanto ao controle do sistema, evitando a presença de sobressinais que interfeririam bastante no sentido de rotação do motor e posicionamento da grua. Porém dificuldades também apareceram, como ruídos de medição no encoder que mostrou-se necessário a implementação de um filtro digital de primeira ordem na entrada.

Por fim, com este trabalho, foi possível ter contato com um projeto complexo, que exigiu o uso de conhecimentos específicos em áreas de conhecimento diferentes - mecânica, eletrônica, computação - ao mesmo tempo em que foram utilizados conceitos de Engenharia de Sistemas para fazer a gestão, a criação e a implementação do projeto proposto, bem como a integração final dos subsistemas citados acima.

O trabalho foi capaz de demonstrar as dificuldades impostas nesse processo de integração, e foi necessário adequar as atividades para serem executadas pelos membros da equipe, que precisou sempre se comunicar e trabalhar em conjunto para obter os resultados que eram esperados.

References

[1] P. Zimmer, "Fizzim - the free fsm design tool," 2018. [Online]. Available: http://www.fizzim.com/

[2] IHCM, "Ihmc cmaptools," Download, 2012. [Online]. Available: http://cmap.ihmc.us/download/index.php

[3] A. Chopra, *Introduction to Google SketchUp*, 2nd ed. Wiley Publishing, 2012.

[4] I. Gomes, "Estudo e análise de treliças," 04 2016.

[5] R. Rischpater, *Application development with qt creator*. Packt Publishing Ltd, 2013.

[6] S. Arduino, "Arduino," *Arduino LLC*, 2015.

[7] A. Sweigart, *Making Games with Python & Pygame*. CreateSpace North Charleston, 2012.

[8] A. Raffin, "Robust arduino serial protocol in python," 2018. [Online]. Available: https://github.com/araffin/python-arduino-serial/