FACULDADE SENAI DE TECNOLOGIA MECATRÔNICA

APLICAÇÃO DE CNC PARA USINAGEM E FURAÇÃO DE PLACAS DE CIRCUITO IMPRESSO

CNC APPLICATION FOR MACHINING AND DRILLING PRINTED CIRCUIT BOARDS

Paulo Roberto de Souza[1], [i]
Paulo Sebastião Ladivez[2], [ii]
Carlos Aurelio González Cardozo[3], [iii]
José Roberto dos Santos[4], [iv]

RESUMO

O objetivo deste artigo é documentar o desenvolvimento de uma aplicação (*software*) capaz de interpretar o código G e, aplicando os respectivos comandos de movimentação de ferramenta de corte a um protótipo de máquina, realizar a usinagem e furação de placas de circuito impresso, visando aumentar a precisão e repetibilidade dessas placas, quando produzidas em pequena escala, testes de conceito de projeto, para atender demandas de profissionais autônomos ou pequenas empresas, reduzindo assim o seu custo final. Descreve-se uma aplicação com processamento de dados em linguagem Python e *interface* com usuário através de um navegador de internet, por exemplo, Google Chrome e também apresenta-se o protótipo de uma estrutura mecânica para realização da usinagem e furação. Finalmente, mostra-se os resultados dos testes realizados com este protótipo, comparando o projeto da placa realizado com auxílio de programas de computador com a placa usinada pela máquina.

ABSTRACT

The purpose of this article is to document the development of an application (software) capable of interpreting the G code and, applying the respective cutting tool movement commands to a machine prototype, performing the machining and drilling of printed circuit boards, in order to increase the precision and repeatability of these boards, when produced in small scale design concept tests to meet the demands of freelance professionals or small companies thus reducing their final cost. An application using Python for data processing and an user interface through an internet browser for example, Google Chrome are described. Also, a mechanical structure for machining and drilling is presented. Finally, the results of the tests carried out with this prototype are shown, comparing the board design carried out with the aid of computer programs with the board machined by the machine.

1 Pós-Graduando em Automação Industrial pela Faculdade SENAI de Tecnologia Mecatrônica. E-mail: paulo.souza87@gmail.com

2 Professor da Faculdade SENAI de Tecnologia Mecatrônica. E-mail: paulo.ladivez@sp.senai.br

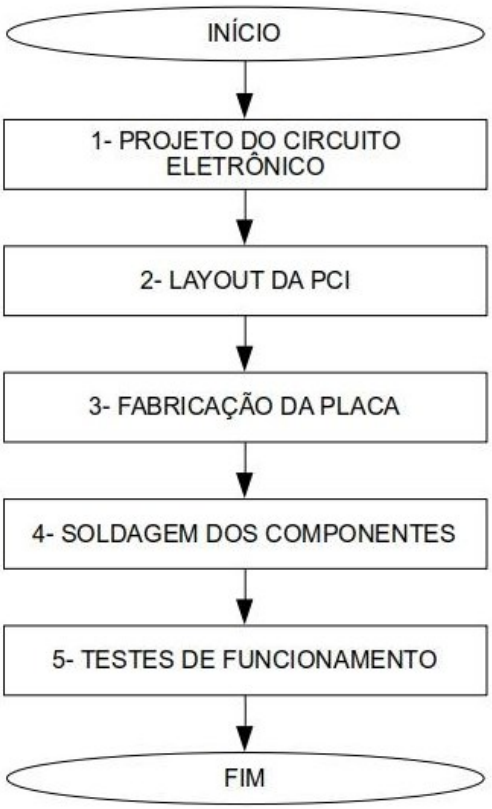
3 Mestre em Engenharia Mecânica, Professor da Faculdade SENAI de Tecnologia Mecatrônica. E-mail: carlos.cardozo@sp.senai.br

4 Professor da Faculdade SENAI de Tecnologia Mecatrônica. E-mail: joseroberto@sp.senai.br

1 INTRODUÇÃO

Atualmente, a eletrônica tem forte presença no nosso dia a dia, o que faz com que entusiastas, profissionais autônomos e pequenas empresas desenvolvam suas próprias Placas de Circuito Impresso (PCIs) para aplicações específicas. O fluxograma 1 mostra as principais etapas a serem realizadas durante o desenvolvimento de uma PCI.

Fluxograma 1 - Etapas para desenvolvimento de PCI



Fonte: Elaborado pelo autor

As etapas 1 e 2 do fluxograma 1 podem ser realizadas com o auxílio de programas EDA (do inglês: Automação do Desenvolvimento Eletrônico) como: KiCad (*Windows, MAC, Linux*), Proteus (*Windows*), *Eagle* (*Windows, MAC, Linux*), etc. Alguns deles, inclusive, oferecem ferramentas para automação de tarefas, como roteamento automático das trilhas.

Um projeto desenvolvido usando este tipo de programa tem precisão de até 0,01 mm para posicionamento e dimensões de trilhas e furos. Porém, se a furação da placa e obtenção das trilhas forem realizadas manualmente, utilizando métodos como serigrafia, deposição metálica, etc; a precisão e repetibilidade ficam condicionadas à habilidade e experiência de quem está fabricando a PCI.

Com o objetivo de melhorar a qualidade do processo de fabricação de PCIs, pode ser utilizada uma máquina com Comando Numérico Computadorizado (CNC) para obtenção das trilhas e furação da placa.

Como as máquinas CNC disponíveis no mercado podem ter custo inviável para a maioria dos entusiastas e profissionais autônomos, este artigo propõe o desenvolvimento de

um *software* (ou aplicação) capaz de interpretar o código G, linguagem comumente utilizada para envio de instruções a uma máquina CNC, e realizar a movimentação correta de uma ferramenta de corte sobre a placa para obtenção das trilhas e/ou furos. Para isso, será construído também um protótipo de uma máquina CNC de pequeno porte, com componentes acessíveis, que possa executar os comandos do código G e assim realizar a fabricação da PCI.

2 MÁQUINA CNC

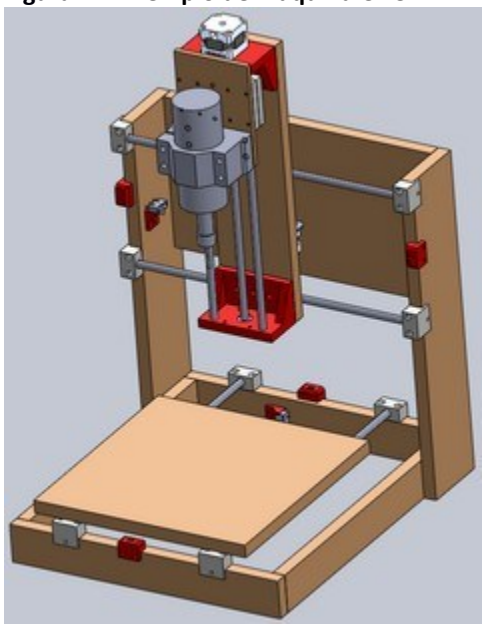
A máquina CNC será dotada de três eixos, dois deles para movimentação da ferramenta na horizontal (eixos X e Y) e um para vertical (eixo Z). Tais eixos serão acionados por motores de passo por meio de correias sincronizadoras (eixos X e Y) e eixo roscado (eixo Z).

A máquina CNC possuirá também sensores para o referenciamento dos eixos, uma vez que será aproveitada uma vantagem dos motores de passo, que é a de poder trabalhar em malha aberta, pois “os motores de passo se movem com incrementos ou passos que podem ser quantificados. Desde que o motor funcione com o torque especificado, a posição do eixo é conhecida a todo tempo sem a necessidade de um mecanismo de realimentação.” (CONDIT; JONES, 2004, p.1).

Para realização da usinagem e furação, a máquina terá ainda um motor de corrente contínua (CC), no qual serão acopladas as ferramentas de corte (fresas ou brocas). Este motor é comumente conhecido como *spindle* ou eixo-árvore.

A figura 1 mostra um modelo de estrutura para máquina CNC. Esta estrutura permite a movimentação da ferramenta nos três eixos (X, Y e Z).

Figura 1 – Exemplo de máquina CNC

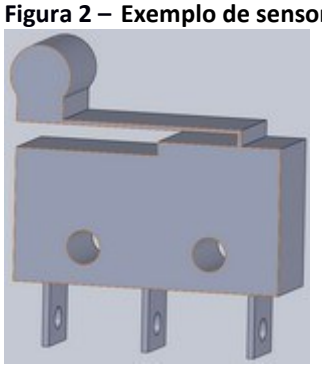


Fonte: Elaborado pelo autor

O modelo apresentado na figura 1 será utilizado na construção do protótipo para este trabalho.

Já a figura 2 mostra um sensor, dispositivo que “muda seu comportamento sob a ação de uma grandeza física {..} convertendo uma quantidade física em um sinal elétrico.” segundo Rosário (2005, p.55), que pode ser utilizado para referenciamento dos eixos. Este sensor é conhecido como sensor de fim de curso, ou *limit microswitch*.

Figura 2 – Exemplo de sensor para referenciamento dos eixos



Fonte: Elaborado pelo autor

3 CONTROLADOR

Para este projeto, será utilizada uma placa *RaspBerry Pi* (RPi) modelo 3B como controlador. A aplicação (programação da máquina) será desenvolvida em linguagem *Python* (versão 3.6) e dividida em duas partes principais (Interação com usuário e Execução da usinagem), a serem detalhadas mais adiante.

O fluxograma 2 mostra o detalhamento do processo de fabricação da PCI utilizando a máquina CNC que será desenvolvida.

Fluxograma 2 - Fabricação de PCI utilizando máquina CNC

Fonte: Elaborado pelo autor

Para o projeto da PCI ainda continuará sendo utilizado um programa EDA.

A geração do código “G” pode ser realizada com programas como FlatCAM ou CooperCam, por exemplo, a partir do arquivo “*Gerber*” gerado pelo EDA.

Antes de enviar o código “G” para a máquina, o usuário deve posicionar e fixar, na mesa da máquina, a placa que será usinada.

3.1 Interação com usuário

Para que o usuário possa enviar o código “G” para a máquina, será desenvolvido um *webserver* com o auxílio de um *FrameWork*, “um conjunto de classes que incorporam um projeto genérico para solucionar uma família de problemas relacionados”, segundo Foote e Johnson (1988, p.1), para Python chamado Flask. Ao ligar a máquina, este *webserver* é iniciado e disponibiliza o acesso, via rede local, a um conjunto de “páginas *web”* HTML (*HyperText Markup Language*, Linguagem de Marcação de HiperTexto) para que o usuário possa interagir com a máquina (executar o referenciamento dos motores e carregar código “G”). Este acesso pode ser feito por qualquer dispositivo eletrônico que possua um navegador de internet (Google Chrome, Internet Explorer, FireFox*,* etc), digitando, na barra de endereços, o endereço IP da RPi e a porta utilizada, por exemplo, “192.168.0.110:8000”, sendo que “192.168.0.110” é o IP do controlador (RPi) na rede local e “8000” é a porta utilizada pelo *webserver*.

O processo para carregar o código “G” será o mesmo utilizado para fazer *upload* de arquivos para um determinado *site*: a página possui um botão, que abre uma janela para procurar e selecionar o arquivo desejado.

3.2 Execução da usinagem

Ao carregar o código “G”, o programa executará as rotinas de verificação e execução da usinagem. Para isso, serão implementadas as funções “G” mais comuns (tanto para verificação como para execução), são elas:

a) G00 – deslocamento linear da ferramenta em velocidade máxima;
b) G01 – deslocamento linear da ferramenta com velocidade controlada;
c) M03 – liga eixo-árvore em sentido horário;
d) M05 – desliga eixo-árvore;
e) M30 – fim de programa.

4 DESENVOLVIMENTO

Para este projeto foi desenvolvido um protótipo, com base no modelo apresentado na figura 1, com estrutura em madeira MDF (*Medium Density Fiberboard*, placa de fibra de madeira de média densidade) e utilizando eixos retificados e rolamentos lineares para servirem de guia para os eixos da máquina.

4.1 Estrutura mecânica

O protótipo da estrutura desenvolvida pode ser visto na figura 3.

Figura 3 – Montagem da nova estrutura mecânica

Fonte: Elaborado pelo autor

Para agilizar o processo de fabricação e montagem, os suportes dos eixos e sensores foram feitos em impressora 3D.

Para movimentar os eixos, um motor de passo é acoplado (via correia dentada para os eixos X e Y, e via fuso para o eixo Z) a cada eixo.

4.2 *Software* da aplicação

Como citado no item 3, a aplicação foi desenvolvida em Python e as páginas de interação com o usuário foram escritas em HTML.

O quadro 1 mostra a declaração, função construtora e definição (ou instância) de Classes Python desenvolvidas para receber as informações adequadas para controle dos motores dos eixos e ferramenta.

Quadro 1 - Declaração das classes dos motores e da ferramenta

```
#Declaração da classe MOTOR, com as características dos eixos da máquina
class MOTOR:
   '''
      --> "eixo" = nome do eixo (X, Y ou Z)
      --> "step_pin" = pino para enviar sinal de 'step' para o driver de motor de passo
      --> "dir_pin" = pino para enviar sinal de direção para o driver de motor de passo
      --> "motor_res" = resolução do motor (steps / revolução)
```

```
    --> "driver_res" = resolução do driver (1-32 => Resolução = 1 / driver_res)
    --> "passo" = mm / rotação do motor
    --> "vel_home" = velocidade para realização do home em mm/min
    --> "direcao_home" = sentido de rotação para realização do home do motor (UP ou DOWN)
    --> "sensor_home_pin" = pino para conexão do sensor de home
    --> "sensor_home_type" = tipo de sensor para home -> 0 = NF; 1 = NA
    --> "offset" = distância entre sensor de home e a posição "zero" do eixo
  '''

  def __init__(self, eixo, step_pin, dir_pin, motor_res, driver_res, passo, vel_home,
direcao_home, sensor_home_pin, sensor_home_type, offset):
    self.eixo = eixo
    self.step_pin = ATUADOR(step_pin)
    self.dir_pin = ATUADOR(dir_pin)
    self.precisao = passo / (motor_res * driver_res) #mm / (u)step
    self.vel_home = vel_home
    self.direcao_home = direcao_home
    self.sensor_home = SENSOR(sensor_home_pin)
    self.sensor_home_type = sensor_home_type
    self.offset = offset

    self.cur_pos_mm = 0.0
    self.target_pos_mm = 0.0
    self.direcao = UP
    self.cur_pos_ustep = int(self.cur_pos_mm/self.precisao)
    self.target_pos_ustep = int(self.target_pos_mm/self.precisao)
    self.period_s = 0
    self.pos_index = 0
    self.check = 0
    self.counter = 0
    self.timestamp = 0
    self.h_ustepsTOgo = 0   #half micro steps to reach target_pos_ustep ((target - current) * 2)

    self.stop_move()

#definição dos motores
motor = [   MOTOR("x", 14, 15, 200.0, 32.0, 40.0, MAX_XY_speed, DOWN, 18, 0, 05.4),
            MOTOR("y", 16, 20, 200.0, 32.0, 40.0, MAX_XY_speed, DOWN, 21, 0, 00.0),
            MOTOR("z", 13, 19, 200.0, 01.0, 1.50, MAX_Z_speed, DOWN, 26, 1, 00.0) ]

#Declaração da classe Tool, com as características da ferramenta
class Tool:

  T_plus_pin = ATUADOR(6)    #tool pin to define CW rotation
  T_minus_pin = ATUADOR(12)  #tool pin to define CCW rotation

  '''
    --> "name" = tool name
    --> "size" = tool length (mm)
    --> "dia" = tool diameter (mm)
```

```
  '''
  def __init__(self, name, size, dia):
    self.name = name    #nome da ferramenta
    self.size = size    #comprimento da ferramenta
    self.dia = dia      #diametro da ferramenta
    self.stat = 0       #1 = Ligada (Girando); 0 = Desligada (Parada)
    self.OFF()

#define uma ferramenta
tool1 = Tool("tool1", 30, 1)
```

Fonte: Elaborado pelo autor

O quadro 2 mostra dois exemplos de funções em Python, utilizando o *framework* Flask, para exibição das páginas em HTML de interação com o usuário.

Quadro 2 - Exemplos de funções para exibição das páginas de interação com o usuário

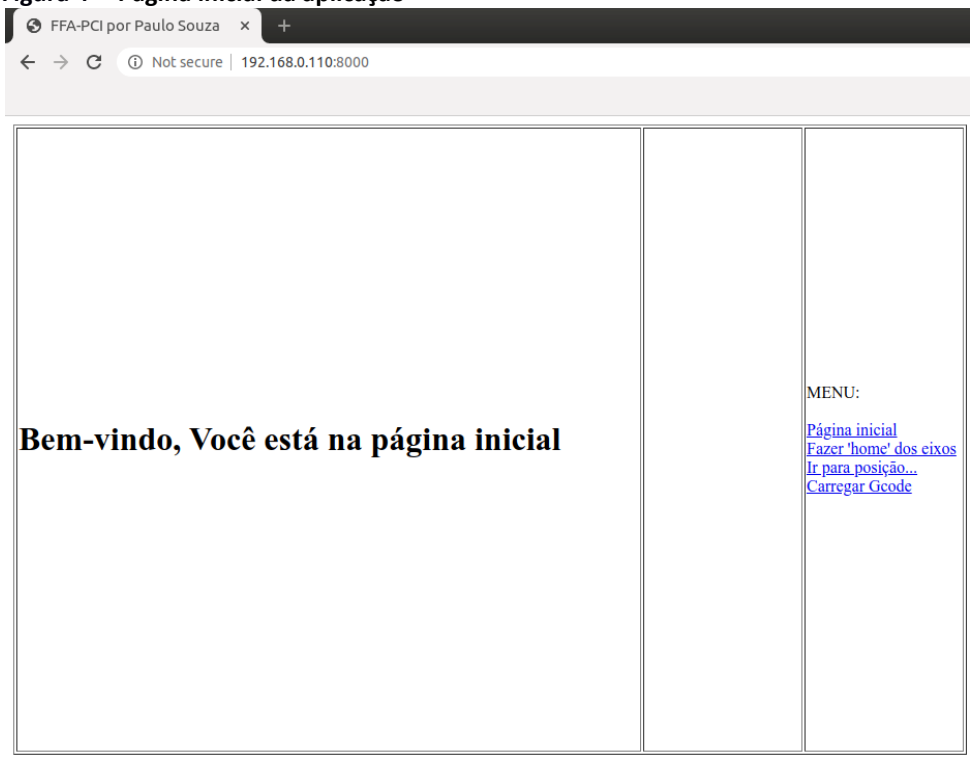
```
#Função para exibir a página inicial quando o usuário digita o IP+Porta da máquina na rede
(192.168.0.110:8000) ou clica em “Página inicial”
@app.route("/")
@app.route("/index")
def index():
  return render_template("index.html", CONTENT = "Bem-vindo, Você está na página inicial")

#Função para exibir a página com instrução inicial para referenciamento dos eixos
@app.route("/home")
def home():
  return render_template("go_home.html")
```

Fonte: Elaborado pelo autor

A figura 4 mostra a página inicial (Endereço: http://192.168.0.110:8000) da aplicação.

Figura 4 – Página inicial da aplicação



Fonte: Elaborado pelo autor

A página inicial possui um menu à direita, a partir do qual é possível acessar as funções da máquina. São elas:

a) “Página inicial” (permite voltar à página inicial);

b)“Fazer *home* dos eixos” (inicia o referenciamento dos eixos);

c) “Ir para posição...” (mover a ferramenta para uma posição específica);

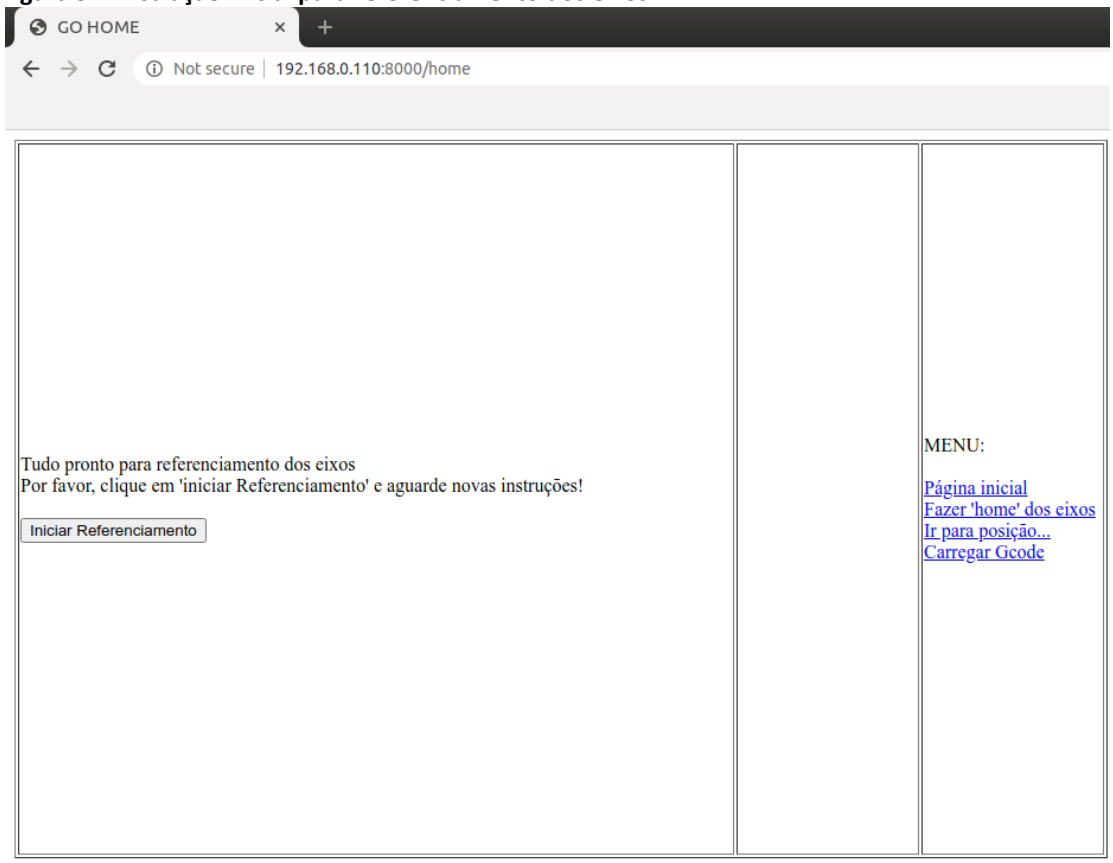
d) “Carregar *Gcode*” (carregar um arquivo com código G que será executado pela máquina).

Para cada uma das funções escolhidas, são apresentadas as orientações necessárias:

a) Fazer *home* dos eixos: A máquina fará o referenciamento dos eixos X e Y e em seguida apresentará instruções para realização do referenciamento do eixo Z. Para o referenciamento deste último eixo, é necessário conectar dois “jacarés”, um na placa a ser usinada e outro na ferramenta, a fim de possibilitar a detecção do contato entre ferramenta e placa.

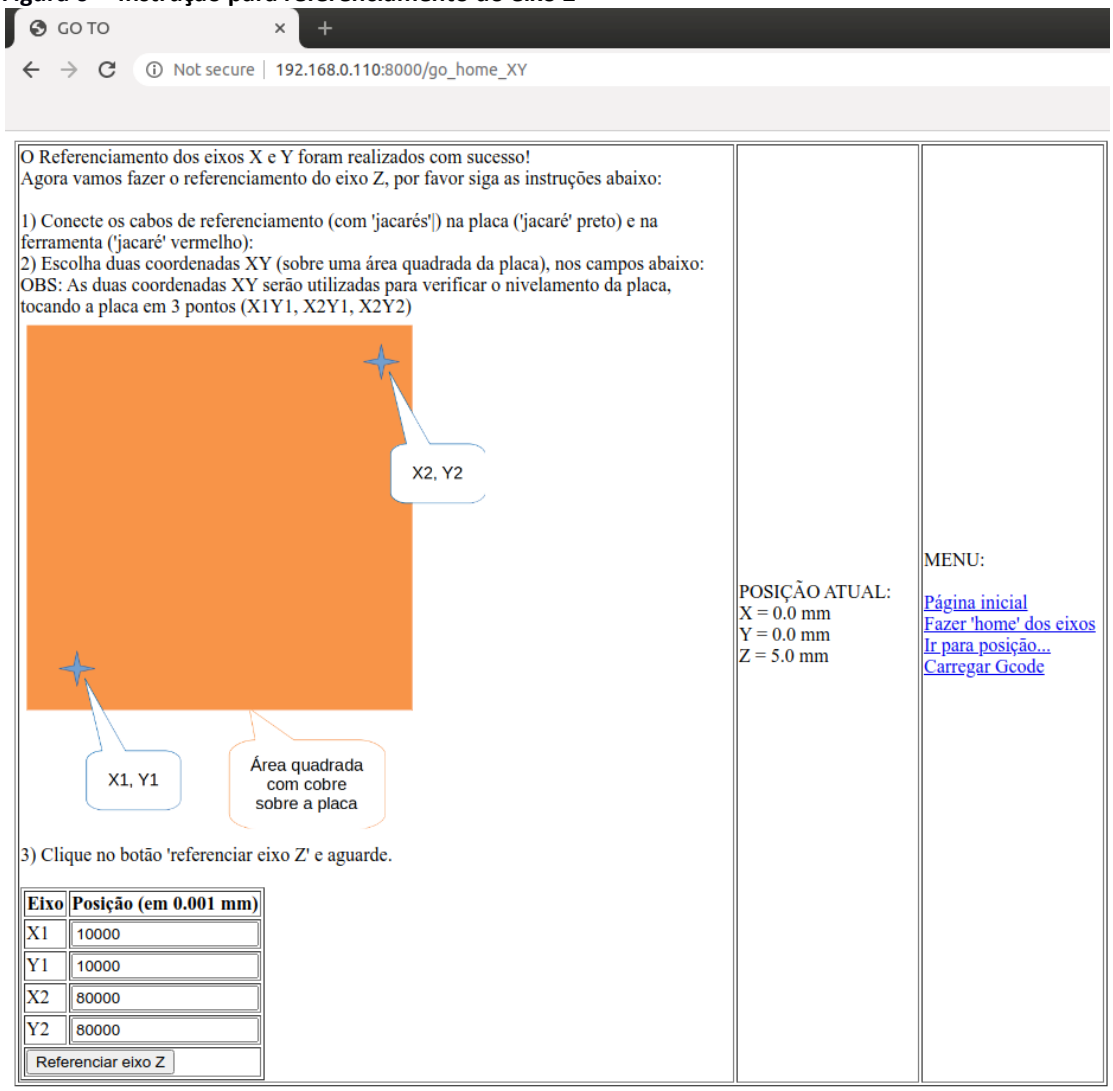
As figuras 5 a 7 mostram as instruções no navegador para referenciamento dos eixos, nos endereços: http://192.168.0.110:8000/home, http://192.168.0.110:8000/go_home_XY, http://192.168.0.110:8000/go_home_Z, respectivamente.

Figura 5 – Instrução inicial para referenciamento dos eixos



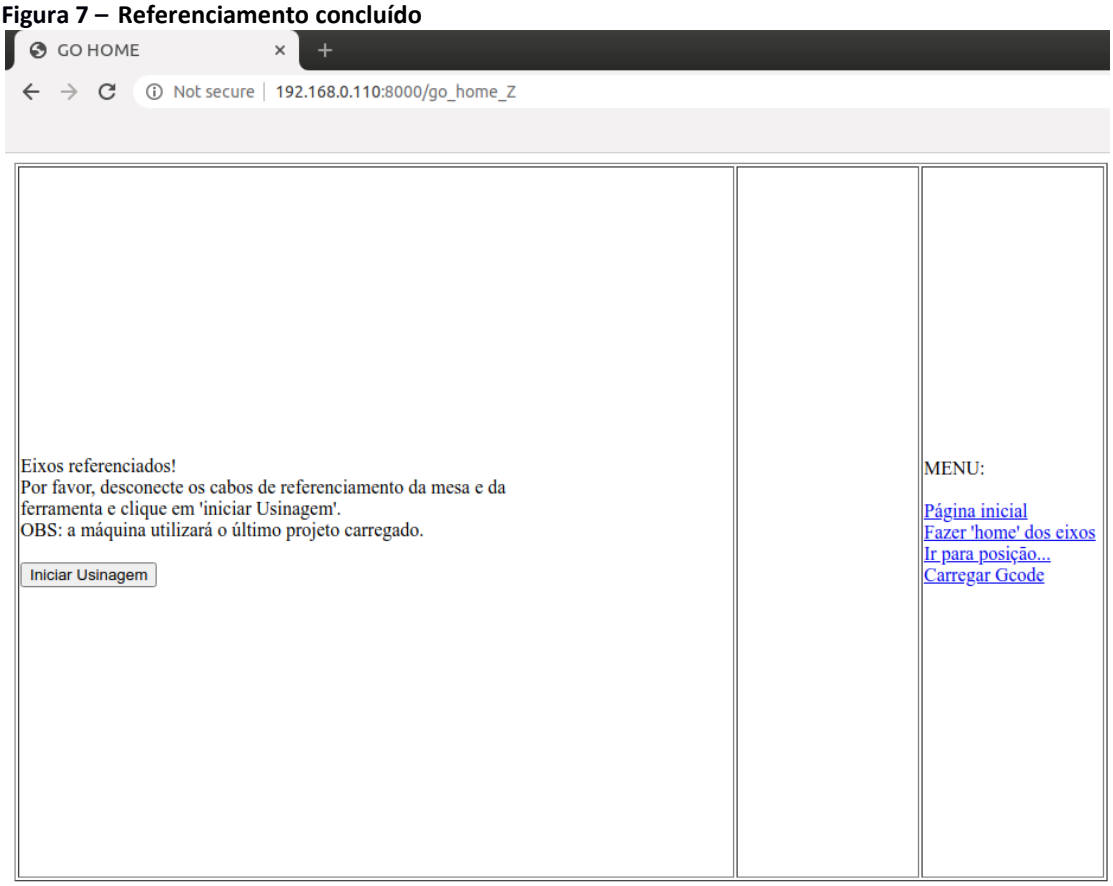
Fonte: Elaborado pelo autor

Figura 6 – Instrução para referenciamento do eixo Z



Fonte: Elaborado pelo autor

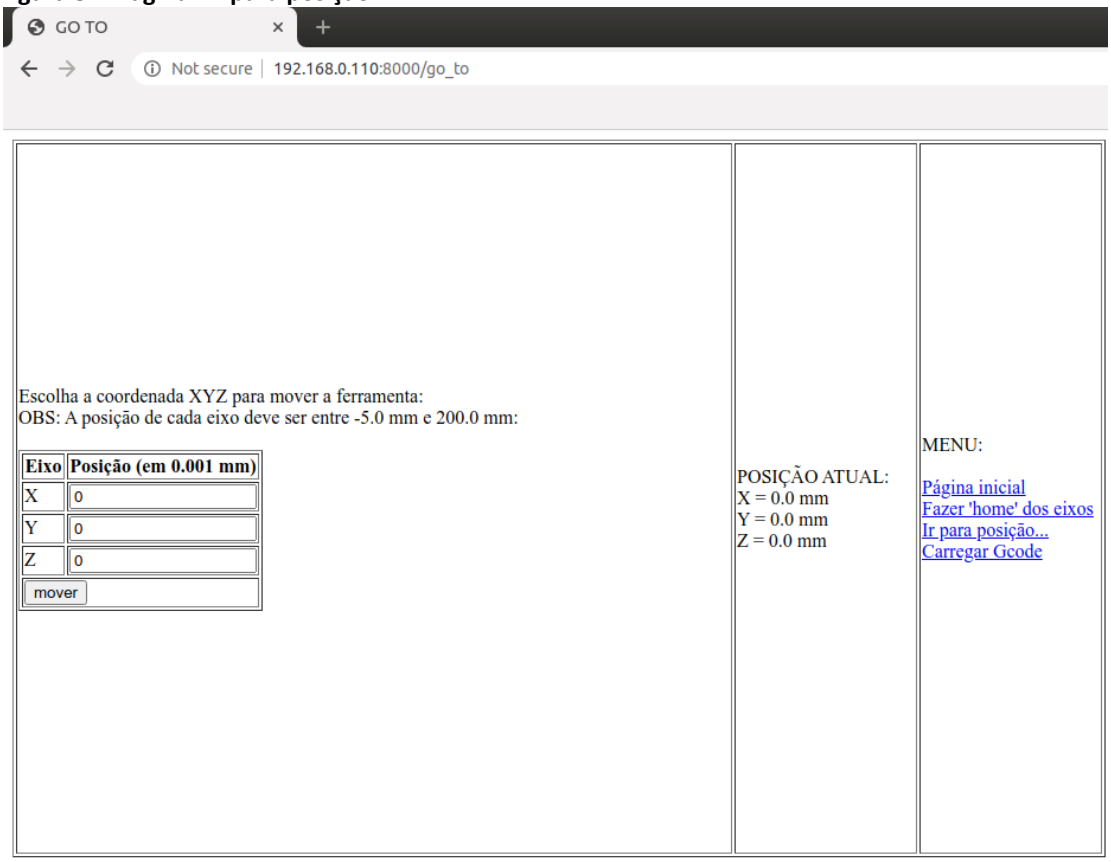
Figura 7 – Referenciamento concluído



Fonte: Elaborado pelo autor

b) Ir para posição...: Será solicitado ao usuário que digite, no formulário que aparecer, as coordenadas X, Y e Z para as quais se deseja movimentar a ferramenta. A unidade está em milésimos de milímetros (0,001 mm), por exemplo, para mover o eixo X para posição 10,0 mm, deve-se colocar 10000 no respectivo campo do formulário. A figura 8 mostra a página em questão, no endereço http://192.168.0.110:8000/go_to.

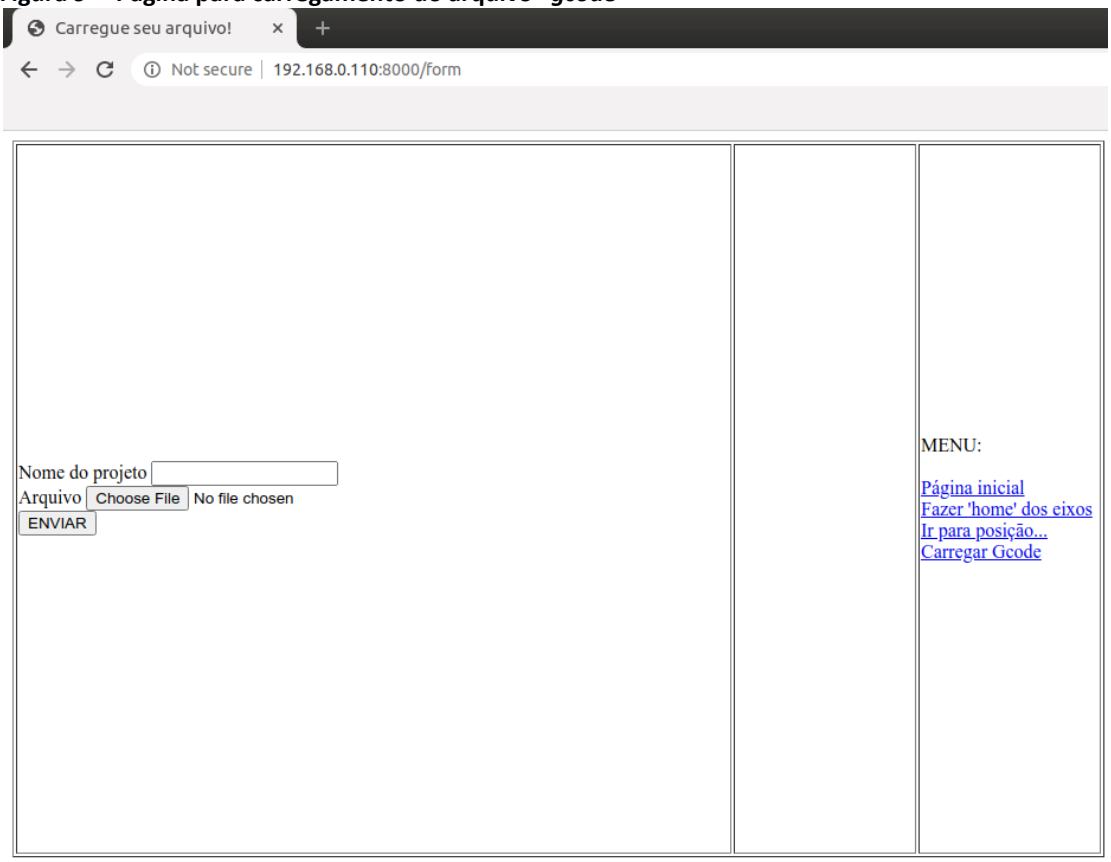
Figura 8 – Página "Ir para posição..."



Fonte: Elaborado pelo autor

c) Carregar *Gcode*: Será disponibilizado um botão para o usuário escolher o arquivo que será carregado para realização da usinagem. São aceitos arquivos “.txt” e “*.gcode*”. Após o carregamento do arquivo, o código é analisado e, na ausência de erros, o referenciamento dos eixos e posterior usinagem da placa são iniciados. A figura 9 mostra a página com botão para carregar o arquivo na máquina CNC, no endereço http://192.168.0.110:8000/form.

Figura 9 – Página para carregamento do arquivo “*gcode*”



Fonte: Elaborado pelo autor

4.3 Teste de usinagem

A fim de testar a máquina, um arquivo “*.gcode*” foi gerado a partir de um arquivo “*Gerber*” utilizando os *softwares* KiCad (circuito e *layout* da placa e geração do arquivo “*Gerber*”) e FlatCAM (obtenção do arquivo “*.gcode*”). Após a geração do arquivo “*.gcode*”, este foi carregado na máquina e a usinagem foi realizada. A figura 10 mostra o projeto da placa no FlatCAM (à esquerda) e uma placa usinada a partir deste arquivo “*.gcode*” (à direita).

Figura 10 – Projeto (esquerda) e usinagem (direita) da placa

Fonte: Elaborado pelo autor

5 CONSIDERAÇÕES FINAIS

A aplicação desenvolvida em Python executa corretamente os comandos presentes no arquivo “*.gcode*”:

a) ao receber os comandos G00 e G01, a aplicação calcula os parâmetros de movimentação e aciona os motores de forma adequada a posicionar a ferramenta na coordenada desejada e com velocidade apropriada;

b) ao receber os comandos M03 e M05, o eixo-árvore da ferramenta é ligado e desligado, respectivamente, de forma correta;

c) ao receber o comando M30, o programa é encerrado e a aplicação permite a realização de usinagem de uma nova placa, ou outro “*.gcode*” na mesma placa.

Observações importantes:

a) como a estrutura deste protótipo foi construída em madeira MDF, ela não se mostrou suficientemente rígida para suportar alguns esforços durante a usinagem da placa. Isso ocasiona, em alguns momentos, uma trepidação da ferramenta e provoca imperfeições na trilha usinada;

b) o *software* utilizado para obtenção do *gcode* (FlatCAM) divide as linhas da trilha em vários pequenos segmentos de reta (com comprimento aproximado de 0,5 mm ou menor), mesmo em momentos nos quais a trilha é uma “linha reta” e, para cada segmento, é gerada uma nova coordenada, o que acaba por afetar a velocidade de movimentação da ferramenta devido ao aumento no número de cálculos realizados durante a execução da usinagem.

Para resolver os problemas observados acima, pode-se aplicar algumas melhorias, conforme as descritas abaixo:

a) construção de uma estrutura utilizando, por exemplo, perfil de alumínio e eixos retificados mais rolamentos lineares (ou guias lineares mais patins), de forma que a estrutura fique rígida o suficiente;

b) melhorar a eficiência da aplicação na realização dos cálculos e/ou utilizar outro *software* que seja capaz de obter o código G de forma mais eficiente.

REFERÊNCIAS

CONDIT, Reston; JONES, Douglas W. ***AN907 - Stepping motors fundamentals***. Microchip Tecnology Inc., 2004. Disponível em: http://ww1.microchip.com/downloads/en/AppNotes/00907a.pdf. Acesso em: 28 dez. 2020.

FOOTE, Brian; JOHNSON, Ralph E. ***Designing reusable classes***. *Department of Computer Science, University of Illinois,* 1988. Disponível em: https://www.cse.msu.edu/~cse870/Input/SS2002/MiniProject/Sources/DRC.pdf. Acesso em: 01 dez. 2020.

ROSÁRIO, João Maurício. **Princípios de mecatrônica**. São Paulo: Prentice Hall, 2005.

REFERÊNCIAS CONSULTADAS

GAROA HACKER CLUBE. **Placas de circuito impresso / fabricação profissional de PCI**. Disponível em: https://garoa.net.br/wiki/Placas_de_Circuito_Impresso/Fabrica%C3%A7%C3%A3o_Profissional_de_PCI. Acesso em: 21 out. 2020.

PALLETS. **Flask web development, one drop at a time**. Disponível em: https://flask.palletsprojects.com/en/1.1.x/. Acesso em: 30 nov. 2020.

SOARES, Márcio José. **Como projetar um robô – parte 1**. Mecatrônica Fácil. São Paulo: Editora Saber Ltda., ano 6, n. 38, p. 30-33, 2008.

SOARES, Márcio José. **Como projetar um robô – parte 2**. Mecatrônica Fácil. São Paulo: Editora Saber Ltda., ano 6, n. 39, p. 30-34, 2008.

UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO". **Motor de passo (aula)**. Disponível em: http://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/aula3-motor-de-passo-2013-1-13-03-2013-final.pdf. Acesso em: 21 out. 2020.

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ. **Minicurso: fabricação de PCI**. Disponível em: http://dainf.ct.utfpr.edu.br/peteco/wp-content/uploads/2013/01/Minicurso_PCI.pdf. Acesso em: 21 out. 2020.

AGRADECIMENTOS

Aos meus pais, Sérgio e Sônia, pelo incentivo aos estudos desde pequeno, por me apresentarem ao SENAI aos 14 anos e imenso apoio durante o Curso de Aprendizagem Industrial, Curso Técnico e Graduação.

A minha esposa, Tatiane, pelo apoio e incentivo durante as aulas da Pós-Graduação e realização deste trabalho.

Ao meu filho Pedro e meu irmão Rafael, que me ajudaram na montagem do protótipo.

A todos os meus professores, por compartilharem seu conhecimento de forma inspiradora.

Ao Prof. Paulo Sebastião Ladivez por toda a orientação e conselhos durante a realização deste trabalho.

Sobre os autores:

_____

[i] **PAULO ROBERTO DE SOUZA – Aluno**

Possui graduação em Engenharia de Controle e Automação pela Universidade Braz Cubas (2011), cursando atualmente a Pós Graduação em Automação Industrial pela Faculdade SENAI de Tecnologia Mecatrônica. Tem experiência na área de Engenharia de Automação, com ênfase em Projetos de Máquinas e Programação de CLPs, IHMs, Inversores de Frequência, etc. É técnico de automação industrial na empresa Silgan Dispensing Systems, responsável técnico pela área de automação da empresa.

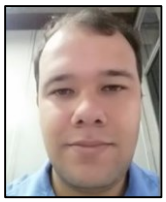
[ii] **PAULO SEBASTIÃO LADIVEZ - Orientador**

Possui graduação em Engenharia Elétrica pela Universidade Mogi das Cruzes (1984) com especialização em Tecnologias e Sistemas de Informação pela Universidade Federal do ABC (2013). Atualmente é professor da Faculdade SENAI de Tecnologia Mecatrônica, lecionando as disciplinas Projetos, Microcontroladores, Linguagem de Programação no Curso de Tecnologia em Mecatrônica Industrial e na Pós-Graduação em Automação Industrial. Tem experiência na área de Engenharia Eletrônica, com ênfase em Automação Industrial e Mecatrônica, atuando principalmente nos seguintes temas: Mecatrônica, Manufatura Digital, Redes Industriais, Automação Industrial, Microcontroladores e Controle.

[iii] CARLOS AURELIO GONZÁLEZ CARDOZO - Banca

Mestre em Engenharia Mecânica (UNICAMP), Engenheiro Industrial Mecânico (UNISANTA), Técnico Mecânico (ETE "Júlio de Mesquita"), Pós-Graduação - Formação Pedagógica para Docência de Ensino Médio Profissionalizante (UNIMEP). Experiência na área de Usinagem Convencional e a CNC, Instrutor Torneiro (SENAI), Técnico de Ensino do Curso Técnico em Mecatrônica Industrial, Professor do Curso Superior de Tecnologia em Mecatrônica Industrial, Professor do curso de especialização de CNC e FMS para TCTP, Professor do Curso de Pós-Graduação em Projeto CAD/CAM/CAE da Faculdade SENAI de Tecnologia Mecatrônica. Ministrou aulas de Machining Simulation no Colégio Diderot de Paris-França.

[iv] JOSÉ ROBERTO DOS SANTOS - Banca



Atualmente ministra aulas na pós-graduação de Indústria 4.0 e na graduação em Tecnologia em Mecatrônica na Faculdade SENAI de Tecnologia Mecatrônica, que fica no SENAI Armando de Arruda Pereira. Assessora também o Instituto SENAI de Tecnologia Metalmecânica em projetos industriais com foco na Indústria 4.0. Durante 9 anos ministrou aulas pelo SENAI-SP, nos cursos de técnico em eletroeletrônica, cursos de aprendizagem industrial eletricista de manutenção e mecânico de usinagem, além de Formação Inicial e Continuada (FIC) com cursos voltados a área de redes de computadores e programação, possui treinamento de Linux, cisco e Microsoft. Possui Pós-graduação na área de segurança da informação pela Uninove (2016), graduação em tecnologia da informação e bacharel em sistema da informação (2009), além de superior em Automação industrial. Tem experiência na área de Segurança da informação, administração de ambientes de redes Windows e Linux, automação indústria.