

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE CIÊNCIAS INTEGRADAS DO PONTAL
GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

KAREN AYUMI NISHIKAWA

**SIMULAÇÃO DE CORTE EM PROTÓTIPO DE MÁQUINA DE
CORTE BIDIMENSIONAL NÃO GUILHOTINADO**

ITUIUTABA

2017

KAREN AYUMI NISHIKAWA

SIMULAÇÃO DE CORTE EM PROTÓTIPO DE MÁQUINA DE CORTE BIDIMENSIONAL NÃO GUILHOTINADO

Trabalho de conclusão de curso em engenharia de
produção, Universidade federal de Uberlândia.

Área de concentração: Pesquisa Operacional e
Automação.

Orientador: Prof. Dr. Eng. Jorge Von Atzingen dos
Reis

ITUIUTABA

2017

Karen Ayumi Nishikawa

Simulação de corte em protótipo de máquina de corte bidimensional não guilhotinado

Trabalho de conclusão de curso em engenharia de produção,
Universidade federal de Uberlândia.

Ituiutaba, Dezembro de 2017

Aprovado em:

Banca Examinadora:

Prof. Dr. Eng. Jorge Von Atzingen dos Reis

Prof. Dr. Eng. Marcus Vinícius Ribeiro Machado

Prof. Dr. Eng. Ricardo Batista Penteado

Dedico este trabalho ao meu pai Sergio Shiroaki, minha mãe Lilian Yuriko e meu irmão Guilherme Yusuke e a toda minha família. Sem o apoio e ajuda eles, nada disso seria possível.

AGRADECIMENTOS

Ao prof. Dr. Eng. Jorge Von Atzingen dos Reis pela orientação de 2 anos e meio ao todo e pela oportunidade de realizar projetos de pesquisas. Ao Prof. Dr. Eng. Marcus Vinícius Ribeiro Machado pelo auxílio neste trabalho. Aos Prof. Dr. Eng. Gleyzer Martins e Prof. Dr. Eng. Antonio Álvaro de Assis Moura pela oportunidade de participar do projeto de ensino em Fenômeno de Transporte.

Aos meus amigos, em especial a Katherine e Amanda pelos apoios quando precisei, pela ajuda em português e aprendizagens fora das matérias que me proporcionaram. Agradecimento especial ao meu amigo Jeová Júnior pelos conhecimentos além do curso, ajuda em português e outros.

À empresa ferragista torneadora situada em Ituiutaba-MG por ter aberto as portas para coleta dos dados para aplicá-los nos dados reais e ao Gustavo Gabriel pela construção da máquina *corexy plotter*. A UFU pela oportunidade de aprender mais, FACIP pela infraestrutura para aprendizagem, CNPq pelo financiamento de iniciação científica.

Todos meus professores que me ensinaram além das matérias, aos meus colegas da sala. Agradeço minha mãe pelo apoio e incentivo nas horas de desânimo e cansaço, meu pai que apesar das dificuldades que tive, me manteve forte para seguir em frente.

Enfim, a todos que direta ou indiretamente que de alguma forma contribuíram para minha formação. Deixo aqui minha gratidão a todos.

“誰よりも三倍、四倍、五倍勉強する者、それが天才だ。”

野口英世（日本の細菌学者）

“Aqueles que estudam três vezes, quatro vezes, cinco vezes mais, é um gênio.”

Hideyo Noguchi, (Bacteriologista)

RESUMO

Para sobressair no mercado global cheio de concorrências, torna-se essencial a otimização do processo de toda cadeia de produção e enxugar ao máximo o custo operacional. Com a evolução das tecnologias, de automação e seu uso cada dia mais frequente, é necessário buscar o *kaizen* dos processos produtivos. Foi implementado o modelo matemático de problema do corte bidimensional não guilhotinado. A solução ocorreu utilizando o pacote de otimização GUROBI. Como alternativa ao modelo matemático do problema do corte foi implementada a heurística *first fit*. Como objetivo final, foi executada a simulação do corte no protótipo de corte bidimensional não guilhotinado na estrutura *corexy plotter* controlada por microcontrolador Arduino com a placa *Ramps* e *firmware Marlin*. O percurso para corte foi determinado com uma heurística construtiva gulosa do problema de carteiro chinês não direcionado e geração automática de *G-code* com um *software* desenvolvido em C++.

Palavras-chave: Problema de corte bidimensional, Problema do carteiro chinês, Protótipo de máquina de corte bidimensional.

ABSTRACT

In order to excel in the competitive market, it is essential to optimize the process of the whole production chain and to maximize the operational cost. With an evolution of technologies, automation and its use more and more frequent, it is necessary to seek the process of the productive processes. It was implemented the mathematical model of the non-guillotined two-dimensional cut problem. A solution in conjunction with the GUROBI optimization package. As an alternative to the mathematical model of the cut problem for the implementation of a first heuristic fit. As a final objective, a non-guillotined two-dimensional cut simulation was performed on the corexy plotter structure controlled by the Arduino microcontroller with the Ramps board and Marlin firmware. The course for the cut was determined by the electronic construction of the non-directed international card problem and automatic generation of G code with a program developed in C ++.

Keywords: Two-dimensional cut problem, Chinese postman problem, Two-dimensional cut machine prototype.

LISTA DE FIGURAS

Figura 2.1 – Etapas de construção	6
Figura 2.2 – Exemplo de cortes guilhotinado e não guilhotinado	7
Figura 2.3 – Orientação do objetos	8
Figura 2.4 – Exemplo de estágios de corte guilhotina	10
Figura 2.5 – Exemplo de níveis de corte guilhotina	10
Figura 3.1 – Fotos do protótipo de máquina de corte bidimensional.....	22
Figura 3.2 – Coordenadas cartesianas	23
Figura 3.3 – Fluxograma da heurística <i>first fit</i> para problema de corte bidimensional não guilhotinado	26
Figura 3.4 – Fluxograma do algoritmo para preencher a matriz distância	28
Figura 3.5 – Fluxograma da heurística construtiva gulosa para problema de carteiro chinês não direcionado	29
Figura 3.6 – Desenho da caixa da ferramenta	31
Figura 3.7 – Desenho explodido da caixa da ferramenta	32
Figura 4.1 – O desenho dos dados de saída, feito manualmente no <i>FreeCAD</i>	33
Figura 4.2 – Foto do papel desenhado pela <i>corexy plotter</i>	36

LISTA DE TABELAS

Tabela 2.1 – Resumo das variáveis para modelo de <i>Kantorovich</i>	8
Tabela 2.2 – Resumo das variáveis para corte com padrão	11
Tabela 2.3 – Resumo das variáveis para problema de corte bidimensional não guilhotinado múltiplas placas com dimensões variáveis	12
Tabela 2.4 – Resumo das variáveis para problema de carteiro chinês não direcionado	14
Tabela 3.1 – Resumo das variáveis para geração de matriz A_{ilwrs}	23
Tabela 3.2 – Resumo das variáveis para <i>quicksort</i>	25
Tabela 3.3 – Resumo das variáveis para heurística para problema de corte bidimensional não guilhotinado	25
Tabela 3.4 – Variáveis de heurística construtiva gulosa para problema de carteiro chinês	27
Tabela 3.5 – Comandos utilizados para geração do <i>G-code</i>	30
Tabela 3.6 – Os dados de entrada do problema de corte bidimensional	32
Tabela 4.1 – Os dados de entrada do problema de carteiro chinês não direcionado	33
Tabela 4.2 – Os dados de saída do problema de carteiro chinês não direcionado	34

LISTA DE SIGLAS

BPP	<i>Bin Packing Problem</i>
CAD	<i>Computer Aided Design</i>
CAM	<i>Computer Aided Manufacturing</i>
CNC	Controle Numérico Computatorizado
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico
ETC	ET Cetera
FACIP	Faculdade de Ciências Integradas do Pontal
FO	Função Objetivo
GB	<i>Giga Byte</i>
GHz	<i>Giga Hertz</i>
G-code	<i>Geometric code</i>
HDD	<i>Hard Disk Drive</i>
IoT	<i>Internet of Things</i>
IP	<i>Internetworking Protocol</i>
JIT	<i>Just In Time</i>
LED	<i>Light Emitting Diode</i>
min	Minutos
mm	Milímetro
NP	Não Polinomial
PCC	Problema de Carregamento de Contêineres
PCCD	Problema do Carteiro Chinês Direcionado
PCCM	Problema do Carteiro Chinês Misto
PCCND	Problema do Carteiro Chinês Não Direcionado
PCE	Problema de Corte e Empacotamento
PDCA	<i>Plan, Do, Check, Action</i>
RFID	<i>Radio-Frequency Identification</i>
TCC	Trabalho de Conclusão de Curso
TCP	<i>Transmission Control Protocol</i>
UFU	<i>Universidade Federal de Uberlândia</i>
un	Unidades
USB	<i>Universal Serial Bus</i>
Wi-Fi	<i>Wireless Fidelity</i>
2D	Duas Dimensões
3D	Três Dimensões

SUMÁRIO

1 INTRODUÇÃO	1
1.1 Objetivo	3
1.2 Justificativa	3
1.3 Estrutura do trabalho	4
2 FUNDAMENTAÇÃO TEÓRICA	5
2.1 Pesquisa operacional	5
2.2 Problema de corte	6
2.2.1 Problema de corte guilhotinado	9
2.2.2 Problema de corte não guilhotinado	11
2.2.3 Problema de corte bidimensional não guilhotinado	12
2.3 Problema de carteiro chinês	13
2.4 Indústria 4.0	15
2.5 Microcontrolador Arduino	18
2.6 Placa <i>Ramps</i> e <i>firmware Marlin</i>	18
2.7 Estrutura <i>corexy plotter</i>	19
3 METODOLOGIA	20
3.1. Materiais utilizadas	20
3.2 Implementação	22
3.2.1 Matriz A_{ilwrs}	23
3.2.2 Heurística para problema de corte bidimensional não guilhotinado	24
3.2.3 Heurística para problema de carteiro chinês não direcionado	26
3.2.4 Geração do <i>G-code</i> com o código em linguagem C++	30
3.3 Os dados iniciais	31
4 RESULTADO E DISCUSSÃO	33

4.1 Resultado geração de layout da corte utilizando ferramentas de pesquisa operacional	37
4.2 Resultado traçar rota para corte com problema de carteiro chinês	38
4.3 Resultado geração do <i>G-code</i> automático com linguagem C++	39
4.4 Resultado execução da simulação de corte no protótipo	39
5 CONCLUSÃO	40
REFERÊNCIAS	42
APÊNDICE A – Pseudocódigo algoritmo de ordenação <i>quicksort</i>	I
APÊNDICE B – Pseudocódigo heurística <i>first fit</i> para problema de corte bidimensional não guilhotinado	II
APÊNDICE C – Pseudocódigo geração da matriz distância	III
APÊNDICE D – Pseudocódigo heurística carteiro chinês não direcionado	V
APÊNDICE E – <i>G-code</i> gerado na aplicação	VII
APÊNDICE F.a – Foto do protótipo da máquina de corte bidimensional (vista perspectiva isométrica)	XI
APÊNDICE F.b – Foto do protótipo da máquina de corte bidimensional (vista superior)	XII
APÊNDICE F.c – Foto do protótipo da máquina de corte bidimensional (vista frontal e traseira)	XIII
APÊNDICE G.a – Desenho da caixa de ferramenta (vista frontal, lateral, superior e perspectiva isométrica)	XIV
APÊNDICE G.b – Desenho da caixa de ferramenta (explodido)	XV
APÊNDICE G.c – Desenho da caixa de ferramenta (detalhes da orelha)	XVI

1 INTRODUÇÃO

O presente trabalho abordará execução de corte em protótipo de corte bidimensional não guilhotinado controlado por microcontrolador. Serão abordados dois problemas clássicos de pesquisa operacional: problema de corte e problema do carteiro chinês, sendo eles para gerar o *layout* e a rota para corte, respectivamente. Posteriormente será abordado geração de *G-code* automático com linguagem C++. Estes problemas são considerados NP-*hard*, ou seja, não é possível que exista uma solução otimizada que seja feita em tempo polinomial (FLESZAR e HINDI, 2000).

A abordagem heurística é necessária devido ao elevado esforço computacional para a resolução dos problemas pertinentes à classe NP-*hard*. A heurística é uma técnica baseada em processos intuitivos que busca encontrar uma solução de boa qualidade dentro de um tempo computacional adequado sem garantir a otimalidade da solução ou a proximidade da solução ótima. A meta-heurística é um tipo de heurística que possui uma estrutura de refinamento da busca local visando escapar de soluções ótimas locais (REIS, 2013).

Devido ao infinito desejo humano e à escassez de matéria-prima torna-se essencial otimização dos insumos. A competitividade cada vez maior no mercado, exige da empresa um diferencial competitivo e obrigando assim, que a empresa otimize seus processos cada vez mais. Para auxiliar na tomada de decisões, surge a pesquisa operacional. A possibilidade de aplicações da pesquisa operacional é excepcionalmente variada, podendo ser aplicada nas áreas como manufatura, transportes, planejamento, construção, telecomunicações e muitos outros (HILLIER e LIBERMAN, 2010).

Inicialmente, as produções eram artesanais com trabalhadores polivalentes e na maioria das vezes, um trabalhador realizava todas as etapas de produção de um produto. O volume de produção era baixo e variabilidade dos produtos finais era alta. Com a utilização das máquinas a vapor na primeira revolução industrial, a utilização das máquinas foi se tornando cada vez mais comum, até que as fontes de energia passaram a ser petróleo e energia elétricas levando à segunda revolução industrial (AIRES, MOREIRA e FREIRE, 2017).

Posteriormente, surgiu a produção em massa nos Estados Unidos, onde os trabalhadores tinham tarefas específicas, alto volume de produção e baixa variedade dos

produtos, diminuindo o preço das mercadorias e tornando o país uma potência mundial (AIRES, MOREIRA e FREIRE, 2017).

Com a destruição do Japão no fim da segunda guerra mundial, o país sofreu com a falta de recursos, os japoneses viram então a necessidade de mudar o método de produção. Surgiu assim, a produção enxuta, conhecida como Toyotismo, onde a característica principal é a flexibilidade da produção. Um trabalhador realiza múltiplas funções, a produção e a variedade dos produtos são adaptados conforme as demandas, sempre mantendo o mínimo de estoque (MERLO e LAPIS, 2007).

Com o avanço tecnológico, popularização da internet e globalização, ocorreu a terceira revolução industrial. Uma modernização na produção, revolução na comunicação e mudança da vida como um todo. A internet se tornou extremamente importante, assim como automação das máquinas, mudando o conceito de produção (KOVÁCS, 2016).

De acordo com Groover (2011), a automação é o uso de equipamentos automatizados. No sistema automatizado, o processo é executado pelas máquinas sem a participação direta do homem. É implementado por meio de um programa de instruções elaborado pelo homem. A automação aumenta a produtividade, reduz os custos, minimiza o efeito da falta de trabalhadores, reduz rotinas manuais, aumenta a segurança do trabalho, melhora qualidade do produto e diminui o tempo de produção (GROOVER, 2011).

The robotic chef é um par de braços robotizados e promete reproduzir todas as funções das mãos humanas com a mesma velocidade, sensibilidade e movimento. Poderá ser operado por *smartphone*. Segundo site do *Moley Robotics*, a empresa espera lançar o produto no mercado consumidor em 2018. Nos hotéis situados em *Nagasaki* e *Chiba* no Japão, os clientes são recepcionados por robô que fala inglês, coreano, chinês e japonês. Os robôs levam os hóspedes até o quarto e o ambiente é repleto de tecnologias, sendo estas as principais atratividades do hotel.

Levando em conta as tendências das tecnologias e as exigências dos mercados atuais e futuros, em termos de produção, uma fábrica inteligente com uma máquina de corte automatizada, onde basta inserir as dimensões das placas e dos objetos no sistema, a partir dos dados, o *software* da máquina determina o melhor *layout* para corte e determina a menor rota para a execução da corte, seria ideal para o cenário. Então o presente trabalho realiza estudo

sobre otimização de corte bidimensional e trata da execução da corte no protótipo de máquina de corte bidimensional não guilhotinado semiautomático.

Ao realizar levantamento bibliográfico, não foram encontrados trabalhos similares. Pois maior parte das empresas fabricantes das máquinas não divulgam os códigos e funcionamentos detalhados. Trata-se de informações confidenciais a fim de evitar cópia ou surgimento de novas concorrências.

Com isto foram realizados levantamento bibliográficos separadamente para cada assunto: problema de corte bidimensional; problema de carteiro chinês; *G-code* e sobre os componentes da máquina similares a estrutura de impressora 3D. Realizou-se a visita na empresa ferragista e torneadora situada na cidade de Ituiutaba, a fim de obter conhecimento sobre as máquinas de cortes guilhotinado e não guilhotinado. Houve estudos sobre as estruturas similares a impressora 3D através das conversas com especialista da área.

1.1 Objetivo

Este trabalho tem como objetivo simulação de corte no protótipo da máquina de corte bidimensional não guilhotinado. Com isto este trabalho tem objetivo específico de:

- Geração de *layout* da corte utilizando ferramentas de pesquisa operacional;
- Traçar rota para corte com problema de carteiro chinês;
- Geração do *G-code* automático com linguagem C++;
- Execução da simulação de corte no protótipo com funcionamento conforme os conceitos da indústria 4.0.

1.2 Justificativa

Essa aplicação pode levar à construção de uma máquina de corte automatizada. Trazendo com isto, benefícios como: redução de desperdícios, redução do tempo de processamento, maior precisão do corte, diminuição da mão de obra, ambiente de produção semiautomático.

O assunto tratado no trabalho é pouco divulgado no meio acadêmico. Levando em conta a evolução das tecnologias, automação e seu uso cada dia mais frequente, é necessário buscar melhoria contínua dos processos produtivos (*kaizen*).

1.3 Estrutura do trabalho

O presente trabalho está estruturado da seguinte forma: o capítulo seguinte apresenta fundamentação teórica onde detalha pesquisa operacional, problema de corte, problema do carteiro chinês e apresenta o microcontrolador Arduino, placa *Ramps*, *firmware Marlin* e estrutura *corexy plotter*. No terceiro capítulo são mostrados os dados, processo e o método utilizado na aplicação. No quarto capítulo estão os resultados e discussão. Por último apresenta conclusões sobre o trabalho. Posteriormente são listados as referências utilizadas para o trabalho seguido de apêndices.

2 FUNDAMENTAÇÃO TEÓRICA

Para a melhor compreensão do trabalho é necessário o conhecimento de alguns problemas clássicos de pesquisa operacional existentes na literatura. O problema de corte, problema de carteiro chinês, microcontrolador Arduino, placa *Ramps*, *firmware Marlin* e estrutura *corexy plotter*. Esses temas serão abordados nesse capítulo e será apresentada a aplicação no próximo capítulo.

2.1 Pesquisa operacional

O termo Pesquisa Operacional (PO) é uma tradução direta do inglês *operational research*. O método teve seu início na segunda guerra mundial para lidar com os problemas existentes e ajudar a tomar melhores decisões para vencer o adversário. Uma definição de PO foi proposta na primeira página do periódico inglês *Operational Research Quarterly* em 1967. Consiste no desenvolvimento de métodos científicos de sistemas complexos, com a finalidade de prever e comparar estratégias ou decisões alternativas (ARENALES et al., 2007).

Segundo Batalha et al. (2008), PO é a aplicação de métodos científicos a problemas complexos, para auxiliar no processo de tomada de decisões. De acordo com Morabito et al. (2008), a pesquisa operacional é frequentemente aplicada nos problemas de logística (designação, roteamento, caminho mínimo, caixeiro viajante, localização de facilidades, carteiro chinês), planejamento, programação e controle de produção (sequenciamento, planejamento de produção, designação das máquinas) entre outros.

Uma das técnicas contidas na PO é a programação linear. A programação linear é o método que busca resolver problemas que podem ser expressos em equações e inequações lineares. São constituídos por Função Objetivo (FO) que indica o que se busca otimizar, seja um problema de maximização ou minimização, e de restrições, geralmente relacionadas com insumos, como quantidade de matéria prima disponível, tempo gasto em cada máquina, conforme especialidade do problema analisado (ARENALES et al., 2007).

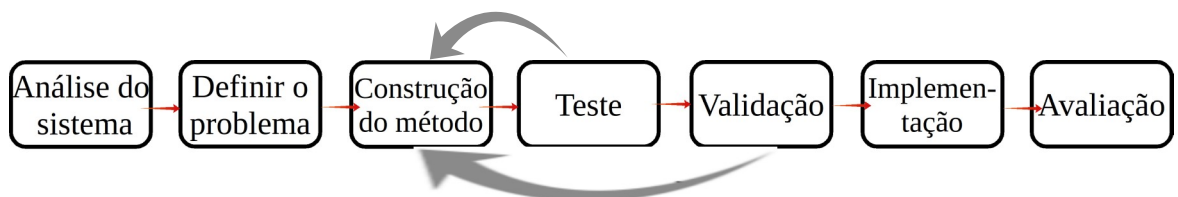
De acordo com Arenales et al. (2007), o modelo matemático é uma representação simplificada do problema real e trata de problemas de decisão, porém não substitui a tomada de decisões. Basicamente, a modelagem matemática de um problema consiste em determinar

um modelo capaz de representar a realidade da situação que será analisada. Este tipo de problema surgiu devido à necessidade do homem em tentar compreender e resolver problemas físicos existentes, facilitando e permitindo a implementação computacional de diferentes formas para determinar uma solução ótima que permita auxiliar nas tomadas de decisões.

A modelagem e resolução dos problemas consiste em um ciclo PDCA, a ferramenta de qualidade que busca solucionar problemas através dos processos *Plan* (planejar), *Do* (executar), *Check* (verificar) e *Action*, (agir). Inicialmente, deve-se analisar o sistema, posteriormente identificar e definir o problema, com isso, construir o método para resolução (modelo matemático) e realizar o teste do método. Caso necessite de adaptações deve-se voltar para o passo anterior, caso não, o modelo deve ser validado, implementado e avaliado.

Durante o processo, é comum sentir necessidades de adaptação ou mudança, então deve-se retornar e revisar o método novamente, quantas vezes for necessário. Em qualquer fase do processo, caso apareça um imprevisto ou for observada falha no modelo, deve-se voltar aos passos anteriores para refazê-los, até que chegue no resultado final. Como o processo é um ciclo, mesmo que chegue até o final, deve-se estar sempre atento para buscar melhoria ou adaptar as mudanças.

Figura 2.1 – Etapas de construção



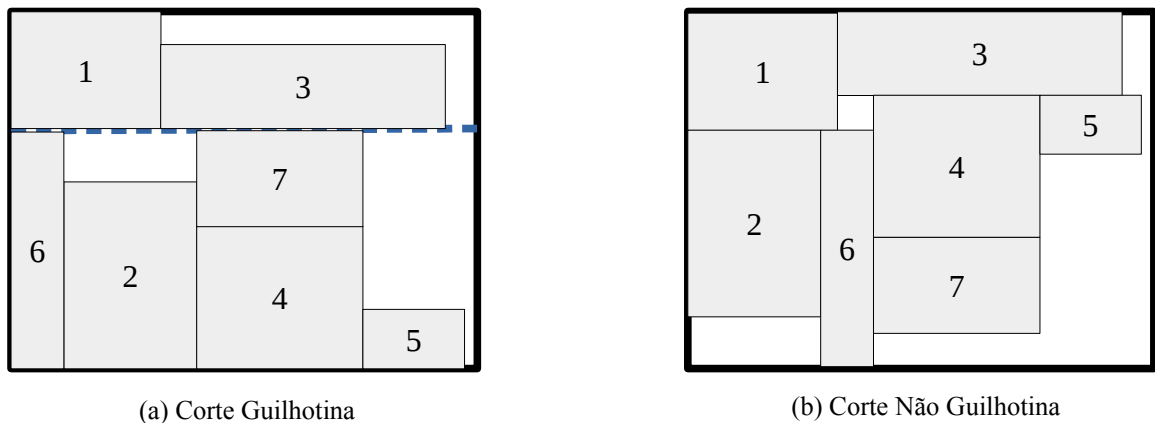
Fonte: Autoria própria.

2.2 Problema de corte

O problema de corte consiste em um problema de otimização cujo objetivo é determinar a melhor maneira de cortar objetos de uma certa matéria-prima (podendo ser bobina, placa, cubo, etc), atendendo todas as restrições. Pode ser maximização do número de objetos cortados, dado quantidade de matéria-prima disponível ou minimização das matérias-primas utilizadas, cortando toda demanda existente.

Existem dois tipos de corte, sendo eles guilhotinados e não guilhotinados. Corte guilhotinado consiste em corte de ponta a ponta, ou seja, corte são realizados de uma aresta até a outra aresta. O corte não guilhotinado não necessariamente é de ponta a ponta, como mostra a Figura 2.2.

Figura 2.2- Exemplo de cortes guilhotinado e não guilhotinado



Fonte: Autoria própria.

O problema de corte não guilhotinado além de problema de corte, pode ser utilizado para problemas de carregamento de contêineres (PCC), que é um caso particular de *bin packing problem* (BPP). Se pensarmos na matéria-prima como contêineres e os objetos a serem cortados como objetos a serem empacotados, o problema de corte se torna um problema de empacotamento ou carregamento de contêineres.

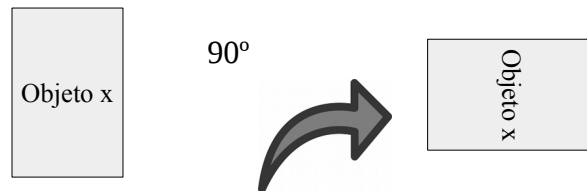
O problema de carregamento de contêineres também é um problema de otimização NP-hard. O problema trata da otimização para guardar objetos em um ou mais contêineres, atendendo restrições. Por conta das similaridades dos problemas, esses problemas são frequentemente agrupados e chamados de problemas de corte e empacotamento (PCE).

O problema poderá conter restrições como o tamanho dos contêineres, peso máximo suportado, restrições de agrupamento de entrega para facilitar na hora de descarregar objetos e ainda poderá conter restrições de acondicionamento de cada objeto, como carga máxima de empilhamento vertical, cuidado quanto a orientação da embalagem, estabilidade do produto para evitar quebras durante o transporte ou armazenamento.

Permitir rotação das peças em 90° é algo simples e eficiente para aprimorar o problema. Isto é, caso necessário, permitir mudar a orientação do objeto, já que o objeto

recortado será o mesmo como mostra na Figura 2.3. Considera-se que existem objetos de mesmo tamanho, porém com dimensões trocadas de comprimento e largura, ou seja, duplicar o número de objetos com as dimensões trocadas.

Figura 2.3 – Orientação do objetos



Fonte: Autoria própria.

O problema de corte unidimensional pode ser aplicado em indústrias que utilizam corte de matéria prima com largura fixa. O problema de corte bidimensional é amplamente utilizado em indústrias que utilizam o corte de placas metálicas, plásticos, tecidos entre outros que tenha duas dimensões que podem ser variadas. Enquanto o corte tridimensional se aplica, por exemplo, ao corte de madeiras, quando possui restrição de largura, comprimento e altura.

O seguinte modelo para problema de corte unidimensional é conhecido como modelo de *Kantorovich*. O Modelo pode ser aplicado em indústria de corte de papel, metal, tecido e outros. Ainda neste modelo, poderia acontecer de cada barra possuir seus comprimentos distintos, em vez de utilizar variável de decisão como binário, poderia utilizar número inteiro positivo. Os modelos são customizáveis conforme necessário. A Tabela 2.1 apresenta resumo das variáveis utilizadas.

Tabela 2.1 – Resumo das variáveis para modelo de *Kantorovich*

Variáveis	Descrição
n	Quantidade da matéria-prima disponível.
m	Quantidade de objetos.
L	Comprimento da matéria-prima.
b_j	Demanda do objeto do tipo j .
l_j	Comprimento de cada objeto j , onde $j=1, \dots, m$.

x_{ij}	Representa número de vezes que o item j é cortado na matéria prima i .
y_i	Variáveis de decisão. Recebe 1 se a matéria prima i é utilizada, 0 caso contrário

Fonte: Adaptado de Arenales et al. (2006).

$$\text{Minimizar } \sum_{i=1}^n y_i \quad [1.1]$$

$$\sum_{i=1}^n x_{ij} \geq b_j, \quad \forall j=1, \dots, m \quad [1.2]$$

$$\sum_{j=1}^m l_j x_{ij} \leq L y_i, \quad \forall i=1, \dots, n \quad [1.3]$$

$$x_{ij} \geq 0 \quad \forall i=1, \dots, n \quad \forall j=1, \dots, m \quad [1.4]$$

$$y_i \in [0, 1], \quad \forall i=1, \dots, n \quad [1.5]$$

$$x_{ij} \in \mathbb{Z} \quad \forall i=1, \dots, n \quad \forall j=1, \dots, m \quad [1.6]$$

Equação [1.1]: Função Objetivo. Minimizar as matérias primas utilizadas.

Equação [1.2]: Restrição 1. Garantia para atender a demanda.

Equação [1.3]: Restrição 2. Garante que comprimento total de objetos cortados não exceda o comprimento da matéria prima.

Equação [1.4]: Restrição 3. Indica o tipo de variável.

Equação [1.5]: Restrição 4. Indica o tipo de variável.

Equação [1.6]: Restrição 5. Indica o tipo de variável.

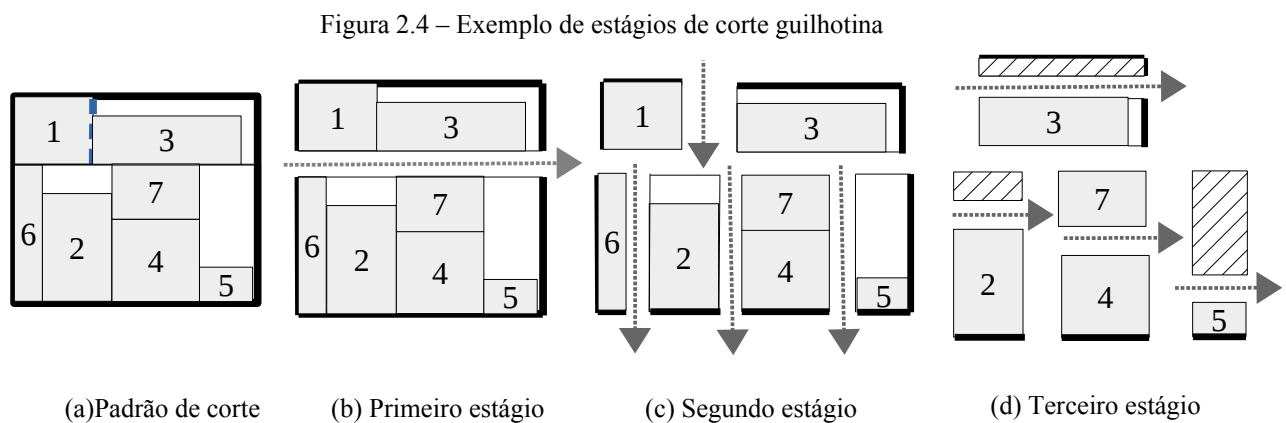
2.2.1 Problema de corte guilhotinado

O corte guilhotinado é o corte ponta a ponta como mostrado na Figura 2.2 (b). O corte é realizado de uma aresta da matéria prima até a aresta do lado oposto da matéria prima. Aplicáveis nas indústrias que utiliza corte de matéria prima de metal, de madeira, espelho, papel para produção de objetos, onde utilizam máquina guilhotina. O estudo deste tipo de corte é relevante por ser bastante comum nas indústrias.

O corte guilhotinado bidimensional aparece em empresas ferragistas por exemplo, para cortar uma placa de metal para produção de lambril, caixa de ferramentas, batente. Também na confecção dos móveis de madeira, para produzir um armário por exemplo. Na

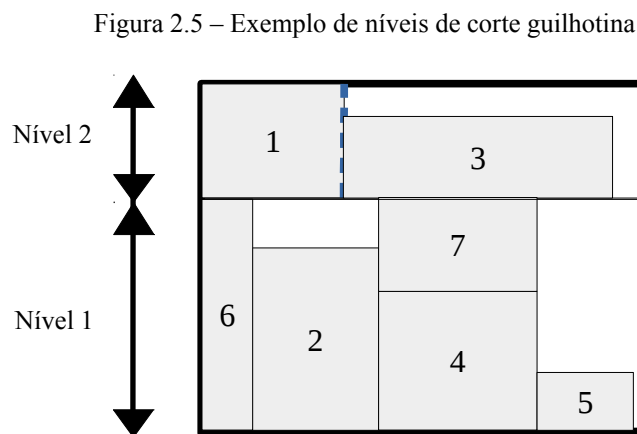
produção do armário, pode precisar de algumas partes em vidros ou espelhos, que entram no caso de corte bidimensional.

No corte guilhotinado, cada mudança na orientação da placa é chamado de estágios. Como o processo de mudança de orientação geralmente é realizado manualmente, toma tempo. Por isso se deve minimizar a quantidade de estágios. Se realizar corte em apenas uma direção, ele é considerado corte com 1 estágio, se mudar de direção uma vez, é chamado de corte com 2 estágios, e se após corte de segundo estágio for necessários separar a placa em parte menores, será considerado corte de 3 estágios. E assim por diante, como pode ser visto na Figura 2.4.



Fonte: Autoria própria.

Os níveis de cortes consistem na quantidade das tiras que serão formadas. Como pode ser visto na Figura 2.5. Também são chamados de faixas ou tiras.



Fonte: Autoria própria.

O problema de corte é dito restrito quando possui limitações para o número máximo de vezes que um tipo de item pode ser cortado a partir de um objeto. Caso contrário, o problema é dito irrestrito (MORABITO e PUREZA, 2007).

Porém, há casos em que o corte não deve ser de ponta a ponta, por exemplo quando se deseja recortar a geometria de um objeto, ou quando máquina de corte utiliza corte a plasma, se torna mais vantajoso trabalhar com corte não guilhotinado, que será apresentado no tópico seguinte.

2.2.2 Problema de corte não guilhotinado

No corte não guilhotinado, os cortes são realizados ortogonalmente a um lado da matéria-prima e não necessariamente são de ponta a ponta. Como alternativa ao modelo de corte unidimensional de *Kantorovich*, o *Gilmore* e *Gomory* (1961, 1963) propôs modelo matemático para o problema de corte, utilizando padrões de corte.

Neste modelo, a geração do padrão deve ser realizada anteriormente, atendendo restrições como dimensões, volume e peso. Como ele depende do padrão de corte, pode ser utilizado para corte unidimensional, bidimensional, n-dimensional e pode ser utilizado para corte não guilhotinado. Para isso, basta adaptar o código e padrão de corte. A Tabela 2.2 apresenta as variáveis utilizadas para cortes com padrão.

Tabela 2.2 – Resumo das variáveis para corte com padrão

Variáveis	Descrição
n	Número de padrões de corte.
b_j	Demanda do objeto do tipo i .
A_{ij}	Matriz de padrão de corte. Número do objeto do tipo i no padrão j .
y_i	Variáveis de decisão. Indica número de objetos cortados conforme padrão j .

Fonte: Adaptado de Arenales et al. (2006).

$$\text{Minimizar } \sum_{i=1}^n y_i \quad [2.1]$$

$$\sum_{i=1}^n A_{ij} y_i = b_j \quad \forall j = 1, \dots, n \quad [2.2]$$

$$y_i \in \mathbb{Z}_+^n \quad \forall i = 1, \dots, n \quad [2.3]$$

Equação [2.1]: Função Objetivo. Minimizar matéria prima utilizada.

Equação [2.2]: Restrição 1. Garantia para atender a demanda.

Equação [2.3]: Restrição 2. Indica o tipo de variável.

2.2.3 Problema de corte bidimensional não guilhotinado

Neste modelo, considerando que a placa e os objetos são retangulares. Os cortes são ortogonais a um lado da placa. O modelo pode ser aplicado em indústria que utiliza corte de placa ou objeto com terceira dimensão fixa. Modelo ainda pode ser utilizado para problemas de carregamento de contêineres. A Tabela 2.3 apresenta as variáveis utilizadas para corte bidimensional não guilhotinado.

O modelo seguinte já considera múltiplas placas com tamanhos variáveis. Pois há casos em que as placas disponíveis possuem dimensões variadas e visa determinar como cada objeto deve ser cortado em qual placa. O seguinte modelo é aplicado neste caso. Assim, aumentando a possibilidade de otimização e complexidade do problema.

Tabela 2.3 – Resumo das variáveis para o problema de corte bidimensional não guilhotinado com múltiplas placas e dimensões variáveis

Variáveis	Descrição
n	Quantidade dos objetos.
m	Quantidade das placas disponíveis.
X_j, Y_j	Dimensões da placa j .
l_i, w_i	Dimensões do objeto i , sendo $i=1, \dots, n$.

A_{ijlws}	Matriz de padrão de corte.
x_{ilw}	Variáveis de decisão.

Fonte: Adaptado de Arenales et al. (2006).

$$\text{Maximizar } \sum_{i=1}^n \sum_{j=1}^m \sum_{l \in X} \sum_{w \in Y} x_{ijlw} \quad [3.1]$$

$$\sum_{i=1}^n \sum_{j=1}^m \sum_{l \in X_j} \sum_{w \in Y_j} A_{ijlws} x_{ijlw} \leq 1, \quad r \in X_j, \quad s \in Y_j \quad [3.2]$$

$$\sum_{l \in X_j} \sum_{w \in Y_j} x_{ijlw} \leq 1, \quad i=1, \dots, n \quad j=1, \dots, m \quad [3.3]$$

$$x_{ijlw} \in [0,1], \quad \forall i=1, \dots, n, \quad j=1, \dots, m, \quad l \in X, \quad w \in Y \quad [3.4]$$

Equação [3.1]: Função Objetivo. Maximizar quantidade de objetos cortados.

Equação [3.2]: Restrição 1. Garante não ocorrência de sobreposição.

Equação [3.3]: Restrição 2. Garante que objeto seja cortado uma única vez.

Equação [3.4]: Restrição 3. Indica o tipo de variável.

2.3 Problema de carteiro chinês

No problema de corte bidimensional, supondo certa quantidade das peças retangulares para serem cortados, a quantidade total dos nós e das arestas será de 4 vezes número dos objetos, alguns deles podendo coincidir, dependendo da forma como devem ser cortados. Considerando a placa como uma cidade, cada nó uma esquina e cada aresta como rua. Para realização de corte, o cabeçote de corte precisa passar por todas as arestas. Este problema pode ser traduzido para o problema de carteiro chinês.

O problema de carteiro chinês é um problema de otimização que objetiva cobrir todos os arcos de um grafo com uma rota. Consiste em determinar o caminho para minimizar o custo, distância ou tempo de travessia para um carteiro. Devem passar por todos arcos pelo menos uma vez. A rota do carteiro permite a repetição das arestas, caso necessário (KONOWALENKO, 2012).

Ainda de acordo com Konowalenko (2012), o problema pode ser aplicado para entrega das cartas e encomendas, coletas de lixo, roteamento de ônibus escolar, entre outros.

Problemas de carteiro chinês, geralmente são classificados em três tipos: problema do carteiro chinês não direcionado (PCCND); problema do carteiro chinês direcionado (PCCD) e problema do carteiro chinês misto (PCCM).

No PCCND, as arestas não possuem restrição quanto a sentido do fluxo (mão dupla). No PCCD os arcos só tem um sentido (mão única). Já no PCCM são utilizados quando existem os dois tipos de fluxo em um grafo. A Tabela 2.4 apresentam as variáveis utilizadas para problema de carteiro chinês não direcionado.

Tabela 2.4– Resumo das variáveis para problema de carteiro chinês não direcionado

Variáveis	Descrição
n	Quantidade dos nós.
m	Quantidade de arestas.
x_{ij}	Variável de decisão. Número de vezes que o nó (i, j) é atravessado
d_{ij}	Distâncias de nó i para nó j

Fonte: Adaptado de Arenales et al. (2006).

$$\text{Minimizar } \sum_{i=1}^n \sum_{j=1}^n d_{ij} x_{ij} \quad [4.1]$$

$$\sum_{j=1}^n x_{ij} - \sum_{j=1}^n x_{ji} = 0, \quad \forall i = 1, \dots, n \quad [4.2]$$

$$x_{ij} + x_{ji} \geq 1, \quad \forall i = 1, \dots, n \quad \forall j = 1, \dots, n \quad [4.3]$$

$$x_{ij} \in \mathbb{Z}_+ \quad \forall i = 1, \dots, n \quad \forall j = 1, \dots, n \quad [4.4]$$

Equação [4.1]: Função Objetivo. Minimiza a distância percorrida.

Equação [4.2]: Restrição 1. Garante que entrada e saída do nó sejam iguais.

Equação [4.3]: Restrição 2. Garante que passe por todas as arestas pelo menos uma vez.

Equação [4.4]: Restrição 3. Indica o tipo de variável.

2.4 Indústria 4.0

A indústria 4.0 ou a 4ª revolução industrial pode ser considerada o futuro do sistema de produção. Onde a tendência é automatização da cadeia de produção, tornando cada vez menor a dependência humana no processo de produção e de prestação de serviços. Se implementado de forma correta, traz grande benefício. Como algumas vantagens podemos citar a redução do custo do produto, maior precisão (aprimorar qualidade), redução do tempo de produção (*leadtime*) e com isso aumento de produtividade, produção flexível, redução dos erros, satisfazendo então, os cinco objetivos de desempenho: qualidade, rapidez, confiabilidade, flexibilidade e custo (GROOVER, 2011; SILVA, FILHO e MIYAGI, 2015).

A implementação da tecnologia ainda possui certa restrição, por conta da falta de tecnologia e mão de obra adequada, falta de segurança ou alto valor de investimento inicial. Todavia é evidente que a tecnologia continuará evoluindo e como a indústria 4.0 é uma grande tendência, futuramente a implementação poderá deixar de ser um diferencial e se tornar uma exigência para sobrevivência no mercado (SILVEIRA e LOPES, 2017).

Segundo Rubmann et al. (2015), a nova tecnologia industrial digital é alimentada por nove avanços tecnológicos fundamentais. Nesta transformação, os sensores, as máquinas, as peças e o sistema de tecnologia e informação serão conectados ao longo da cadeia, onde os componentes interagirão uns com os outros através dos protocolos de comunicação baseados na internet. Sendo os nove pilares da indústria 4.0 segundo Rubmann et al. (2015):

- 1) Análise de dados e *big data*;
- 2) *Cloud computing*;
- 3) *Internet of Things* (IoT);
- 4) Manufatura aditiva;
- 5) Realidade aumentada;
- 6) Robôs autônomos;
- 7) Segurança cibernética;
- 8) Simulação;
- 9) Sistema de interligação horizontal e vertical.

A análise de dados, ou do inglês *big data*, se refere à grande quantidade de dados armazenados nos sistemas. A tecnologia consegue otimizar o processo industrial, economizando energia, melhorando serviços e equipamentos, facilitando tomadas de decisões em tempo real. Estes dados, podem ser armazenados em uma nuvem, onde ficam disponíveis

para todos que se relacionam com a produção, permitindo atualização em tempo real, com utilização da internet, proporcionando maior agilidade (SILVEIRA e LOPES, 2017).

O *cloud computing* ou computação nas nuvens, permite a reação rápida da empresa, em questão de segundos, permitindo rapidez na produção, maior flexibilidade, produção personalizada, exatamente por permitir que os dados sejam armazenados em uma nuvem, que pode ser acessada de qualquer lugar do mundo, desde que se tenha acesso à internet. (SOUZA, MOREIRA e MACHADO, 2010)

A *Internet of Things* (IoT), Internet das Coisas, conecta os componentes (como ambiente, máquina, objetos, usuários), permite a coleta e troca de dados através da internet utilizando tecnologias padrão. Permite descentralização da análise e da tomada de decisão. A tecnologia já é bastante comum no nosso dia a dia, como por exemplo, a câmera de segurança com visualização no tempo real, a *smart TV* para assistir *netflix* ou *youtube* e conectar ao *facebook*, lâmpada, sistema de segurança ou ar-condicionado que permite controle via *smartphone*. Na indústria 4.0 a tecnologia interligará os componentes que atualmente estão separados, melhorando o sistema e ampliando a usabilidade (LEMOS, 2012).

A manufatura aditiva será amplamente utilizada na indústria 4.0. Bastante empregada na prototipagem rápida, trata-se de produção a partir de camadas de materiais sobrepostas. Oferece vantagem na construção dos objetos, possibilitando produzir produtos personalizados e complexos sobre encomenda. Reduz desperdício dos materiais, já que os materiais são depositados conforme necessidade. Possui grande versatilidade, podendo ser utilizado nos casos de objetos com formas complexas, que impossibilite ou inviabiliza utilização de outras tecnologias de fabricação. Com a tecnologia é possível produzir objetos, próteses dentárias, pele e ossos humanos, comida, casas, carros, entre outros (SOUZA e ULBRICH, 2009)

A realidade aumentada é uma tecnologia relativamente nova, está em construção atualmente. Trata-se da junção de um mundo virtual com o real. Possibilita informações em tempo real para tomada de decisões e procedimentos de trabalho. Permite ensaiar ou simular aquilo que está em desenvolvimento de forma clara e real (KIRNER e SISCOOTTO, 2007).

Os Robôs mais autônomos, flexíveis e cooperativos entre eles, sendo que a tendência é se tornarem capazes de interagir com outras máquinas e com os humanos (robótica colaborativa), se tornarem mais baratos, maior variedade e mais capacitados, permitindo produção automatizada, flexível e com custo menor.

Desta forma, se torna indispensável o cuidado com a segurança dos dados. O espaço cibernético é um ambiente virtual. A segurança cibernética surge para que a integração dos dados seja segura, confiável e isenta de falhas, este será um dos desafios para popularização da revolução (CANONGIA e JUNIOR, 2009)

A simulação computacional, chamado de *digital twin*, a simulação deixará de ser tratada em partes separadas e será tratada na indústria como um todo, auxiliando no desenvolvimento e aperfeiçoamento de processos e aproveitará os dados em tempo real para refletir o mundo físico em modelo virtual, permitindo assim, diferentes análises, aumento da produtividade, detecções rápidas de erro e otimização do tempo.

Por último, para atingir um bom nível de automação é fundamental elaboração de sistema para que possa se comunicar com vários dispositivos de controle em vários níveis. Isto é, um sistema integrado, sendo ele vertical e/ou horizontal (PRUDENTE, 2011). O sistema de interligação horizontal e vertical interliga os sistemas de toda cadeia (redes de comunicação, computadores, máquinas, fábricas) para acabar com a perda de dados, otimizar o tempo, maior interação entre os componentes, permissão para influenciar mutuamente e possibilidade de acesso à informação para todos do ecossistema.

Segundo Rubmann et al. (2015), a indústria 4.0 permite uma resposta mais rápida às necessidades do cliente do que é possível hoje. Melhora a flexibilidade, velocidade, produtividade e qualidade do processo de produção. Isso permitirá um novo nível de personalização em massa à medida que as indústrias investirem mais em tecnologias da indústria 4.0 para aprimorar e personalizar as ofertas.

A indústria 4.0 torna possível os produtos personalizados para produção em massa. Receber o pedido do cliente através de um site ou aplicativo, a informação é armazenado em nuvem, o sistema analisa todos pedidos, organiza, otimiza e envia os comandos para as máquinas, que por sua vez produz conforme os pedidos. Feito isso, o cliente é informado pelo aplicativo que o produto já está disponível para sua retirada, ou que foi despachada ao endereço cadastrado, tudo isso praticamente sem intervenção humana no processo.

2.5 Microcontrolador Arduino

Segundo Comer (2011), o Arduino é um pequeno computador programável para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele. O Arduino tem se destacado por conta de baixo custo, versatilidade que o mesmo oferece, e por conta da facilidade tanto na programação, quanto na utilização. É bastante utilizada na prototipagem de várias áreas. Como automação industrial, automação residencial, controle robótica entre outros.

Permite conexão com vários tipos de *sheield*, LEDs, motores, sensores (umidade, pressão, distância, temperatura, etc), módulo *ethernet*, monitor, botão e muitas outras, dependendo da necessidade. Possui grande possibilidade de aplicação.

O Arduino é uma plataforma que foi construída para promover a interação física entre o ambiente e o computador utilizando dispositivos eletrônicos de forma simples e baseada em *softwares* e *hardwares* livres (CAVALCANTE, TAVOLARO e MOLISSANI, 2011).

Para programação, utiliza-se o IDE do Arduino, onde a linguagem é baseada em linguagem C. O mesmo permite escrever uma instrução de execução e realiza gravação para o Arduino via cabo USB.

2.6 Placa *Ramps* e *firmware Marlin*

A placa *Ramps* é um *sheield* frequentemente utilizado nas máquinas de controle numérico computadorizado (CNC). A CNC permite o controle das máquinas simultaneamente, com o código específico com os comandos.

A combinação de microcontrolador Arduino, *firmware Marlin* e placa *Ramps* 1.4 é bastante comum nas máquinas de impressão 3D. Segundo Souza e Ulbrich (2009), além da máquina-ferramenta, a aplicação de CNC permite controlar equipamentos para desenho, máquinas para montagens, máquina de corte a laser, entre outros.

O *Computer Aided Manufacturing* (CAM) ou traduzindo, a manufatura auxiliada por computador é um sistema generalizado para programas que podem definir a trajetória e rotinas que a máquina CNC executa. O *Geometric code* (*G-code*) é a linguagem utilizada por controladores para se comunicar com as impressoras ou máquinas com estruturas similares.

Usando o *G-code*, o controlador diz à máquina a que velocidade mover, a qual coordenada mover, qual temperatura manter e outros comandos.

Um *firmware* é conjunto de instruções operacionais programadas diretamente no hardware de equipamentos eletrônicos, ou seja, *software* que pode ser instalado diretamente no *hardware*.

O *Marlin* é um *firmware* para impressoras 3D de código aberto. Segundo o site do fornecedor, a plataforma de referência para *Marlin* é o Arduino Mega 2560 com *Ramps* 1.4. O código é customizável conforme as necessidades do usuário. Ele coordena sensores, luzes, *display*, botões, motores, acesso a cartão SD e outros componentes envolvidos na estrutura da impressora 3D ou de algo similar. O *Marlin* fica disponível na internet, após *download* cada usuário realiza suas adaptações, carrega no Arduino e o mesmo controla a estrutura.

2.7 Estrutura *corexy* *plotter*

A *corexy* é uma estrutura particular de mesa coordenada. Tem a função de posicionar nas coordenadas desejadas para determinado objetivo, por exemplo para perfuração, deposição de materiais e fresa.

Sua funcionalidade se baseia no sistema de coordenadas do plano cartesiano. O sistema de coordenada cartesiano foi inventado por um francês filósofo e matemático René Descartes. Formado por dois eixos x e y, com objetivo de localizar pontos.

O *plotter* é um sistema para deposição de pigmentos (tinta, giz, grafite), para criar o desenho em materiais bidimensionais (papel, madeira, tecido). A estrutura *corexy plotter* é uma estrutura parecida com a estrutura de impressora 3D, mesa de fresa e corte a plasma, por exemplo. Porém, em vez de depositar materiais para criar um objeto ou cortar objetos, a *plotter* cria a forma com deposição de pigmentos.

3 METODOLOGIA

Foram realizadas pesquisas bibliográficas em livros, artigos, sites, monografias, dissertações e teses, a fim de obter propriedade para o embasamento teórico do tema. Realizou-se a coleta de dados na empresa ferragista e torneadora, através de entrevistas com os operadores das máquinas e com pessoas da área administrativa da empresa para testar o protótipo de máquina de corte com os dados similares a real. E por fim, foi programando o protótipo da máquina de corte bidimensional. O modelo matemático utilizado foi proposto por Arenales et al. (2006). O processo de programação computacional e execução do corte é realizado nas seguintes etapas:

- i. Leitura dos dados do corte do arquivo *txt*;
- ii. Resolução do problema de corte com heurística *first fit*;
- iii. Exportação dos resultados do corte no arquivo *txt*;
- iv. Determinar a rota para corte com heurística construtiva gulosa;
- v. Exportação dos resultados da rota no arquivo *txt*;
- vi. Gerar arquivo no formato *G-code* através da linguagem C++;
- vii. Armazenar arquivo *G-code* no cartão de memória *Security Digital* (SD) manualmente;
- viii. Comando manual na máquina *corexy plotter* para execução da simulação da corte.

3.1. Materiais utilizados

Partes computacionais:

- Intel *Core* i3-3217U, memória 4GB, sistema operacional Linux Ubuntu 16.04 LTS 64-*bits*;
- Intel Xeon E5-1620, memória 48GB, sistema operacional Linux Ubuntu 16.04 LTS 64-*bits*;
- *Software* GUROBI versão 7.5.1;
- *Software* Arduino IDE;
- *Firmware* Marlin;
- Cartão de memória *micro SD Kingston* 4GB e seu adaptador.

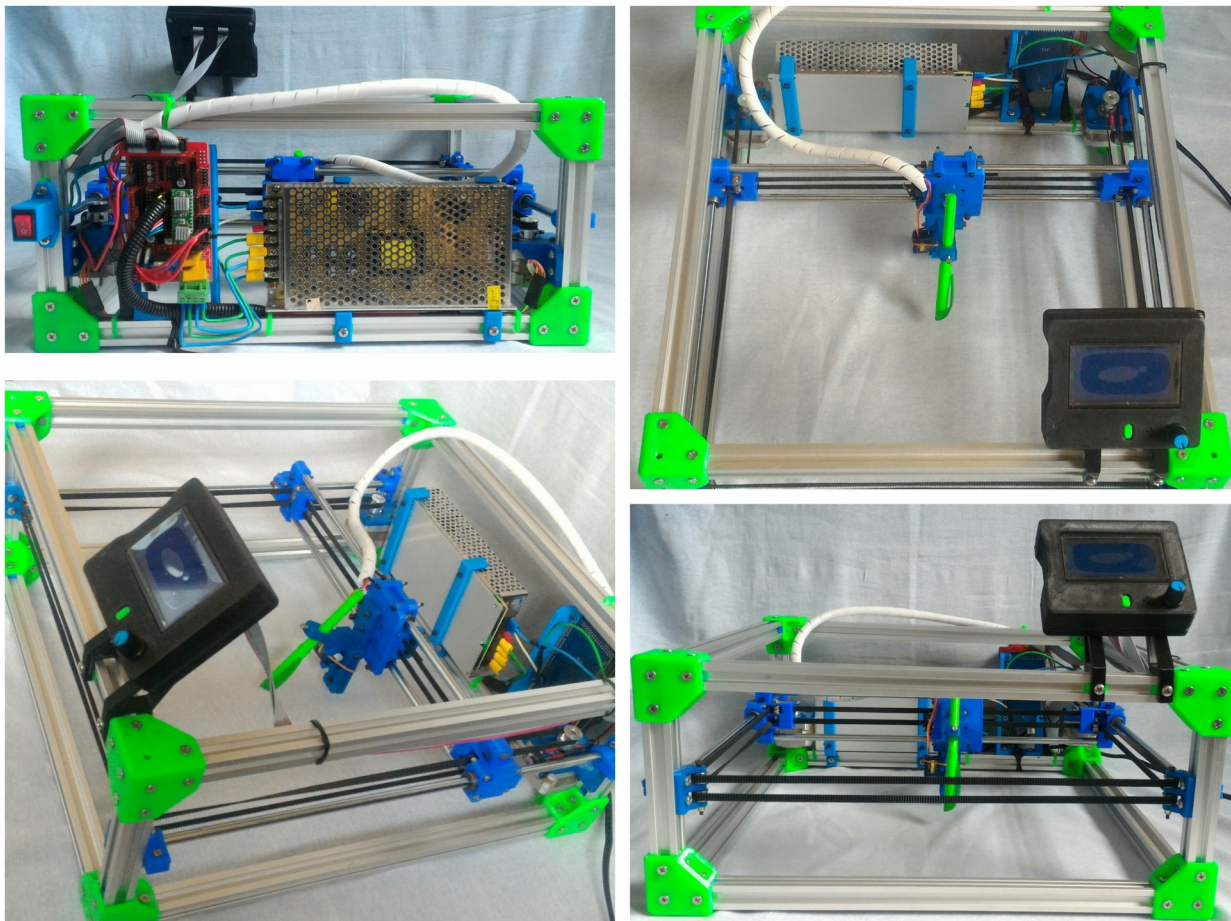
Estrutura da *corexy plotter* para simulação da máquina de corte:

- Arduino MEGA 2560;
- Placa *Ramps* 1.4;
- Barra de alumínio para estrutura eixo X 20x20 385 mm (4un);

- Barra de alumínio para estrutura eixo Y 20x20 420 mm (4un);
- Alumínio para estrutura eixo Z 20x20 190 mm (4un);
- Aço inox para eixo X 8x405 mm (2un);
- Aço inox para eixo Y 8x420 mm (2un);
- Rolamento linear LM8UU (8un);
- Rolamento F623ZZ (16un);
- Polia GT2 (2un);
- Correia GT2 6 mm (4un);
- Motor de passo NEMA 17 (2un);
- *Driver* motor de passo A4988 (2un);
- Servo motor SG90 9G;
- Sensor fim de curso KM11-3Z-3 (2un);
- Botão *on/off*;
- Fonte 10A;
- *Display full graphic*;
- Acessório de plástico impresso em impressora 3D (ligamento, suporte para estrutura);
- Parafusos e porcas;
- Jumperes;
- Caneta simulando cabeçote da corte a plasma e papel A4 simulando placa a ser cortada.

Na Figura 3.1 se encontram as fotos do protótipo de máquina de corte bidimensional. Superior esquerda é a vista traseira, superior direita a vista superior, inferior esquerda a vista perspectiva isométrica e inferior direita vista frontal. Uma versão ampliada do desenho está anexada no APÊNDICE F.

Figura 3.1 – Fotos do protótipo de máquina de corte bidimensional



Fonte: Autoria própria.

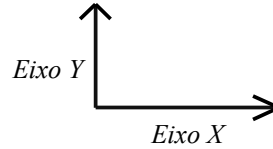
3.2 Implementação

Inicialmente, o código principal lê os dados contidos em arquivo no formato *txt*, criado pelo usuário previamente. Dentro da função de corte, são chamadas as funções do problema de carteiro chinês para determinar rota otimizada para execução do corte. Por fim, é chamada a função para gerar o *G-code*. Em cada passo, o código gera um arquivo de saída no formato *txt* contendo as respostas, para facilitar a compreensão do resultado.

A implementação do código principal, a resolução do problema de corte com a heurística *first fit*, a resolução do problema de carteiro chinês com a heurística construtiva gulosa e geração de *G-code* foi feita em linguagem C++ e a resolução de problema de corte bidimensional não guilhotinado por modelo matemático, com *software* de otimização

GUROBI. No presente trabalho adota-se seguinte modelo de coordenadas cartesianas em que a horizontal é x e vertical y , como na Figura 3.2.

Figura 3.2 – Coordenadas cartesianas



Fonte: Autoria própria.

3.2.1 Matriz A_{ilwrs}

Na geração do padrão de corte foi construída a matriz A_{ilwrs} do tipo binário. A Tabela 3.1 apresenta o resumo das variáveis utilizadas para geração da matriz.

Tabela 3.1 – Resumo das variáveis para geração de matriz A_{ilwrs}

Variáveis	Descrição
n	Quantidade de Objetos.
X, Y	Dimensões das placas.
$comprimento_i, largura_i$	Dimensões do objeto i , sendo $i=1, \dots, n \in \mathbb{Z}$.
l, w	Variáveis para laço de repetição, sendo $\forall l = 1, \dots, X \in \mathbb{Z}$ $\forall w = 1, \dots, Y \in \mathbb{Z}$
r, s	Posições da placa, sendo $\forall r = 1, \dots, X \in \mathbb{Z}$ e $\forall s = 1, \dots, Y \in \mathbb{Z}$

Fonte: Adaptado de Arenales et al. (2006).

A lógica utilizado para geração foi da seguinte forma: $A_{ilwrs}=1$ se

$$(r < l + largura_i) \text{ e } (s < w + comprimento_i) \quad [5.1]$$

$$\text{e } (r \geq l) \text{ e } (s \geq w) \quad [5.2]$$

$$\text{e } (l + comprimento_i \leq X) \text{ e } (w + largura_i \leq Y) \quad [5.3]$$

As condições [5.1] verificam se a soma de posição atual com a dimensão do objeto é menor que as dimensões da placa. As condições [5.2] diz que as dimensões do objeto devem

estar dentro das dimensões da placa. As duas últimas condições [5.3] verificam se a coordenada da posição atual somada às dimensões referente às do objeto i , não ultrapassa a dimensões da placa. Isto é, se a peça cabe na placa, a partir da coordenada atual. $A_{ilwrs}=0$, caso contrário.

Supondo que o objeto i com dimensões $comprimento_i$ e $largura_i$ esteja na primeira posição viável da placa. Por exemplo posição (1,1), e o objeto tenha dimensões 2x2, desta forma as posições entre (1,1) e (3,3) receberão o valor 1 e as demais posições da placa (matriz) receberão o valor 0.

Se, $A_{ilwrs}=1$ significa que o objeto i ocupará a posição (r,s) da placa. Se, $A_{ilwrs}=0$, a posição não será ocupada por objeto i . Antes de passar os dados para o problema de carteiro chinês, deve-se excluir os nós coincidentes, para evitar passar pelas mesmas arestas várias vezes.

3.2.2 Heurística para problema de corte bidimensional não guilhotinado

Aumentando as dimensões da placa e/ou o número de objetos, não foi possível executar o modelo matemático de corte bidimensional, apresentando falha de segmentação. A matriz A_{ilwrs} precisa de uma quantidade de posições de memória muito grande e junto com as demais variáveis acarreta um erro em tempo de execução. Com isso foi implementada a heurística *first fit* para resolução da corte na aplicação.

A heurística basicamente ordena os objetos em ordem decrescente e vai encaixando os objetos nas placas. Após leitura dos dados de entrada, o código multiplica a largura pelo comprimento para obter a área de cada objeto. Posteriormente, com o algoritmo de ordenação por comparação *quicksort*, os dados são ordenados de forma decrescente. O pseudocódigo do *quicksort* encontra-se no APÊNDICE A e a Tabela 3.2 apresenta as variáveis utilizadas no código de ordenação.

Tabela 3.2 – Resumo das variáveis para *quicksort*

Variáveis	Descrição
$pivô$	Variável para marcar o pivô.
$esquerda, direita, i, j$	Variáveis para controle do código.
$comprimento_i$, $largura_i$, $area_i$	Dimensões do objeto i , sendo $i=1, \dots, n \in \mathbb{Z}$.

Fonte: Autoria própria.

O primeiro objeto sempre poderá ser encaixado na coordenada (0,0) da placa. Do segundo em diante, os objetos serão encaixados nas próximas colunas, se a posição da placa não estiver ocupada, se a soma das coordenadas atuais com dimensões dos objetos não exceder as dimensões da placa. Caso não satisfazer as condições, passa para linha seguinte e realiza a mesma verificação. Se o caso objeto não couber na placa atual, a seguinte placa poderá ser utilizada. A Tabela 3.3 apresenta as variáveis utilizadas no código.

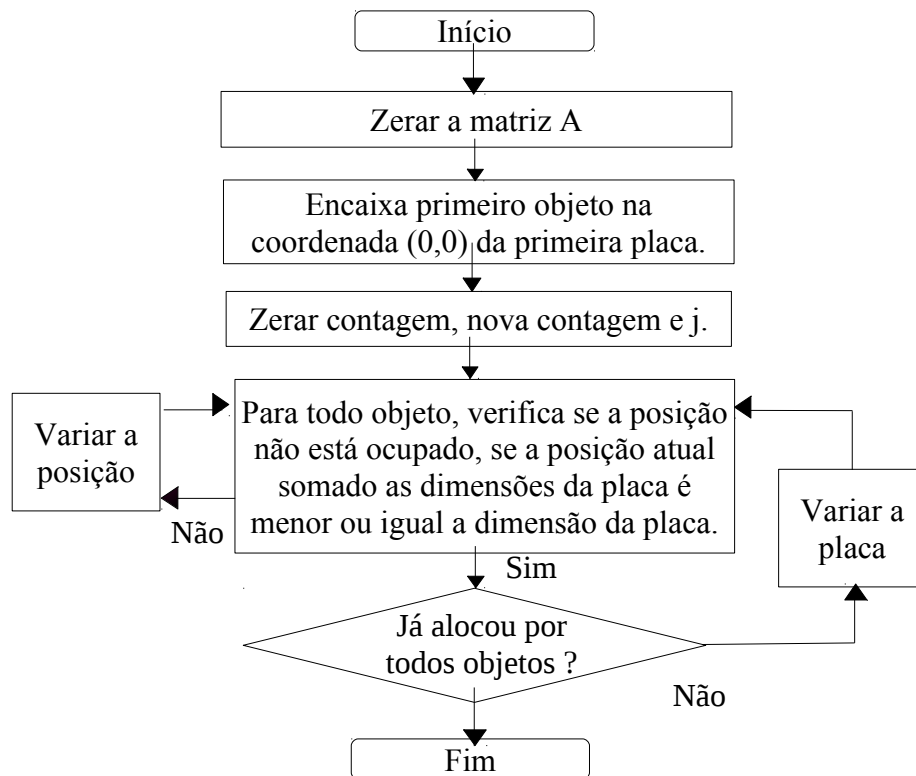
Tabela 3.3 – Resumo das variáveis para heurística para problema de corte bidimensional não guilhotinado

Variáveis	Descrição
n	Quantidade de Objetos.
m	Quantidade das placas disponíveis.
X , Y	Dimensões das placas.
$comprimento_i$, $largura_i$, $area_i$	Dimensões do objeto i , sendo $i=1, \dots, n \in \mathbb{Z}$.
A_{jrs}	A matriz com dimensões $[m] \times [X] \times [Y]$, armazena o número do objeto encaixado em cada posição ou zero, caso posição esteja vazia.

Fonte: Autoria própria.

A Figura 3.3 demonstra o fluxograma para implementação. O pseudocódigo encontra-se no APÊNDICE B

Figura 3.3 - Fluxograma da heurística *first fit* para problema de corte bidimensional não guilhotinado



Fonte: Autoria própria.

3.2.3 Heurística para problema de carteiro chinês não direcionado

Depois de ler os dados do arquivo no formato *txt*, gerar os padrões de corte e otimizar a placa com *software* de otimização GUROBI, para determinar o percurso para corte, primeiramente foi construída a matriz quadrada que armazena os dados da distância e indica se os nós são adjacentes. A lógica utilizada para construção é apresentada na fluxograma da Figura 3.4 e no pseudocódigo disponível no APÊNDICE C.

No problema, deseja-se minimizar a distância total percorrida, então o critério de seleção para melhor caminho foi a distância total percorrida. A função objetivo é soma das distâncias percorridas. A heurística utilizada para resolução do problema é de construtiva gulosa. Chamado construtiva por construir uma resposta e é chamado gulosa porque o método pega uma referência e usa como base para definir o restante. A Figura 3.5 apresenta o

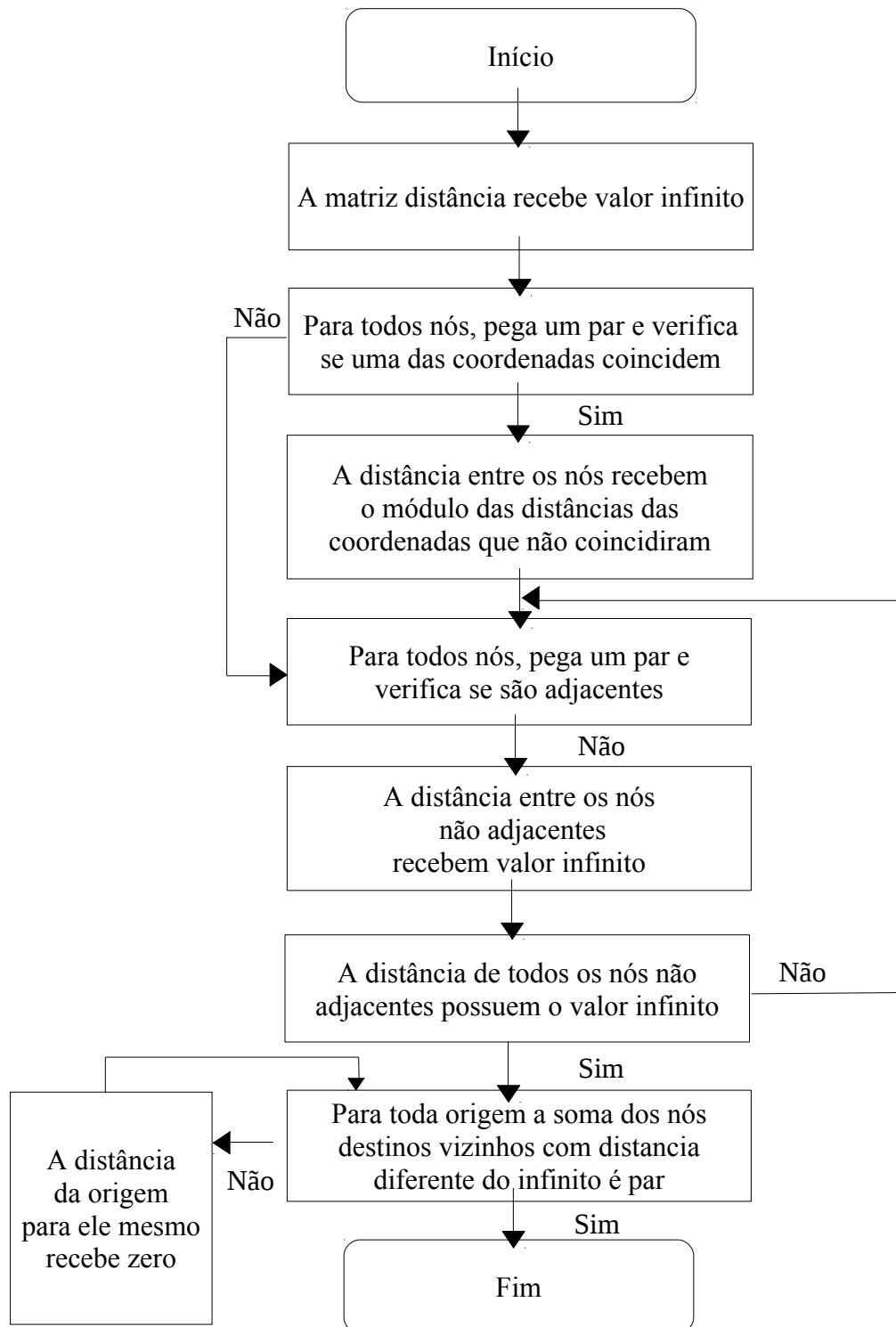
fluxograma com lógica da heurística e o pseudocódigo encontra-se disponível no APÊNDICE D. A Tabela 3.4 apresenta resumo das variáveis utilizadas na heurística.

Tabela 3.4 –Variáveis da heurística construtiva gulosa para problema de carteiro chinês não direcionado

Variáveis	Descrição
<i>node</i>	Quantidade dos nós.
x_i, y_i	Coordenadas do nó i , sendo $i=1,\dots,node \in \mathbb{Z}$.
d_{ij}	Matriz quadrada distância cuja dimensão é número de nós, onde os índices representam origem e destino respectivamente.
$vizinho_{ij}$	Matriz quadrada cuja dimensão é número de nós, onde os índices representam origem e destino respectivamente. A matriz recebe 1 se os nós forem adjacentes, 0 caso contrário.
$fluxo_{ij}$	Matriz quadrada cuja dimensão é número de nós. Onde os índices representam origem e destino respectivamente. A matriz para controle do caminho. Conta quantas vezes o pivô passou por aquele caminho.
<i>left, right, up, down, next</i>	Variáveis utilizados para controle.
<i>menor_f, menor_d</i>	Variáveis utilizados para controle de fluxo e distâncias percorridas.
<i>somavizinho</i>	Variável para guardar o valor de soma da matriz de vizinhos.
<i>pivô</i>	Variável que recebe o valor do nó pivô.
<i>FO</i>	Variável para armazenar o valor do FO.
<i>path</i>	Objeto que armazena os dados do nó (número do nó, coordenadas x e y) em ordem que o pivô passou, isto é, o caminho percorrido.
<i>star</i>	Objeto que armazena os dados do nó (número do nó, coordenadas x e y) em ordem que o pivô passou, isto é, o caminho percorrido. Difere do objeto <i>path</i> porque o <i>star</i> armazenar os dados no melhor caminho no momento.

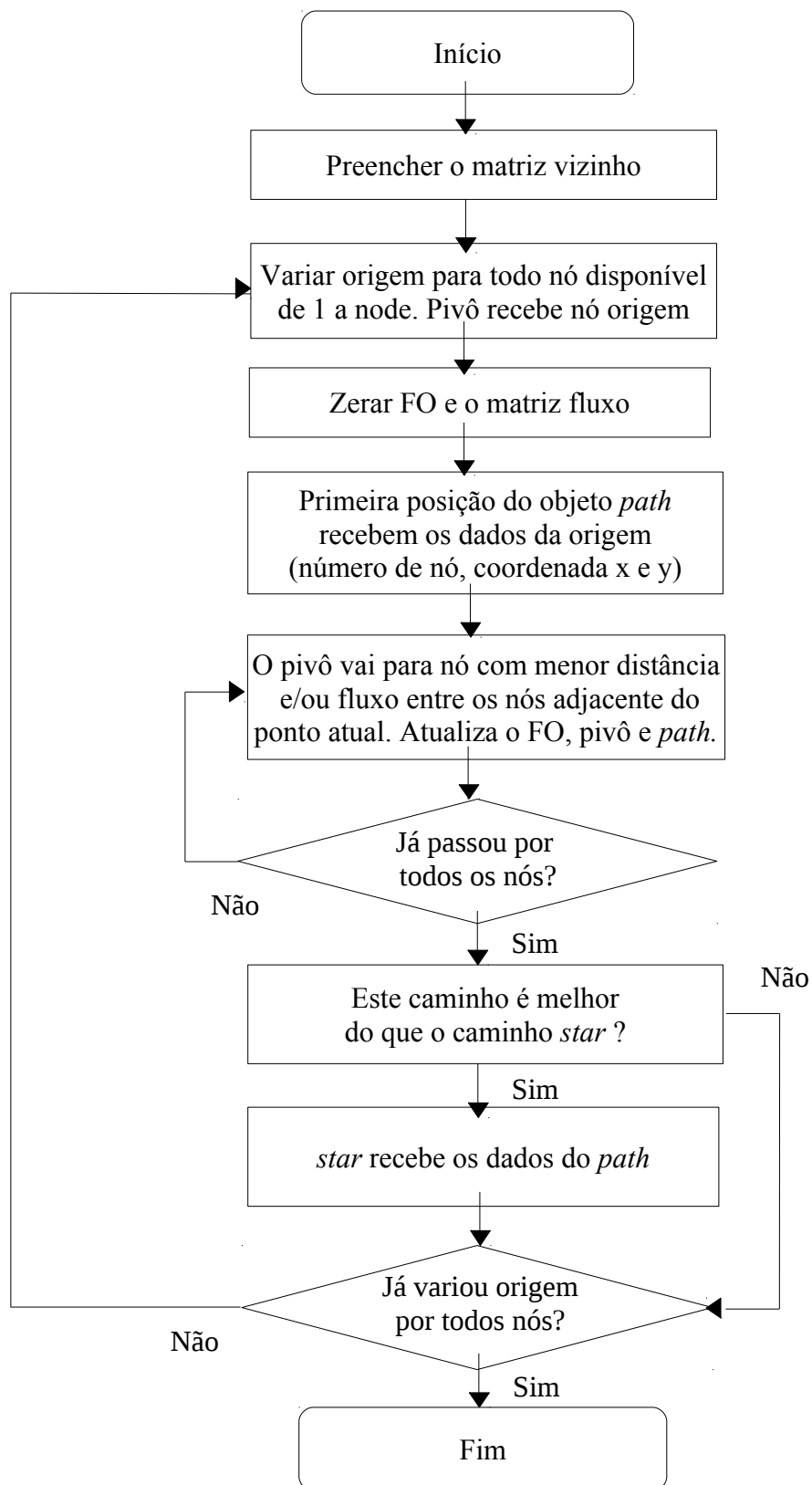
Fonte: Autoria própria.

Figura 3.4 - Fluxograma do algoritmo para preencher a matriz distância



Fonte: Autoria própria.

Figura 3.5 - Fluxograma da heurística construtiva gulosa para problema de carteiro chinês não direcionado



Fonte: Autoria própria.

3.2.4 Geração do *G-code* com o código em linguagem C++

Como presente trabalho trata apenas de retângulos, a geração do *G-code* para comandar a máquina com código em C++ possui baixa complexidade computacional. Possuindo as coordenadas dos nós e o caminho a ser seguido, gerado anteriormente pela heurística de carteiro chinês não direcionado, deve-se imprimir o arquivo externo do tipo *gcode*. A Tabela 3.5 apresenta os comandos utilizados para geração e o processo de geração foi da seguinte forma:

1. Inicializar o código;
2. Para garantir, colocar comando para levantar a caneta;
3. Movimentar para origem (dados do primeiro nó armazenado no objeto *star*);
4. Comando para baixar a caneta;
5. Movimentar para as respectivas coordenadas, até o último nó armazenado no objeto *star*;
6. Comando para levantar a caneta;
7. Voltar para origem;
8. Finalizar o código.

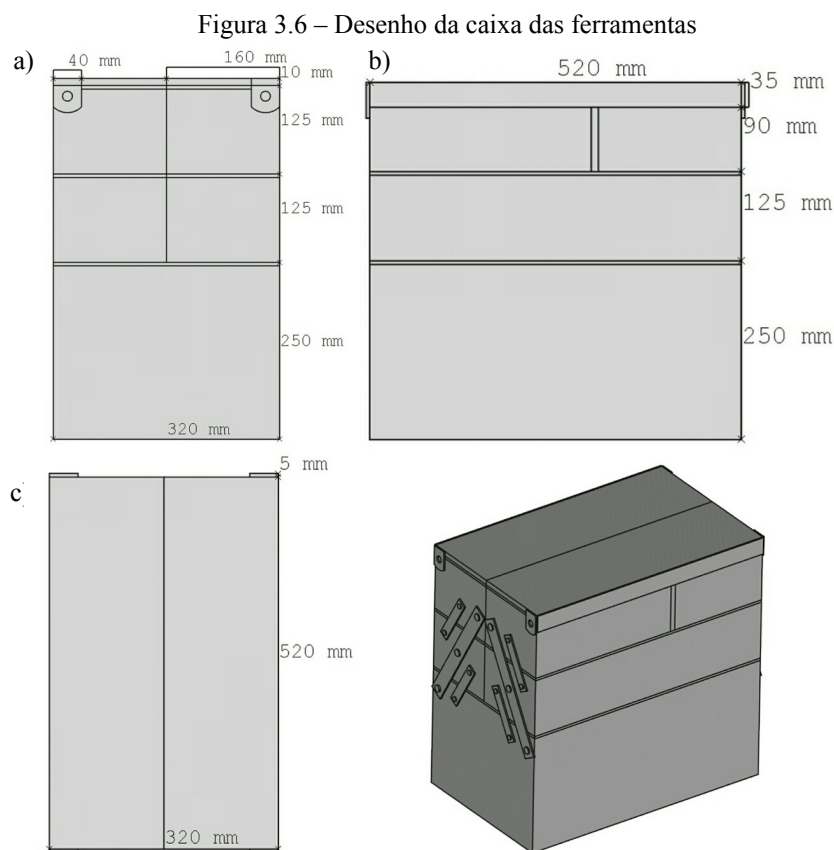
Tabela 3.5 – Comandos utilizados para geração do *G-code*

Código	Descrição
G21	Indica que as unidades estão em <i>milímetros</i> .
M280 Pn Sn	Comandos para movimentar o servo motor. Onde o <i>n</i> do P deve ser o pino onde o motor está ligado na placa <i>Ramps</i> e o <i>n</i> do S a angulação que deseja.
G0 Xn Yn	Mover à velocidade máxima para coordenada desejada, sendo <i>n</i> coordenada desejada em X e Y. É possível determinar a velocidade com o comando <i>Fn</i> onde <i>n</i> indica a velocidade em milímetros por minutos ou <i>inch</i> por minutos, dependendo da unidade definida anteriormente.
G1 Xn Yn	Mover para coordenada em linha reta, sendo <i>n</i> coordenada desejada em X e Y. É possível determinar a velocidade com o comando <i>Fn</i> onde <i>n</i> indica a velocidade em milímetros por minutos ou <i>inch</i> por minutos, dependendo da unidade definida anteriormente.
G90	Entende coordenada de modo absoluto.
G91	Entende coordenada relativa a posição atual.

Fonte: Adaptado do site de *Marlin* (<http://marlinfw.org>)

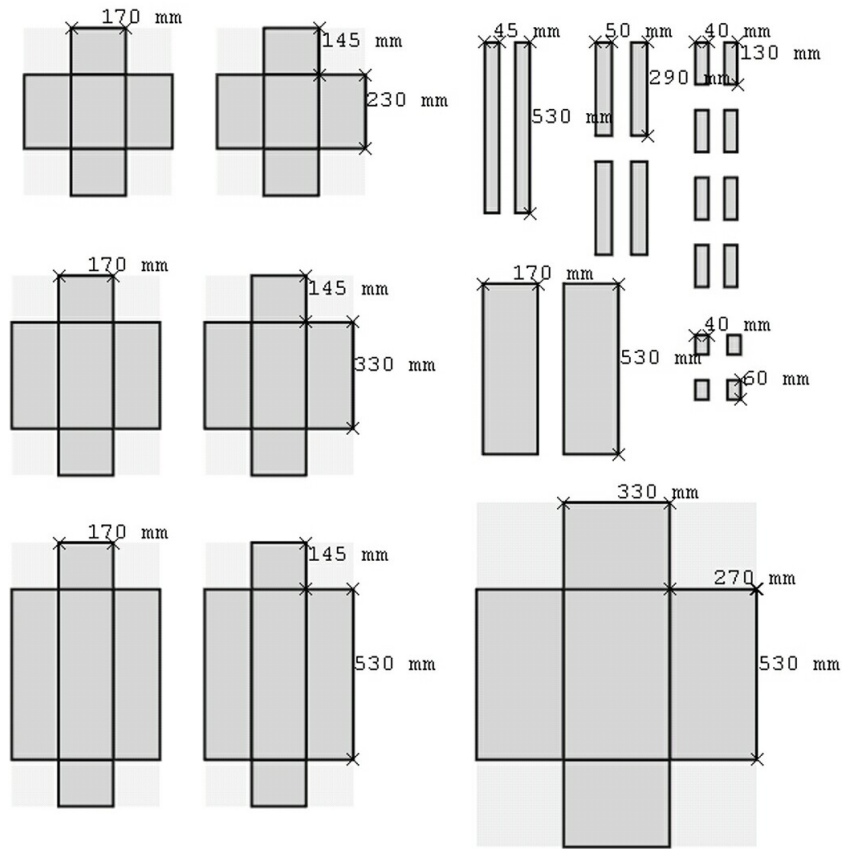
3.3 Os dados iniciais

Os dados de entrada do problema de corte bidimensional não guilhotinado são as dimensões da placa (1,20 x 3,00 metros) e as dimensões dos objetos. Os objetos para corte serão os componentes da caixa de ferramentas. Como na Figura 3.6, sendo ele desenhos da caixa de ferramentas com: (a) vista frontal; (b) vista lateral; (c) vista superior; (d) perspectiva isométrica. A Figura 3.7 mostra o desenho explodido dos componentes a serem cortados, já considerando as perdas nos processamentos. Uma versão ampliada do desenho está anexada no APÊNDICE G.



Fonte: Autoria própria.

Figura 3.7 – Desenho explodido da caixa das ferramentas



Fonte: Autoria própria.

Na Tabela 3.6 constam dimensões dos objetos a serem cortados, as unidades estão em centímetros. Sendo seus componentes 1: Base; 2: Gaveta 1 (D); 3: Gaveta 1 (E); 4: Gaveta 2 (D); 5: Gaveta 2 (E); 6: Gaveta 3 (D); 7: Gaveta 3 (E); 8 e 9: Tampas; 10 e 11: Orelhas frontais; 12 ~ 15: Fixadores maiores; 16 ~ 23: Fixadores menores; 24 ~ 27: Orelhas laterais. O pegador não entra na lista pois já é fornecido no tamanho certo e bastando realizar a dobra.

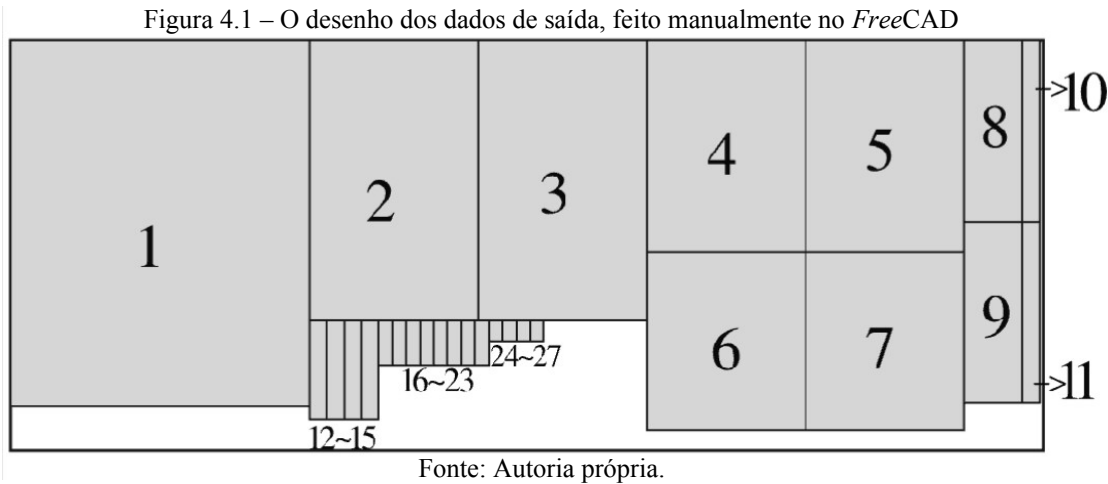
Tabela 3.6 – Os dados de entrada do problema de corte bidimensional

Nº (l x w)	Nº (l x w)	Nº (l x w)	Nº (l x w)	Nº (l x w)
1 (87 x 107)	7 (46 x 52)	13 (5 x 29)	19 (4 x 13)	25 (4 x 6)
2 (49 x 82)	8 (17 x 53)	14 (5 x 29)	20 (4 x 13)	26 (4 x 6)
3 (49 x 82)	9 (17 x 53)	15 (5 x 29)	21 (4 x 13)	27 (4 x 6)
4 (46 x 62)	10 (5 x 53)	16 (4 x 13)	22 (4 x 13)	
5 (46 x 62)	11 (5 x 53)	17 (4 x 13)	23 (4 x 13)	
6 (46 x 52)	12 (5 x 29)	18 (4 x 13)	24 (4 x 6)	

Fonte: Autoria própria.

4 RESULTADO E DISCUSSÃO

Os dados de saída do problema de corte bidimensional representam o *layout* de corte, apresentado como o número do objeto em cada posição ocupada pelo mesmo. A visualização do resultado fica difícil no arquivo *txt*, devido as dimensões da placa serem grandes e cada posição é preenchida com o número do objeto ou zero. O desenho foi feito no *software* modelador 3D paramétrico de código aberto *FreeCAD* para facilitar a visualização. A Figura 4.1 mostra o desenho CAD com base nos dados de saída do problema de corte.



Ainda no código de corte, a partir dos resultados, obtém-se os dados de coordenadas dos nós e os comprimentos das arestas. Os dados de entrada para problema de carteiro chinês não direcionado são as coordenadas dos objetos. Caso haja mais de um nó com mesmas coordenadas, os nós coincidentes devem ser excluídos, deixando apenas nós com coordenadas únicas. As unidades da Tabela 4.1 estão em centímetros.

Tabela 4.1 – Os dados de entrada do problema de carteiro chinês não direcionado

Nº	(x , y)	Nº	(x , y)	Nº	(x , y)	Nº	(x , y)	Nº	(x , y)
1	(0 , 0)	13	(0 , 277)	25	(106 , 299)	37	(95 , 111)	49	(95 , 135)
2	(107 , 0)	14	(62 , 277)	26	(111 , 87)	38	(82 , 115)	50	(82 , 139)
3	(0 , 87)	15	(114 , 185)	27	(82 , 92)	39	(95 , 115)	51	(95 , 139)
4	(107 , 87)	16	(114 , 231)	28	(111 , 92)	40	(82 , 119)	52	(88 , 139)
5	(82 , 87)	17	(114 , 277)	29	(82 , 97)	41	(95 , 119)	53	(82 , 143)
6	(0 , 136)	18	(53 , 277)	30	(111 , 97)	42	(82 , 123)	54	(88 , 143)
7	(82 , 136)	19	(0 , 294)	31	(82 , 102)	43	(95 , 123)	55	(82 , 147)

8 (0 , 185)	20 (53 , 294)	32 (111 , 102)	44 (82 , 127)	56 (88 , 147)
9 (82 , 185)	21 (106 , 277)	33 (82 , 107)	45 (95 , 127)	57 (82 , 151)
10 (62 , 185)	22 (106 , 294)	34 (111 , 107)	46 (82 , 131)	58 (88 , 151)
11 (0 , 231)	23 (0 , 299)	35 (95 , 107)	47 (95 , 131)	59 (82 , 155)
12 (62 , 231)	24 (53 , 299)	36 (82 , 111)	48 (82 , 135)	60 (88 , 155)

Fonte: Autoria própria.

Os dados de saída gerados no problema de carteiro chinês serão as coordenadas dos nós em ordem da rota, como mostra a Tabela 4.2. As unidades estão em centímetros.

Tabela 4.2 – Os dados de saída do problema de carteiro chinês não direcionado

Nº	Nó	(x , y)	Nº	Nó	(x , y)	Nº	Nó	(x , y)	Nº	Nó	(x , y)
1	32	(82 , 107)	36	8	(82 , 185)	71	10	(0 , 231)	106	27	(111 , 92)
2	35	(82 , 111)	37	9	(62 , 185)	72	7	(0 , 185)	107	29	(111 , 97)
3	37	(82 , 115)	38	11	(62 , 231)	73	5	(0 , 136)	108	31	(111 , 102)
4	39	(82 , 119)	39	13	(62 , 277)	74	2	(0 , 87)	109	33	(111 , 107)
5	41	(82 , 123)	40	17	(53 , 277)	75	4	(82 , 87)	110	34	(95 , 107)
6	43	(82 , 127)	41	12	(0 , 277)	76	26	(82 , 92)	111	36	(95 , 111)
7	45	(82 , 131)	42	18	(0 , 294)	77	28	(82 , 97)	112	35	(82 , 111)
8	47	(82 , 135)	43	19	(53 , 294)	78	30	(82 , 102)	113	37	(82 , 115)
9	6	(82 , 136)	44	21	(106 , 294)	79	32	(82 , 107)	114	38	(95 , 115)
10	49	(82 , 139)	45	24	(106 , 299)	80	34	(95 , 107)	115	40	(95 , 119)
11	52	(82 , 143)	46	23	(53 , 299)	81	36	(95 , 111)	116	39	(82 , 119)
12	54	(82 , 147)	47	19	(53 , 294)	82	38	(95 , 115)	117	41	(82 , 123)
13	56	(82 , 151)	48	17	(53 , 277)	83	40	(95 , 119)	118	42	(95 , 123)
14	58	(82 , 155)	49	12	(0 , 277)	84	42	(95 , 123)	119	44	(95 , 127)
15	59	(88 , 155)	50	10	(0 , 231)	85	44	(95 , 127)	120	43	(82 , 127)
16	57	(88 , 151)	51	11	(62 , 231)	86	46	(95 , 131)	121	45	(82 , 131)
17	55	(88 , 147)	52	15	(114 , 231)	87	48	(95 , 135)	122	46	(95 , 131)
18	53	(88 , 143)	53	16	(114 , 277)	88	50	(95 , 139)	123	48	(95 , 135)
19	51	(88 , 139)	54	20	(106 , 277)	89	51	(88 , 139)	124	47	(82 , 135)
20	49	(82 , 139)	55	21	(106 , 294)	90	53	(88 , 143)	125	6	(82 , 136)
21	6	(82 , 136)	56	24	(106 , 299)	91	52	(82 , 143)	126	5	(0 , 136)
22	5	(0 , 136)	57	21	(106 , 294)	92	54	(82 , 147)	127	2	(0 , 87)
23	7	(0 , 185)	58	20	(106 , 277)	93	55	(88 , 147)	128	4	(82 , 87)
24	10	(0 , 231)	59	13	(62 , 277)	94	57	(88 , 151)	129	3	(107 , 87)
25	12	(0 , 277)	60	11	(62 , 231)	95	56	(82 , 151)	130	25	(111 , 87)

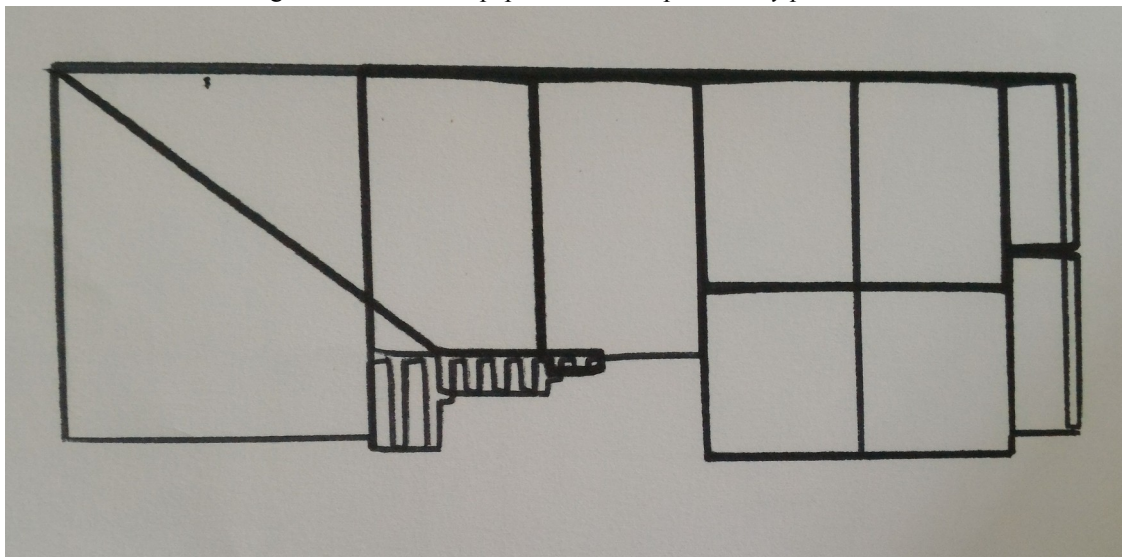
26	18	(0 , 294)	61	9	(62 , 185)	96	58	(82 , 155)	131	3	(107 , 87)
27	22	(0 , 299)	62	7	(0 , 185)	97	8	(82 , 185)	132	4	(82 , 87)
28	23	(53 , 299)	63	10	(0 , 231)	98	9	(62 , 185)	133	26	(82 , 92)
29	19	(53 , 294)	64	12	(0 , 277)	99	7	(0 , 185)	134	27	(111 , 92)
30	17	(53 , 277)	65	18	(0 , 294)	100	5	(0 , 136)	135	29	(111 , 97)
31	13	(62 , 277)	66	22	(0 , 299)	101	2	(0 , 87)	136	28	(82 , 97)
32	20	(106 , 277)	67	18	(0 , 294)	102	0	(0 , 0)	137	30	(82 , 102)
33	16	(114 , 277)	68	22	(0 , 299)	103	1	(107 , 0)	138	31	(111 , 102)
34	15	(114 , 231)	69	18	(0 , 294)	104	3	(107 , 87)			
35	14	(114 , 185)	70	12	(0 , 277)	105	25	(107 , 87)			

Fonte: Autoria própria.

Com os dados de saída gerados no problema de carteiro chinês, basta traduzir as informações das coordenadas para o *G-code*. O arquivo de saída do código de geração do *G-code* será um arquivo com os comandos para execução da simulação da corte. Os dados de entrada para o código, serão os dados de saída do problema de carteiro chinês (as coordenadas dos nós em ordem a ser seguida). Como *G-code* trabalha com unidades milímetro ou polegada, as unidades devem ser transformadas, caso necessário. Nesse caso, como a área útil do protótipo é de 260 x 240 milímetros, a unidade não sofrerá alteração, trabalhando com escala de 1:100.

Feito isso, copiar o arquivo *G-code* gerado para cartão de memória SD, conectar o cartão de memória na máquina e dar comando para executar. A Figura 4.2 apresenta a folha gerada na simulação do corte feito pela máquina *corexy plotter*. O *G-code* gerado encontra-se disponível no APÊNDICE E.

Figura 4.2 – Foto do papel desenhado pela *corexy plotter*



Fonte: Autoria própria.

Para validação dos códigos, foram utilizados os dados disponíveis na *OR-Library* (<http://www.brunel.ac.uk/~mastjjb/jeb/info.html>) do tópico *Two-dimensional cutting/packing* para o modelo matemáticos de corte bidimensional. A validação da heurística do problema de corte bidimensional, heurística de carteiro chinês e geração do *G-code* ocorreu de forma manual.

Com os dados de 27 objetos e as dimensões da placa 120 x 300, não foi possível executar o modelo matemático de corte bidimensional, apresentando falha de segmentação durante a execução. No estado atual não é possível resolver os dados da aplicação. A matriz A precisa de uma quantidade de posições de memória muito grande e junto com as demais variáveis acarreta um erro em tempo de execução.

A solução para este caso seria redução da escala dos dados como um todo. Porém como matriz A é construída com posições de memória, dependendo dos dados, não há diferença trabalhar com escala, já que não existe meia posição de memória. Com isso a utilização do modelo matemático se torna inviável, sendo mais vantajosa a resolução com heurística. Então foi implementada heurística *first fit* para resolução do caso. O código se mostrou válido e eficiente para resolução do problema.

Foram realizados testes em um computador com 4GB de memória RAM e em outro com 48GB de memória RAM, todos com sistema operacional Ubuntu 16.04 LTS e 64-bits. As dimensões dos objetos foram fixados em 1 x 1, variando apenas as dimensões da placa e

quantidade dos objetos. Já que as dimensões dos objetos possui relação direta com o tempo de execução e não com quantidade de espaço de memória utilizado.

Como esperado, computador com maior memória RAM suportou maior número de variáveis. Porém não foi possível obter as relações entre a quantidade de variáveis máximo e capacidade de memória. A versão com heurística permite trabalhar com os números de variáveis tende a infinito, comparado com outros. Não foi realizado análise profundo sobre os dados pois este não é o foco do presente trabalho.

4.1 Resultado geração de layout da corte utilizando ferramentas de pesquisa operacional

A geração de layout ocorreu com utilização de duas ferramentas de pesquisa operacional. Com o método exato (programação linear), resolução com pacote de otimização GUROBI e com o método heurística, resolvido com *first fit*.

A heurística do problema de corte bidimensional, apesar de não permitir rotação das peças atualmente, é aplicável em casos de múltiplas placas. Pode ser trabalhado um maior número de objetos do que no modelo matemático e não corre o risco de se gastar horas na resolução do problema.

O código do problema de corte com modelo matemático, apesar de permitir a rotação dos objetos em 90°, é bastante restritivo por conta da particularidade do código, que resolve o caso de corte não guilhotinado de objetos bidimensionais retangulares, com uma placa e corre o risco de demandar horas para resolução do problema, o que é imprevisível.

Ainda sobre o código de problema de corte com modelo matemático, em alguns casos, ao analisar o resultado obtido, fica evidente que, caso se modifique as posições de algumas peças, de uma forma que as posições vagas fiquem agrupadas, a área que sobra ficaria maior, possibilitando o uso da matéria-prima remanescente para possíveis demandas futuras. Contudo o código atual não possui uma função para agrupar os objetos na placa exatamente porque o código atual utiliza o modelo matemático para resolução do problema de corte. Se sofrer esta mudança, não seria mais programação linear, e sim, uma heurística, ou misto de programação linear com heurística.

Para utilização do código do problema de corte com modelo matemático é necessário certo bom senso do usuário, por exemplo, para executar programa com a área total dos objetos

sem ultrapassar a área da placa, pois o código atual é válido apenas para uma placa. Se deixar a área total dos objetos ultrapassarem a área da placa, o programa encaixará o máximo de objetos que couber na placa, mas não conseguirá encaixar todos os objetos.

Ao analisar o arquivo de saída do corte do tipo *txt* contendo os dados da saída (*layout* do corte), nota-se a dificuldade em visualização do *layout* da corte conforme aumentam as dimensões da placa. Este problema será maior no corte 3D. Pois é difícil representar três dimensões em espaços bidimensionais. A dificuldade em visualizar a posição do corte inviabiliza a utilização do código. Uma solução para isso seria gerar a geometria 3D automaticamente com *software* de CAD e uma tabela auxiliar com os dados do objeto e a matéria-prima.

Como o presente trabalho aborda os problemas NP-*hard*, apesar de garantir o resultado ótimo, o modelo matemático pode demorar horas para achar a solução dependendo dos dados. Muitas vezes o usuário precisará da sequência de produção e/ou *layout* do corte imediatamente. Na indústria, o tempo de espera para determinar a otimização da placa com programação linear é perda de tempo. Já que existe heurística que determina resultado ótimo ou próximo do ótimo, utilizando muito menos tempo.

4.2 Resultado traçar rota para corte com problema de carteiro chinês

Houve tentativa de utilização de um modelo matemático para o problema de carteiro chinês para otimização do percurso para corte, porém foi constatada uma grande dificuldade em representar o problema devido à sua elevada complexidade. Levando em conta esse fator, foi implementado heurística *first fit* para problema de carteiro chinês.

Foram constatados algumas carências quanto a heurística baseada no problema do carteiro chinês. Quando a quantidade dos objetos é baixa, o código consegue achar a rota otimizada, ou bem próxima da ótima. Conforme cresce a quantidade dos objetos, as rotas deixam de ser ótimas devido à complexidade do problema e a simplicidade do código. A solução neste caso seria tratar o resultado do *first fit* como sendo solução inicial e implementar uma função para realizar as mudanças na rota para chegar o mais perto possível da solução ótima, utilizando meta-heurística de método de pesquisas em vizinhança variável ou algoritmo genético.

4.3 Resultado geração do *G-code* automático com linguagem C++

Pelo fato de presente trabalho tratar apenas de objetos retangulares, geração do *G-code* com a linguagem C++ ocorreu bem. Comparado a estrutura de impressora ou máquina de corte a plasma, o *G-code* de máquina de corte com a lâmina será mais simples, por não envolver a temperatura do cabeçote e a velocidade não interferir tanto no resultado final.

Se não tivesse este código, o grau de automação seria menor, pois teria que ser utilizado *software* capaz de gerar *G-code*. Desenhar o *layout* de corte, vetorizar o desenho para gerar o código e isso toma o tempo do operário. Com código de heurística de problema de carteiro chinês e geração automática com linguagem C++, reduz o processo manual significativamente.

4.4 Resultado execução da simulação de corte no protótipo com funcionamento conforme os conceitos da indústria 4.0

Por fim, quanto a objetivo final de simulação de corte no protótipo da máquina de corte bidimensional não guilhotinado, apesar das carências no código de heurística de problema de carteiro chinês, simulação de corte no protótipo ocorreu bem. A máquina em si, atende a necessidade de simular a corte. Porém é necessário aprimoramento nos códigos para obter melhores resultados.

No momento ainda não foi possível automatizar todo processo. A partir do momento que o *G-code* é gerado, o processo de transferência dos arquivos para máquina (via cartão SD) e comando para execução máquina são realizadas manualmente. Podendo ser utilizado módulo *ethernet*, módulo Wi-Fi ou por meio de protocolo de comunicação TCP/IP para transferência dos dados, por exemplo.

5 CONCLUSÃO

O presente trabalho trata da simulação de corte no protótipo de corte bidimensional não guilhotinado na estrutura *corexy plotter* controlada por microcontrolador Arduino com a placa *Ramps* e *firmware Marlin*. Onde o *layout* de corte é otimizado por heurística *first fit* para problema de corte bidimensional e percurso para corte determinado por heurística construtiva gulosa do problema de carteiro chinês não direcionado e geração automática de *G-code* com um *software* desenvolvido em C++.

Com os dados para aplicação não foi possível executar o modelo matemático de corte bidimensional, apresentando falha de segmentação. A matriz precisa de uma quantidade de posições de memória muito grande e junto com as demais variáveis acarreta um erro em tempo de execução. Ou seja, não foi possível resolver com o modelo matemático, neste caso. Com isto, foi implementado heurística *first fit* para problema de corte bidimensional com múltiplas placas.

O código se mostrou válido para aplicação no problema real em casos de ambiente de produção pequenos com poucos objetos. Todavia a otimização com modelo matemático acaba sendo muito restritiva pois o problema precisa ser representável em equações para ser aplicável. Enquanto as heurísticas são mais flexíveis com as condições do problema.

Seria válida a implementação do modelo matemático para conferir os resultados das heurísticas e obter comparações quanto a tempo e variabilidade dos resultados, para melhor análise das heurísticas implementadas. Porém a implementação do modelo matemático para utilização na indústria é inviável por conta do tempo de execução e falta de flexibilidade na implementação.

É viável investir nos estudos dos temas abordados pois a tendência é de automatização do processo de manufatura. Diminuindo *leadtime*, a intervenção humana no processo, consequentemente reduzindo o erro humano, o impacto da falta de trabalhadores, permite otimização do tempo e das máquinas e assim aumentando a qualidade dos produtos e a receita da empresa.

Apesar da carência no código de heurística de problema de carteiro chinês, o protótipo de máquina de corte bidimensional não guilhotinado funcionou bem e a máquina atende os

requisitos. Atualmente, o processo de corte é semiautomático, onde processo de transferência dos arquivos para máquina ainda ocorre manualmente, com auxílio de cartão SD.

Algumas sugestões para possíveis trabalhos futuros relacionados a tema abordado no presente trabalho:

- a) No problema de corte dimensional, tornar a heurística uma meta-heurística para buscar melhores resultados e acrescentar restrições de agrupamento das sobras da placa e armazenar os pedaços que sobraram do corte para possíveis demandas futuras. Também tornar problema múltiplas matérias-primas com tamanhos variáveis e acrescentar uma terceira dimensão. Ou seja, implementação da heurística de corte tridimensional com múltiplas matérias-primas variáveis com restrição de agrupamento das sobras para possíveis demandas futuras, aumentando a versatilidade do programa.
- b) Utilizar um software de CAD ou *software* similar para gerar a figura 3D com os *layout* de corte independente das dimensões da matéria-prima, para fácil entendimento do resultado do problema.
- c) Sobre o protótipo de corte bidimensional, ampliar o protótipo para simular um ambiente *job shop*. Isto é, aumentar as máquinas como de dobras, corte guilhotina e solda, máquina que possa identificar local de armazenamento e pegar a matéria-prima certa através dos sensores ou rádio frequência (RFID). Tornando assim, protótipo de uma fábrica como todo.
- d) Para o protótipo, criar um aplicativo onde usuário entra com as descrições das encomendas (materiais, cores, dimensões e outras informações), e para quando precisa. O sistema armazena os dados na nuvem, com *cloud computing*. O sistema da fábrica recebe e consegue acessar informações com auxílio de um módulo Wi-Fi, *ethernet* ou com protocolo de comunicação TCP/IP, o código resolve o problema de *job shop* e de corte e repassa informações para as máquinas, para permitir que o usuário realize a encomenda mesmo estando longe fisicamente, praticamente sem intervenção humana e agilizando ainda mais o processo de toda cadeia.
- e) Acrescentar roteamento de entrega dos produtos finais no código principal. Com o endereço, demanda e data prometida e necessidade especiais do cliente e/ou produto, determinar a rota e a frota a ser utilizada para otimizar tempo e percurso.

REFERÊNCIAS

- AIRES, R. W. A., MOREIRA, F. K., FREIRE, P. S., **Indústria 4.0: Competências requeridas aos profissionais da quarta revolução industrial**. VII congresso internacional de conhecimento e inovação. Foz do Iguaçu, 2017. Disponível em: <<http://proceeding.ciki.ufsc.br/index.php/ciki/article/view/314/153>>. Acesso em: 22 Nov. 2017.
- ARENALES, M. N., ARMENTANO, V., MORABITO, R., **Pesquisa Operacional**. Rio de Janeiro: Campus. 2006.
- BATALHA, M. O. et al., **Introdução à engenharia de produção**. Rio de Janeiro: Elsevier, 2008.
- BOGOTOBO. **Sockets – Server & Client – 2017 C++**. Disponível em: <http://www.bogotobogo.com/cplusplus/sockets_server_client.php>. Acesso em: 4 Jul. 2017.
- CANONGIA, C., JUNIOR, R. M., **Segurança cibernética: o desafio da nova sociedade da informação**. Parcerias Estratégicas, v. 14, n. 29, p. 21-46. Brasília, 2009. Disponível em: <http://seer.cgee.org.br/index.php/parcerias_estrategicas/article/viewFile/349/342>. Acesso em: 22 Nov. 2017.
- CAVALCANTE, M. A., TAVOLARO, C. R. C., MOLISANI, E., **Física com Arduino para iniciantes**. Revista Brasileira de Ensino de Física, v. 33, n. 4, 2011. Disponível em: <<http://www.scielo.br/pdf/rbef/v33n4/18.pdf>>. Acesso em: 6 Jul. 2017.
- CLARO, F. D., **O avanço tecnológico no mundo econômico**. Vitrine da Conjuntura, Curitiba, v.2, n.8, outubro 2009. Disponível em: <<http://img.fae.edu/galeria/getImage/1/731424487250267.pdf>>. Acesso em: 26 Set. 2017.
- CORRÊA, H. L., CORRÊA C. A., **Administração de produção e operações: manufatura e serviços: uma abordagem estratégica**. 3.ed. São Paulo: Atlas, 2012.
- CORRÊA, H. L., GIANESI, I. G. N., CAON, M., **Planejamento, programação e controle da produção: MRP II/ERP Conceitos, Uso e Implantação Base para SAP, Oracle Applications e outros softwares Integrados de Gestão**. 5.ed. São Paulo: Atlas, 2012.
- COSTA, C., **Indústria 4.0: o futuro da indústria nacional**. POSGERE, v. 1, n. 4, p.5-14 set 2017. Número especial automação. Disponível em: <<http://seer.spo.ifsp.edu.br/index.php/posgere/article/view/82/pdf>>. Acesso em: 25 Out. 2017.
- FERNANDES, F. C. F., FILHO, M. G., **Planejamento e controle da produção: Dos Fundamentos ao Essencial**. São Paulo: Atlas, 2010.
- FLESZAR, K., HINDI, K. S., **New heuristics for one-dimensional bin-packing**. Department of Systems Engineering, Brunel University, Uxbridge, Middlesex UB8 3PH, UK, 2000.

- FONCECA, E. G. P., VEGA, A. S., **Tutorial sobre introdução a projetos utilizando o kit de desenvolvimento arduino**. XXXIX COBENGE 03 a 06 de OUT. Blumenau. 2011. Disponível em: <http://www.telecom.uff.br/pet/petws/downloads/artigos/cobenge_2011/COBENGE_art1677_2_ARDUINO_2011_08_30.pdf>. Acesso em: 4 Jul. 2017.
- GROOVER, M. P., **Automação industrial e sistema de manufatura**. Tradução de Jorge Ritter, Luciana do Amaral Teixeira, Marcos Vieira. Revisão técnica de José Hamilton Chaves Gorgulho Júnior. 3.ed. São Paulo: Pearson, 2011.
- Henn na Hotel. Disponível em: <<http://www.h-n-h.jp/>> Acesso em: 22 Jul. 2017.
- HILLIER, F. S. LIBERMAN, G. J., **Introdução à pesquisa operacional**. Tradução de Ariovaldo Griesi. 8ªed. Porto Alegre: AMGH, 2010.
- JUNQUEIRA, L., MORABITO, R., YAMASHITA. D. S., **Modelos de otimização para problemas de carregamento de contêineres com considerações de estabilidade e de empilhamento**. Pesquisa Operacional, v. 30, n. 1, p. 73-98, Jan./Abr. 2010. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0101-74382010000100005>. Acesso em: 26 Ago. 2017.
- KIRNER, C., SISCOOTTO, R., **Realidade virtual e amentada: conceitos, projeto e aplicações**. Pré-Simpósio IX Symposium on virtual and augmented reality, Petrópolis, 2007. Disponível em: <http://www.marcelohsantos.com.br/aulas/downloads/2Semestre_2014/novasmidias/Jogos_PE_Novas_Midias_e_Tecnologias_para_Jogos_Aula4_01.pdf>. Acesso em: 22 Nov. 2017.
- KONOWALENKO, F., COSTA, D. M. B., NUNES, L. F., **Problema do carteiro chinês não-orientado e misto para a otimização de rotas na cidade de Irati/PR**. 2012. 108 f. Tese (Mestrado em Ciências) - Universidade Federal do Paraná, Curitiba. 2012.
- KOVÁCS, I., **Os avanços tecnológicos e o futuro do trabalho: debates recentes**. XVI ENCONTRO NACIONAL DE SIOT, p. 10-23. Lisboa, 2016. Disponível em: <https://www.researchgate.net/publication/304040159_Os_avancos_tecnologicos_e_o_futuro_do_trabalho_debate_recents?enrichId=rgreq-9aba37972b5327b2e730d7e20f182854-XXX&enrichSource=Y292ZXJQYWdlOzMwNDA0MDE1OTtBUzozNzQwNzM1Mjg5OTU4NDBAMTQ2NjE5NzQ4NzE1Ng%3D%3D&el=1_x_3&_esc=publicationCoverPdf>. Acesso em: 20 Nov. 2017.
- LEMONS, A., **A comunicação das coisas. Internet das coisas e teoria ator-rede: Etiquetas de radiofrequência em uniformes escolares na Bahia**. Bahia, 2012. Disponível em: <https://s3.amazonaws.com/academia.edu.documents/36911184/Andre_Lemos.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1511366821&Signature=ZRBdbgDkCcbequ%2BSp56YK7XEhIM%3D&response-content-disposition=inline%3B%20filename%3DAndre_Lemos.pdf> Acesso em: 22 Nov. 2017.
- LIKER, J. K., **O modelo Toyota: 14 princípios de gestão do maior fabricante do mundo**. Tradução de Lene Belon Ribeiro. Porto Alegre: Bookman, 2005.

- LUSTOSA, L. J. et al., **Planejamento e controle da produção**. Rio de Janeiro: Elsevier, 2008.
- MARCHESAN, M., **Sistema de monitoramento residencial utilizando a plataforma arduino**. Santa Maria. 2012. Disponível em: <http://www.redes.ufsm.br/docs/tccs/Marcelo_Marchesan.pdf>. Acesso em: 6 Jul. 2017.
- Marlin*, Disponível em: <<http://marlinfw.org/>> Acesso em: 22 Ago. 2017.
- MCROBERTS, M., **Arduino básico**. Tradução de Rafael Zanolli. São Paulo: Novatec, 2011.
- MERLO, Á. R. C., LAPIS, N.L., **A saúde e os processos de trabalho no capitalismo: reflexões na interface da psicodinâmica do trabalho e da sociologia do trabalho**. Psicologia & Sociedade, Porto Alegre, v. 19, n. 1, p.61-68, jan./abr. 2007. Disponível em: <<http://www.redalyc.org/html/3093/309326394009/>>. Acesso em: 22 Nov.2017.
- Moely Robotic. Disponível em: <<http://www.moley.com/>> Acesso em: 22 Jul. 2017.
- MORABITO, R., PUREZA, V., **Geração de padrões de cortes bidimensionais guilhotinados restritos via programação dinâmica e busca em grafo-e/ou**. Produção, v. 17, n. 1, p. 033-051, Jan./Abr. 2007. Disponível em: <<http://www.scielo.br/pdf/prod/v17n1/02.pdf>>. Acesso em: 21 Jul. 2017.
- MOREIRA, D. A., **Administração da produção e operações**. São Paulo: Pioneira, 1993.
- OR-Library disponível em:<<http://www.brunel.ac.uk/~mastjjb/jeb/info.html>>.
- Palro. Disponível em: <<https://palro.jp/preventive-care/nursing-home.html>> Acesso em: 22 Jul. 2017.
- PERLINGEIRO, C. A. G., **Engenharia de processos: abálise, simulação, otimização e síntese de processos químicos**. São Paulo: Blucher, 2005.
- PRUDENTE, F., **Automação industrial PLC: teoria e aplicações: custo básico**. Rio de Janeiro: LTC, 2011.
- REIS, J. V., CUNHA, C. B., **Meta-heurísticas baseadas em busca de vizinhança variável aplicados a problemas de operação de transportes**. 2013. 221f. Tese (Doutorado em engenharia de transporte) – Universidade de São Paulo. São Paulo, 2013.
- RODRIGUES, M. V., **Entendendo, aprendendo e desenvolvendo sistemas de produção Lean Manufactring**. 2ªed. Elsevier: 2016.
- RUBMANN, M. et al., **Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries**. 2015. Disponível em: <https://www.bcgperspectives.com/content/articles/engineered_products_project_business_industry_40_future_productivity_growth_manufacturing_industries/>. Acesso em: 22 Jul.2017.
- RUSSWURM. S., **Industrie 4.0 – from vision to reality**. Background Information, 2014. Disponível em: <<https://www.siemens.com/press/pool/de/events/2014/industry/2014-04-hannovermesse/background-indutrie40-e.pdf>>. Acesso em: 23 Jul. 2017.

- SCHRODER, R., NUNES, F. L., VIERO, C. F., MENEZES, F. M., **Análise da implantação de um processo automatizado em uma empresa calçadista: um estudo de caso a luz do sistema hyundai de produção e a indústria 4.0.** Espacios. v. 36, n. 18, p.18, 2015. Disponível em: <<http://www.revistaespacios.com/a15v36n18/15361819.html>>. Acesso em: 23 Jul. 2017.
- SILVA, E. M.C., **Modelos e métodos de otimização para problemas de corte e empacotamento a duas dimensões.** 2011. 299 f. Tese (Doutorado em Engenharia Industrial e de Sistemas) – Universidade do Minho, Escola de Engenharia, Braga, 2011.
- SILVA, R. M., FILHO, D. J. S., MIYAGI, P. E., **Modelagem de sistema de controle da indústria 4.0 baseada em holon, agente, rede de petri e arquitetura orientada a serviços.** São Paulo, 2015. Disponível em: <https://www.researchgate.net/publication/282292095_MODELAGEM_DE_SISTEMA_DE_CONTROLE_DA_INDUSTRIA_40_BASEADA_EM_HOLON_AGENTE_REDE_DE_PETRI_E_ARQUITETURA_ORIENTADA_A_SERVICOS>. Acesso em: 22 Jul. 2017.
- SILVEIRA, C. B., LOPES, G. C., **O que é indústria 4.0 e como ela vai impactar o mundo.** Citisystems. 2017. Disponível em: <<https://www.citisystems.com.br/industria-4-0/>>. Acesso em: 21 Nov. 2017.
- SLACK, N., CHAMBERS, S., JOHNSTON, R., **Administração da produção.** Tradução de Henrique Luiz Corrêa. 3.ed. São Paulo: Atlas, 2009.
- SOUZA, A. F., ULBRICH, C. B. L., **Engenharia integrada por computador e sistemas cad/cam/cnc – princípio e aplicações.** São Paulo: Artliber editora, 2009.
- SOUZA, F. R. C., MOREIRA, L. O., MACHADO, J.C., **Computação em nuvem: conceitos, tecnologias, aplicações e desafios.** Ceará, 2010. Disponível em: <https://www.researchgate.net/profile/Javam_Machado/publication/237644729_Computacao_em_Nuvem_Conceitos_Tecnologias_Aplicacoes_e_Desafios/links/56044f4308aea25fce3121f3.pdf>. Acesso em: 22 Nov. 2017.
- SUGAYAMA, R., NEGRELLI, E., **Connected vehicle on the way of Industry 4.0.** Disponível em: <https://pdfdocumento.com/connected-vehicle-on-the-way-of-industry-4_59cd91d51723dd3b102b723b.html>. Acesso em: 22 Jul. 2017.
- TechTudo, Disponível em: < <http://www.techtudo.com.br/dicas-e-tutoriais/noticia/2014/04/firmware-o-que-e-e-como-atualizar.html>> Acesso em: 22 Ago. 2017.
- TEODORO, J. N., ALMEIDA, N. D., **Indústria 4.0: visões de futuro.** Ituiutaba, 2015.
- Toyoriki, Disponível em: <<http://www.toyoriki.co.jp/>> Acesso em: 22 Jul. 2017.
- TUBINO, D. F., **Planejamento e Controle da Produção: teoria e prática.** 2.ed. São Paulo: Atlas, 2009.
- União Geek. Disponível em: <<https://www.uniaoageek.com.br/arquitetura-de-redes-tcpip/>>. Acesso em: 24 Jul. 2017.

APÊNDICE A – Pseudocódigo algoritmo de ordenação *quicksort*

Início:

```
1.    $i \leftarrow$  esquerda;
2.    $j \leftarrow$  direita;
3.    $pivô \leftarrow ((esquerda + direita) / 2)$ ;
4.    $pivô \leftarrow área_{pivô}$  ;
5.   faça enquanto (  $i \leq j$  )
6.       enquanto ( (  $área_i > pivô$  ) e (  $i < direita$  ) ) faça
7.            $i++$  ;
8.       fim enquanto;
9.       enquanto ( (  $área_j < pivô$  ) e (  $j > direita$  ) ) faça
10.           $j--$  ;
11.      fim enquanto;
12.      se (  $i = j$  ) então
13.          troca (  $área_i$  ,  $área_j$  );
14.          troca (  $comprimento_i$  ,  $comprimento_j$  );
15.          troca (  $largura_i$  ,  $largura_j$  );
16.           $i++$  ;
17.           $j--$  ;
18.      fim se;
19.  fim enquanto;
20.  se (  $esquerda < j$  ) então
21.      chamar função quicksort ( esquerda,  $j$  );
22.  fim se;
23.  se (  $i < direita$  ) então
24.      chamar função quicksort (  $i$  , direita );
25.  fim se;
Fim.
```

**APÊNDICE B – Pseudocódigo heurística *first fit* para problema de
corte bidimensional não guilhotinado**

Início:

```

1.   para todo  $j=1,\dots,m_{corte}$ 
2.       para todo  $r=1,\dots,L$ 
3.           para todo  $s=1,\dots,W$ 
4.                $A_{jrs} \leftarrow 0$ ;
5.           fim para todo;
6.       fim para todo;
7.   fim para todo;
8.   para todo  $r=1,\dots,comprimento_1$ 
9.       para todo  $s=1,\dots,largura_1$ 
10.             $A_{0rs} \leftarrow 1$ ;
11.        fim para todo;
12.    fim para todo;
13.    contagem, nova_contagem,  $j \leftarrow 0$ ;
14.    para todo  $i=2,\dots,n_{corte}$ 
15.        contagem, nova_contagem  $\leftarrow 0$ ;
16.        para todo  $r=1,\dots,L$ 
17.            para todo  $s=1,\dots,W$ 
18.                se ( (  $A_{jrs} = 0$  ) e ( contagem = nova_contagem ) e (  $r +$ 
19.                    comprimento $i$   $\leq L$  ) e (  $s + largura_i \leq W$  ) ) então
20.                        para todo  $l=r,\dots,r+comprimento_i$ 
21.                            para todo  $w=s,\dots,s+largura_i$ 
22.                                 $A_{jrs} \leftarrow i$  ;
23.                                contagem  $\leftarrow$  nova_contagem  $++$  ;
24.                            fim para todo;
25.                        fim para todo;
26.                    fim se;
27.                fim para todo;
28.            se ( contagem = nova_contagem ) então
29.                 $j ++$  ;
30.            Retornar para linha (16);
31.        fim se;
32.    fim para todo;
Fim.
```

APÊNDICE C – Pseudocódigo geração da matriz distância

Início:

```

1.  para todo  $i=1, \dots, node$ 
2.      para todo  $j=1, \dots, node$ 
3.          faça:  $d_{ij} \leftarrow \infty$  ;
4.      fim para todo;
5.  fim para todo;
6.  para todo  $i=1, \dots, node$ 
7.      para todo  $j=1, \dots, node$ 
8.          se (  $i \neq j$  ) então
9.              se (  $x_i = x_j$  ) então
10.                   $d_{ij} \leftarrow |y_i - y_j|$  ;
11.              fim se;
12.              caso contrário se (  $y_i = y_j$  ) então
13.                   $d_{ij} \leftarrow |x_i - x_j|$  ;
14.              fim se;
21.              se ( (  $x_i = x_j$  ) e (  $d_{ij} \neq \infty$  ) ) então
22.                  se (  $y_i < y_j$  ) então
23.                       $left \leftarrow y_i$  ;
24.                       $right \leftarrow y_j$  ;
25.                  fim se;
26.                  caso contrário
27.                       $left \leftarrow y_j$  ;
28.                       $right \leftarrow y_i$  ;
29.                  fim se;
30.                  distância  $\leftarrow |left - right|$  ;
31.                  ponto  $\leftarrow j$  ;
32.              fim se;
33.              para todo  $k=1, \dots, node$ 
34.                  se ( (  $k \neq i$  ) e (  $k \neq j$  ) e (  $x_i = x_k$  ) e (
                     $d_{ij} \neq \infty$  ) ) então
35.                      se ( (  $y_k > left$  ) e (  $y_k < right$  ) ) então
36.                           $left \leftarrow y_j$  ;
37.                           $right \leftarrow y_k$  ;
38.                          distância  $\leftarrow |left - right|$  ;
39.                           $d_{(i, ponto)} \leftarrow \infty$  ;
40.                           $d_{(ponto, i)} \leftarrow \infty$  ;
41.                          ponto  $\leftarrow k$  ;
42.                          retorna para linha (35);
43.                      fim se;
44.                  fim se;
45.              fim para todo;
46.              se ( (  $y_i = y_j$  ) e (  $d_{ij} \neq \infty$  ) ) então
47.                  se (  $x_i < x_j$  ) então

```

```

48.                                      $up \leftarrow x_i$  ;
49.                                      $down \leftarrow x_j$  ;
50.                                 fim se;
51.                                 se contrário
52.                                      $up \leftarrow x_j$  ;
53.                                      $down \leftarrow x_i$  ;
54.                                 fim se;
55.                                 distância  $\leftarrow |up - down|$  ;
56.                                 ponto  $\leftarrow j$  ;
57.                             fim se;
58.                             para todo  $k=1, \dots, node$ 
59.                                 se ( (  $k < i$  ) e (  $k < j$  ) e (  $y_i = y_k$  ) e (
                                      $d_{ij} < \infty$  ) ) então
60.                                     se ( (  $x_k > up$  ) e (  $x_k < down$  ) ) então
61.                                          $up \leftarrow x_j$  ;
62.                                          $right \leftarrow x_k$  ;
63.                                         distância  $\leftarrow |up - down|$  ;
64.                                          $d_{(i, ponto)} \leftarrow \infty$  ;
65.                                          $d_{(ponto, i)} \leftarrow \infty$  ;
66.                                         ponto  $\leftarrow k$  ;
67.                                         retorna para linha (60);
68.                                     fim se;
69.                                 fim se;
70.                             fim para todo;
71.                         fim se;
72.                         fim para todo;
73.                     fim para todo;
74.                     para todo  $i=1, \dots, node$ 
75.                         contagem  $\leftarrow 0$ ;
76.                         para todo  $j=1, \dots, node$ 
77.                             se (  $d_{ij} < \infty$  ) então
78.                                 contagem = contagem + 1;
79.                             fim se;
80.                         fim para todo;
81.                         se ( contagem % 2 < 0 ) então
82.                              $d_{ii} \leftarrow 0$ ;
83.                         fim se;
84.                     fim para todo;
Fim.

```

APÊNDICE D – Pseudocódigo heurística carteiro chinês não direcionado

Início:

```

1.   somavizinho ← 0;
2.   para todo i=1,...,node
3.       para todo j=1,...,node
4.           vizinhoij ← 0;
5.       fim para todo;
6.   fim para todo;
7.   para todo i=1,...,node
8.       para todo j=1,...,node
9.           se ( ( dii < ∞ ) e ( dii < 0 ) ) então
10.              vizinhoij ← 1;
11.              somavizinho ← somavizinho + 1;
12.           fim se;
13.       fim para todo;
14.   fim para todo;
15.   para todo t=1,...,node
16.       para todo i=1,...,node
17.           para todo j=1,...,n
18.               fluxoij ← 0;
19.           fim para todo;
20.       fim para todo;
21.       pivô ← t ;
22.       next ← t ;
23.       FO ← 0;
24.       Alocar memória dinâmica para receber os dados (número de nó, coordenadas x
e y);
25.       menor_f ← soma_vizinho;
26.       menor_d ← ∞ ;
27.       para todo i=1,...,node
28.           se ( ( pivô < i ) e ( vizinho(pivô,i) ) e ( fluxo(pivô,i) + fluxo(i,pivô) = 0 ) e
( d(pivô,i) <= menor_d ) ) então
29.               next ← i ;
30.               menor_d ← d(pivô,i) ;
31.               menor_f ← fluxo(pivô,i) + fluxo(i,pivô)
32.           fim se;
33.       se ( next = pivô )
34.           para todo i=1,...,node
35.               se ( ( pivô < i ) e ( vizinho(pivô,i) ) e ( fluxo(pivô,i) + fluxo(i,pivô)
<= menor_f ) e ( d(pivô,i) <= menor_d ) ) então
36.                   next ← i ;
37.                   menor_d ← d(pivô,i) ;
38.                   menor_f ← fluxo(pivô,i) + fluxo(i,pivô) ;
39.               fim se;
40.           fim para todo;

```

```

41.      fim se;
42.      FO = FO +  $d_{(pivô,next)}$  ;
43.       $fluxo_{(pivô,next)} = fluxo_{(pivô,next)} + 1$ 
44.      Alocar memória dinâmica para receber os dados (número de nó, coordenadas x
e y);
45.      pivô  $\leftarrow next$ ;
46.      para todo  $i=1,\dots,node$ 
47.          para todo  $j=1,\dots,node$ 
48.              se ( ( $vizinho_{ij}$ ) e ( $i < j$ ) e (  $fluxo_{ij}+fluxo_{ji} = 0$ ) ) então
49.                  retorna para linha (25);
50.                  break;
51.          fim se;
52.      fim para todo;
53.      fim para todo;
54.      soma_fluxo  $\leftarrow 0$ ;
55.      para todo  $i=1,\dots,node$ 
56.          para todo  $j=1,\dots,node$ 
57.              soma_fluxo = soma_fluxo +  $fluxo_{ij}$  ;
58.          fim para todo;
59.      fim para todo;
60.      se ( (FO < novo_FO) ou (soma_fluxo < novo_soma_fluxo) ) então
61.          novo_FO  $\leftarrow$  FO;
62.          novo_soma_fluxo = soma_fluxo;
63.          para todo  $k=1,\dots,node$ 
64.              Alocar memória dinâmica para receber os dados (número de nó,
coordenada x e y);
65.          fim para todo;
66.      fim se;
67.      fim para todo;
Fim.

```

APÊNDICE E – *G-code* gerado na aplicação

Início:

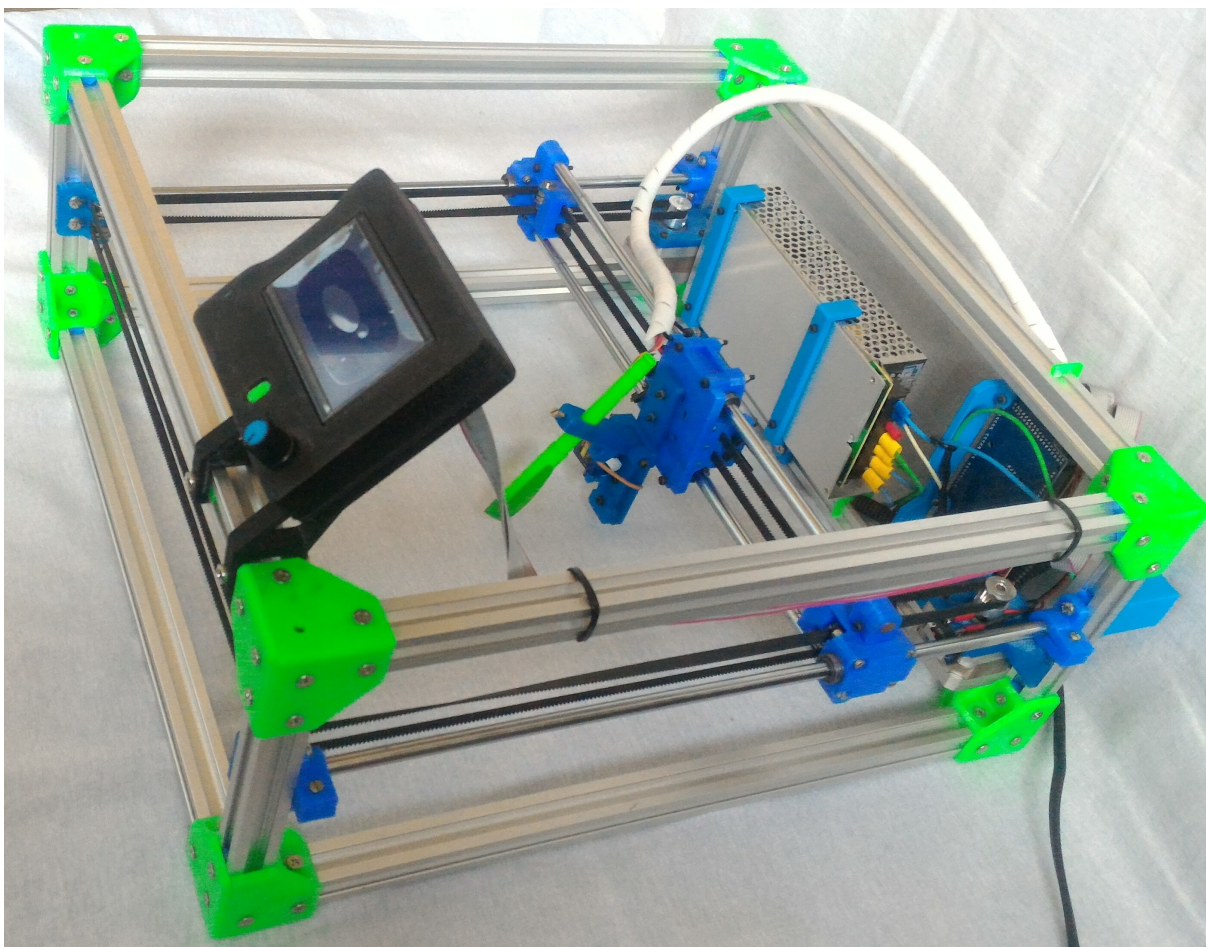
- | | | |
|-----|--------------------|--------------------------------|
| 1. | % | (inicializar o G-code) |
| 2. | G21 | (definir as coordenadas em mm) |
| 3. | M280 P0 S25 | (levantar o servo) |
| 4. | | |
| 5. | G0 X 41 Y 53 F 400 | (ponto 0, mover a origem) |
| 6. | | |
| 7. | M280 P0 S40 | (baixar o servo) |
| 8. | | |
| 9. | G1 X 41 Y 55 | (ponto 1) |
| 10. | G1 X 41 Y 57 | (ponto 2) |
| 11. | G1 X 41 Y 59 | (ponto 3) |
| 12. | G1 X 41 Y 61 | (ponto 4) |
| 13. | G1 X 41 Y 63 | (ponto 5) |
| 14. | G1 X 41 Y 65 | (ponto 6) |
| 15. | G1 X 41 Y 67 | (ponto 7) |
| 16. | G1 X 41 Y 68 | (ponto 8) |
| 17. | G1 X 41 Y 69 | (ponto 9) |
| 18. | G1 X 41 Y 71 | (ponto 10) |
| 19. | G1 X 41 Y 73 | (ponto 11) |
| 20. | G1 X 41 Y 75 | (ponto 12) |
| 21. | G1 X 41 Y 77 | (ponto 13) |
| 22. | G1 X 44 Y 77 | (ponto 14) |
| 23. | G1 X 44 Y 75 | (ponto 15) |
| 24. | G1 X 44 Y 73 | (ponto 16) |
| 25. | G1 X 44 Y 71 | (ponto 17) |
| 26. | G1 X 44 Y 69 | (ponto 18) |
| 27. | G1 X 41 Y 69 | (ponto 19) |
| 28. | G1 X 41 Y 68 | (ponto 20) |
| 29. | G1 X 0 Y 68 | (ponto 21) |
| 30. | G1 X 0 Y 92 | (ponto 22) |
| 31. | G1 X 0 Y 115 | (ponto 23) |
| 32. | G1 X 0 Y 138 | (ponto 24) |
| 33. | G1 X 0 Y 147 | (ponto 25) |
| 34. | G1 X 0 Y 149 | (ponto 26) |
| 35. | G1 X 26 Y 149 | (ponto 27) |
| 36. | G1 X 26 Y 147 | (ponto 28) |
| 37. | G1 X 26 Y 138 | (ponto 29) |
| 38. | G1 X 31 Y 138 | (ponto 30) |

39.	G1 X 53 Y 138	(ponto 31)
40.	G1 X 57 Y 138	(ponto 32)
41.	G1 X 57 Y 115	(ponto 33)
42.	G1 X 57 Y 92	(ponto 34)
43.	G1 X 41 Y 92	(ponto 35)
44.	G1 X 31 Y 92	(ponto 36)
45.	G1 X 31 Y 115	(ponto 37)
46.	G1 X 31 Y 138	(ponto 38)
47.	G1 X 26 Y 138	(ponto 39)
48.	G1 X 0 Y 138	(ponto 40)
49.	G1 X 0 Y 147	(ponto 41)
50.	G1 X 26 Y 147	(ponto 42)
51.	G1 X 53 Y 147	(ponto 43)
52.	G1 X 53 Y 149	(ponto 44)
53.	G1 X 26 Y 149	(ponto 45)
54.	G1 X 26 Y 147	(ponto 46)
55.	G1 X 26 Y 138	(ponto 47)
56.	G1 X 0 Y 138	(ponto 48)
57.	G1 X 0 Y 115	(ponto 49)
58.	G1 X 31 Y 115	(ponto 50)
59.	G1 X 57 Y 115	(ponto 51)
60.	G1 X 57 Y 138	(ponto 52)
61.	G1 X 53 Y 138	(ponto 53)
62.	G1 X 53 Y 147	(ponto 54)
63.	G1 X 53 Y 149	(ponto 55)
64.	G1 X 53 Y 147	(ponto 56)
65.	G1 X 53 Y 138	(ponto 57)
66.	G1 X 31 Y 138	(ponto 58)
67.	G1 X 31 Y 115	(ponto 59)
68.	G1 X 31 Y 92	(ponto 60)
69.	G1 X 0 Y 92	(ponto 61)
70.	G1 X 0 Y 115	(ponto 62)
71.	G1 X 0 Y 138	(ponto 63)
72.	G1 X 0 Y 147	(ponto 64)
73.	G1 X 0 Y 149	(ponto 65)
74.	G1 X 0 Y 147	(ponto 66)
75.	G1 X 0 Y 149	(ponto 67)
76.	G1 X 0 Y 147	(ponto 68)
77.	G1 X 0 Y 138	(ponto 69)
78.	G1 X 0 Y 115	(ponto 70)
79.	G1 X 0 Y 92	(ponto 71)
80.	G1 X 0 Y 68	(ponto 72)

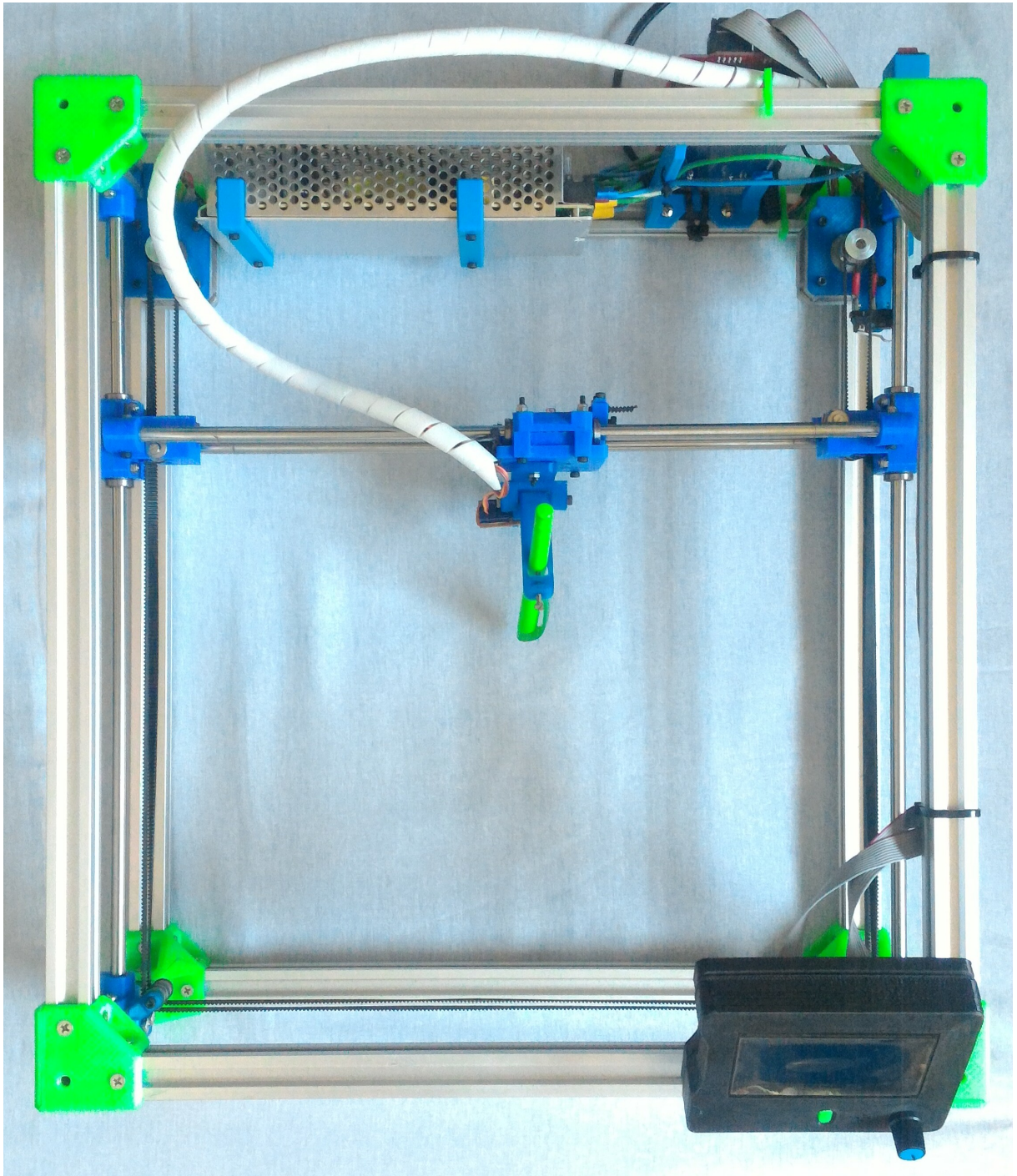
81.	G1 X 0 Y 43	(ponto 73)
82.	G1 X 41 Y 43	(ponto 74)
83.	G1 X 41 Y 46	(ponto 75)
84.	G1 X 41 Y 48	(ponto 76)
85.	G1 X 41 Y 51	(ponto 77)
86.	G1 X 41 Y 53	(ponto 78)
87.	G1 X 47 Y 53	(ponto 79)
88.	G1 X 47 Y 55	(ponto 80)
89.	G1 X 47 Y 57	(ponto 81)
90.	G1 X 47 Y 59	(ponto 82)
91.	G1 X 47 Y 61	(ponto 83)
92.	G1 X 47 Y 63	(ponto 84)
93.	G1 X 47 Y 65	(ponto 85)
94.	G1 X 47 Y 67	(ponto 86)
95.	G1 X 47 Y 69	(ponto 87)
96.	G1 X 44 Y 69	(ponto 88)
97.	G1 X 44 Y 71	(ponto 89)
98.	G1 X 41 Y 71	(ponto 90)
99.	G1 X 41 Y 73	(ponto 91)
100.	G1 X 44 Y 73	(ponto 92)
101.	G1 X 44 Y 75	(ponto 93)
102.	G1 X 41 Y 75	(ponto 94)
103.	G1 X 41 Y 77	(ponto 95)
104.	G1 X 41 Y 92	(ponto 96)
105.	G1 X 31 Y 92	(ponto 97)
106.	G1 X 0 Y 92	(ponto 98)
107.	G1 X 0 Y 68	(ponto 99)
108.	G1 X 0 Y 43	(ponto 100)
109.	G1 X 0 Y 0	(ponto 101)
110.	G1 X 53 Y 0	(ponto 102)
111.	G1 X 53 Y 43	(ponto 103)
112.	G1 X 55 Y 43	(ponto 104)
113.	G1 X 55 Y 46	(ponto 105)
114.	G1 X 55 Y 48	(ponto 106)
115.	G1 X 55 Y 51	(ponto 107)
116.	G1 X 55 Y 53	(ponto 108)
117.	G1 X 47 Y 53	(ponto 109)
118.	G1 X 47 Y 55	(ponto 110)
119.	G1 X 41 Y 55	(ponto 111)
120.	G1 X 41 Y 57	(ponto 112)
121.	G1 X 47 Y 57	(ponto 113)
122.	G1 X 47 Y 59	(ponto 114)

123.	G1 X 41 Y 59	(ponto 115)
124.	G1 X 41 Y 61	(ponto 116)
125.	G1 X 47 Y 61	(ponto 117)
126.	G1 X 47 Y 63	(ponto 118)
127.	G1 X 41 Y 63	(ponto 119)
128.	G1 X 41 Y 65	(ponto 120)
129.	G1 X 47 Y 65	(ponto 121)
130.	G1 X 47 Y 67	(ponto 122)
131.	G1 X 41 Y 67	(ponto 123)
132.	G1 X 41 Y 68	(ponto 124)
133.	G1 X 0 Y 68	(ponto 125)
134.	G1 X 0 Y 43	(ponto 126)
135.	G1 X 41 Y 43	(ponto 127)
136.	G1 X 53 Y 43	(ponto 128)
137.	G1 X 55 Y 43	(ponto 129)
138.	G1 X 53 Y 43	(ponto 130)
139.	G1 X 41 Y 43	(ponto 131)
140.	G1 X 41 Y 46	(ponto 132)
141.	G1 X 55 Y 46	(ponto 133)
142.	G1 X 55 Y 48	(ponto 134)
143.	G1 X 41 Y 48	(ponto 135)
144.	G1 X 41 Y 51	(ponto 136)
145.	G1 X 55 Y 51	(ponto 137)
146.		
147.	M280 P0 S25	(levantar o servo)
148.		
149.	G0 X 41 Y 53	(ponto 0, voltar a origem)
150.		
151.	M280 P0 S25	(levantar o servo)
152.	%	(finalizar o G-code)
Fim.		

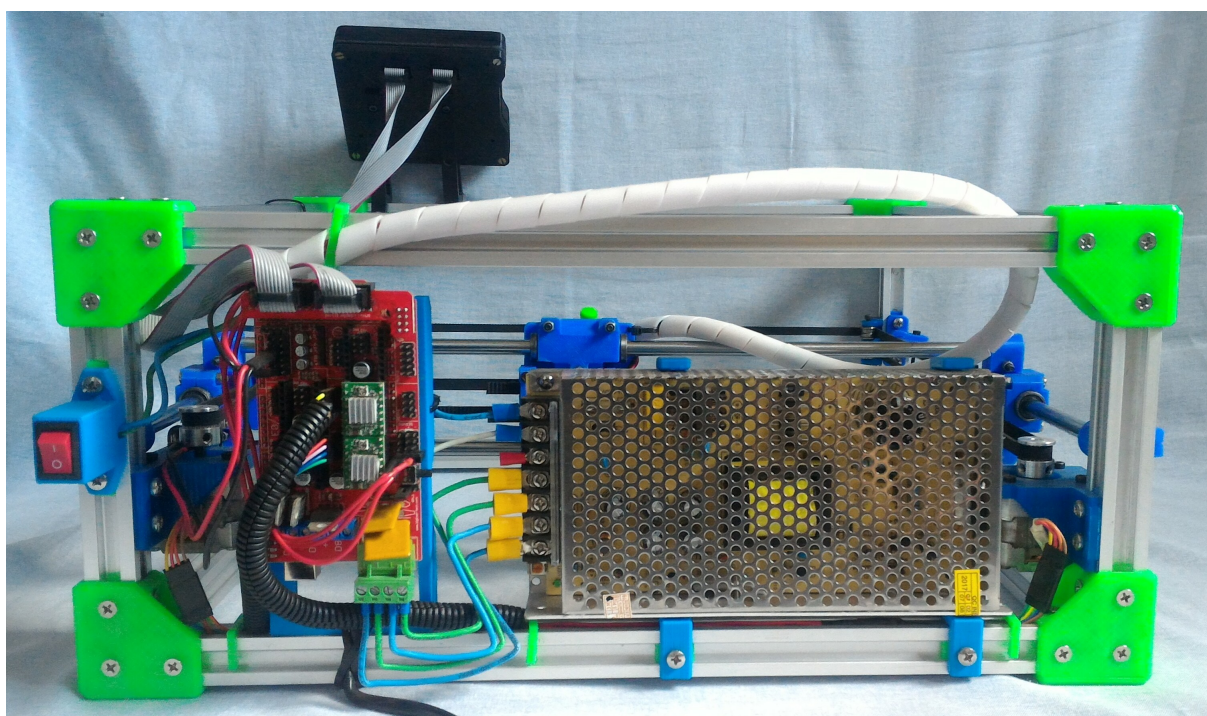
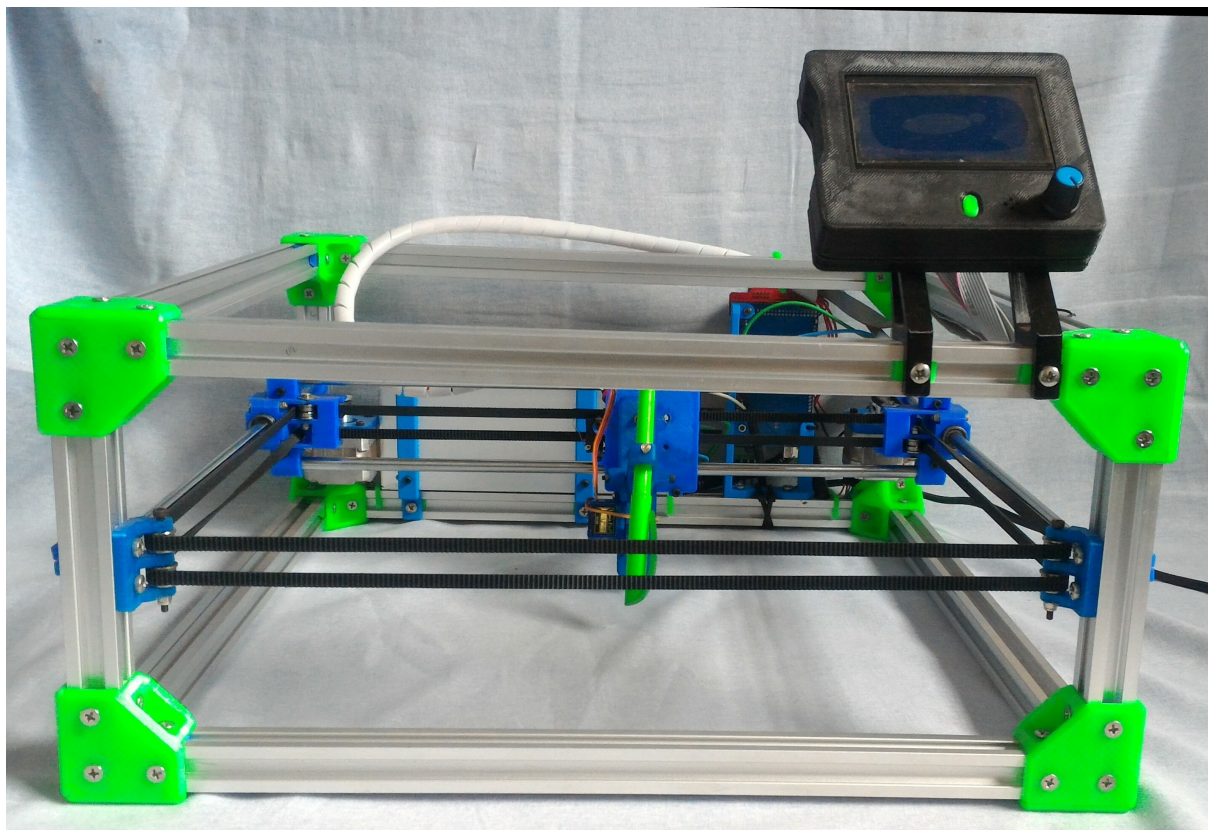
**APÊNDICE F.a – Foto do protótipo da máquina de corte bidimensional
(vista perspectiva isométrica)**



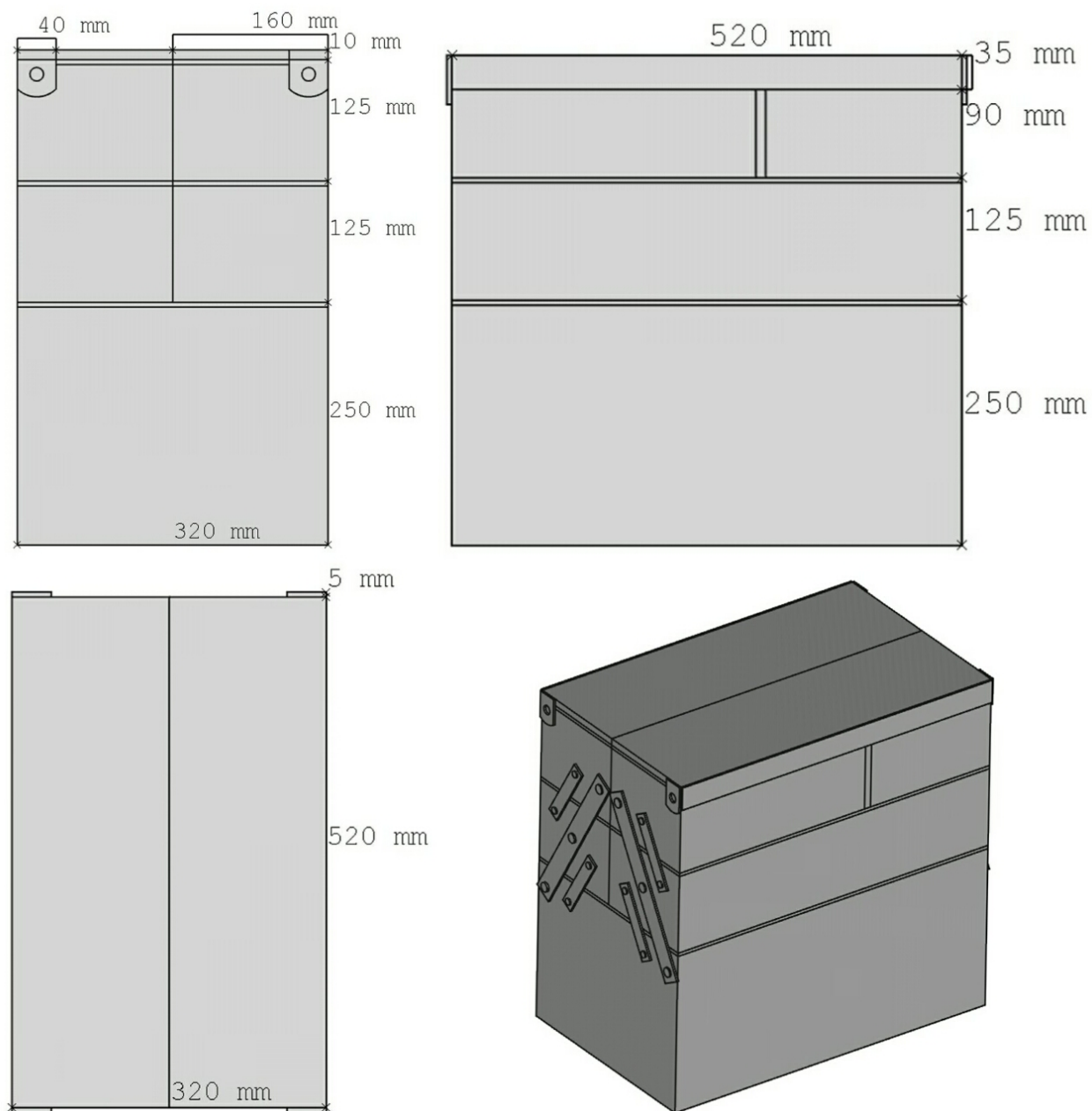
**APÊNDICE F.b – Foto do protótipo da máquina de corte bidimensional
(vista superior)**



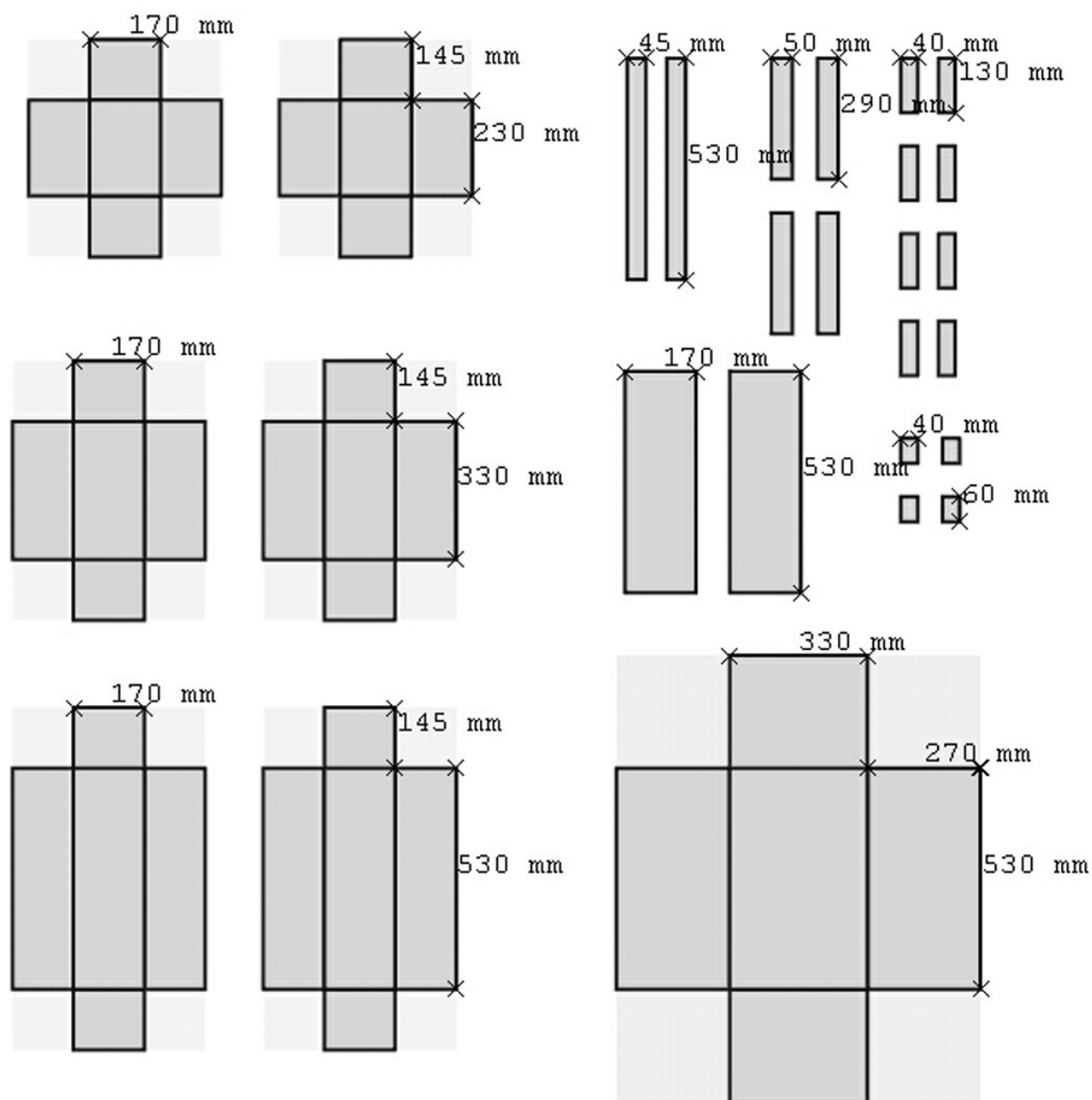
APÊNDICE F.c – Foto do protótipo da máquina de corte bidimensional
(vista frontal e traseira)



APÊNDICE G.a – Desenho da caixa de ferramenta
(vista frontal, lateral, superior e perspectiva isométrica)



APÊNDICE G.b – Desenho da caixa de ferramenta
(explodido)



APÊNDICE G.c – Desenho da caixa de ferramenta
(detalhe da orelha)

