

## Programação visual de algoritmos aplicada à geração de código-g para manufatura aditiva

Luiz F. Farias Serravalle<sup>1,2</sup>, Jorge Vicente Lopes da Silva<sup>2</sup>, José Luis Dávila<sup>2</sup>

{lfserravalle, jorge.silva, jose.davila}@cti.gov.br

<sup>1</sup>Faculdade de Engenharia Mecânica – Universidade Estadual de Campinas (UNICAMP)

<sup>2</sup>Laboratório Aberto de Impressão 3D– LAprint CTI/MCTI Renato Archer – Campinas/SP

**Abstract.** Additive manufacturing (AM) is a group of processes in which material is selectively structured layer upon layer. Usually, objects are manufactured starting from a 3D computational model, which is sliced to generate the trajectories for the material structuring. Algorithm-Aided Design (AAD) is a programming language that combines Computer-Aided Design (CAD) and programming for the parameterization of geometries. This work used AAD to generate g-codes for AM processes for material extrusion. The results obtained support the potential use of AAD for applications in AM.

**Keywords:** Additive manufacturing, Algorithm-Aided Design, g-code, 3D printing.

**Resumo.** A manufatura aditiva (MA) é um conjunto de processos de fabricação em que material é estruturado seletivamente em camadas sucessivas. Usualmente, os objetos são fabricados partindo de um modelo computacional 3D, o mesmo que é fatiado visando gerar as trajetórias para a estruturação do material. O desenho assistido por algoritmos (AAD/Algorithm-Aided Design) é uma linguagem que combina ferramentas CAD (Computer-Aided Design) e de programação para a parametrização de geometrias. Neste trabalho, utilizou-se AAD para a geração de código-g para processos de MA por extrusão de material. Os resultados obtidos sustentam o potencial do uso de AAD para aplicações em MA.

**Palavras chave:** Manufatura Aditiva (MA), Algoritmos AAD, código-g, impressão 3D.

### 1. Introdução

A manufatura aditiva (MA), comumente referida como impressão 3D, é um conjunto de processos de fabricação, em que geometrias tridimensionais são formadas pela sucessiva adição de material em camadas [1]. Desde o lançamento da primeira impressora 3D funcional, em meados de 1984, a MA tem revolucionado a maneira que peças são projetadas e manufaturadas, permitindo que geometrias anteriormente

impossíveis de serem fabricadas por outros métodos, sejam produzidas através da superposição de camadas [2]. Atualmente, diversas tecnologias de MA como Estereolitografia (*SLA*), Sinterização Seletiva a Laser (*SLS*), Moldagem por Fusão e Deposição (*FDM*), jato de tinta (*inkjet*), entre outras, são utilizadas em diferentes áreas da engenharia, medicina e na pesquisa, não apenas para a fabricação de protótipos, mas também para fabricação de peças de uso final [3].

De maneira geral, o fluxo de trabalho da impressão 3D pode ser dividido em 3 etapas: (i) pré-processamento, (ii) impressão e (iii) pós-processamento. O processo inicia-se com o design da peça a ser manufaturada, comumente em um *software* de desenho assistido por computador (*CAD*). Em seguida, um algoritmo de fatiamento decompõe o contorno da malha do arquivo \*.stl. Assim, uma série de trajetórias (*toolpaths*), que o cabeçote da impressora deverá percorrer depositando material, são definidas. Por fim, a impressora 3D reproduz os *toolpaths* e fabrica a peça camada por camada, a mesma que pode ser direcionada para uso final ou para etapas adicionais de pós-processamento, tendo como objetivo melhorar a qualidade superficial da peça ou também para retirada de suportes [4].

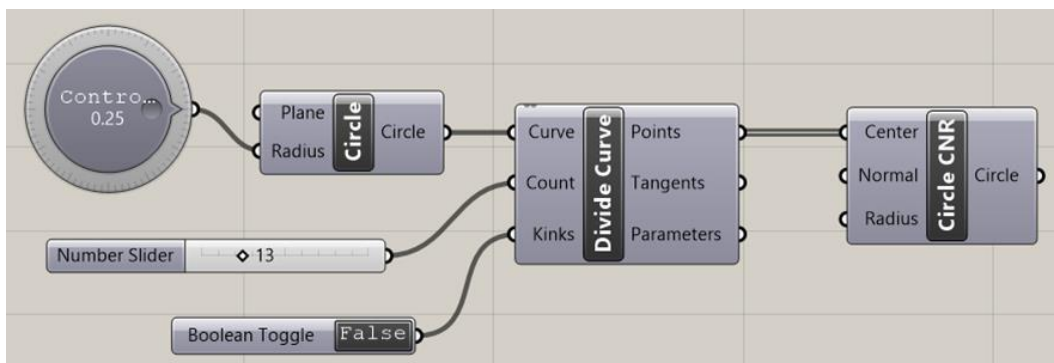
Atualmente, existe uma ampla diversidade de *software* especializado para modelagem *CAD*. De maneira similar, existem várias opções de *software* comercial e *open-source* para o fatiamento de modelos 3D; programas como Repetier-Host, Slicer, MatterControl, Cura, etc., utilizam diferentes algoritmos para realizar o fatiamento do modelo computacional *CAD*. Nesse ponto, é importante salientar que o algoritmo de fatiamento terá grande influência no tempo final de manufatura, qualidade, e precisão da peça produzida [5]. O código-g é uma das linguagens de programação mais utilizadas na indústria de manufatura assistida por computador (*CAM*). No fluxo de trabalho ordinário da manufatura aditiva, apenas o arquivo \*.gcode é interpretado pelo *firmware*; o código contém os comandos que controlam a operação de motores, termistores e outros componentes da impressora 3D, assim, é definido o caminho e operação do cabeçote da impressora.

O escopo deste trabalho de pesquisa se alinha com o desenvolvimento de algoritmos por meio de programação visual, os mesmos que foram aplicados para a geração de código-g. O uso de ferramentas *AAD* (acrônimo em inglês para “Desenho Assistido por Algoritmos) permite uma integração apropriada do processo de design e fatiamento de uma peça a ser manufaturada aditivamente; a utilização de *software AAD* admite uma abordagem holística em todo o processo de design, isto em termos de controle de superfícies geradas e parametrização de geometrias. Nessa pesquisa foram desenvolvidos algoritmos para a geração de geometrias parametrizadas, as mesmas que foram compiladas no formato de códigos-g. Os desenvolvimentos foram implementados em um único ambiente de programação, eliminando assim a necessidade de *software* de fatiamento. Os códigos-g compilados foram verificados por meio de simulação de movimento utilizando *software CAM (Computer-Aided Manufacturing)*. Em conclusão, o uso de *AAD* constitui uma ferramenta promissora para a geração de código-g, principalmente para aplicações específicas como fabricação de *scaffolds*, fármacos modulares e com morfologia controlada, metamateriais, entre outros.

## 2. Metodologia

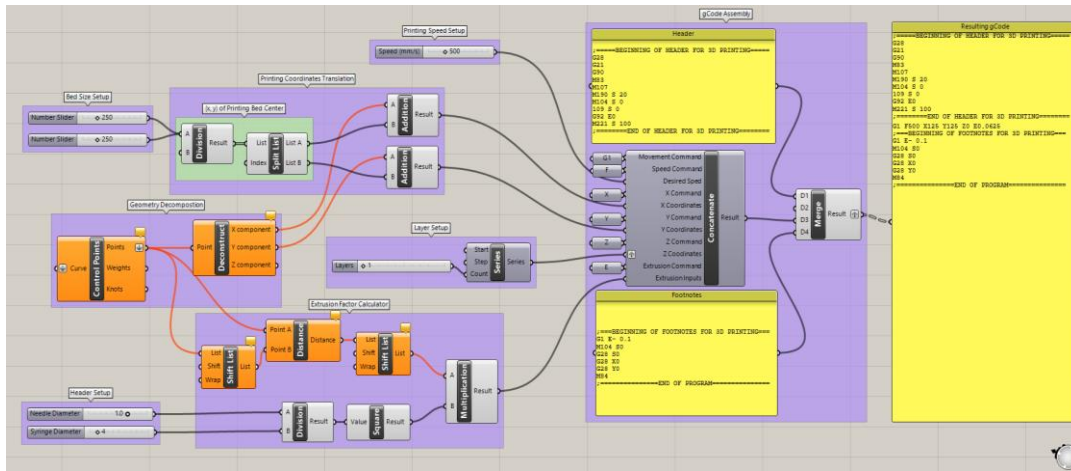
### 2.1 Desenvolvimento e implementação de algoritmos

Para o desenvolvimento de algoritmos por meio de programação visual foi utilizado o *software* Rhinoceros 7<sup>®</sup>, especificamente o *plug-in* Grasshopper<sup>®</sup>. Esse *plug-in* possui uma suíte de funções em blocos, as mesmas que são adequadas para composição, análise e decomposição de geometrias a partir de parâmetros especificados pelo usuário. Dessa maneira, algoritmos podem ser criados para desenvolver inúmeras geometrias, mas que compartilhem determinados atributos. Nesse contexto, algoritmos visuais foram desenvolvidos e implementados para a compilação de código-g compatível com impressoras 3D do projeto aberto RepRap [6]. Seis geometrias foram parametrizadas e algoritmos desenvolvidos e implementados; parâmetros genéricos como diâmetro, comprimento e largura, podem ser facilmente modificados em cada algoritmo. Além disso, parâmetros específicos como por exemplo: número de divisões radiais e número de células de Voronoi, podem ser facilmente controlados pelo usuário, tudo por meio de *sliders*. A Figura 1 ilustra o algoritmo de geração de uma das seis geometrias implementadas. Como observado, o usuário pode escolher rapidamente os parâmetros geométricos por meio dos controladores tipo *slider*, o que permite definir as variáveis de entrada, que posteriormente permitirão a compilação de código-g.



**Figura 1.** Algoritmo desenvolvido para geração de geometrias parametrizadas. Fonte: De autoria própria.

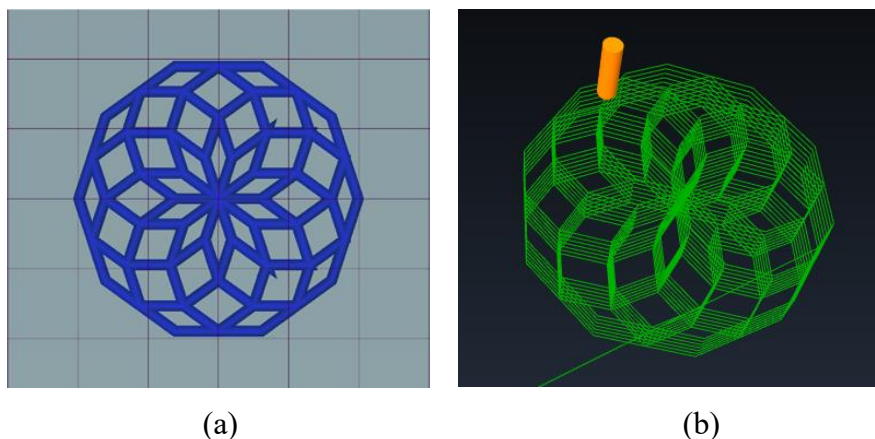
Uma vez finalizado o desenvolvimento e implementação de algoritmos para a parametrização de geometrias, o próximo passo foi o desenvolvimento de algoritmos para compilação de código-g. A Figura 2 abaixo retrata o algoritmo desenvolvido, separado em diferentes grupos de blocos para diferentes funções. O fluxo de informação no algoritmo é bastante característico e diferente daqueles utilizados em algoritmos de fatiamento disponíveis no mercado [7]. Inicialmente, o algoritmo da geometria é conectado ao algoritmo da Figura 2 utilizando um bloco “Curve.” Em seguida, a curva gerada pelo algoritmo da geometria é decomposta em uma série de pontos utilizando um bloco de função “Divide Curve”, que divide uma curva com base na mudança de direção de um vetor tangente. Com os pontos extraídos da *Polyline*, o código armazena as coordenadas e transforma-as em códigos de comando de movimentação e extrusão para a impressora 3D. É importante destacar que o comando que permitirá definir as coordenadas é:  $G1\ X<pos>\ Y<pos>\ Z<pos>\ E\ <pos>$ , que corresponde à movimentação linear utilizando as coordenadas (x,y,z) da trajetória parametrizada.



ao redor do mundo [6]. Dessa maneira, a compatibilidade foi testada utilizando o *software open-source* Repetier-Host V2.2.2 [10], o mesmo que é compatível com impressoras 3D baseadas no projeto RepRap.

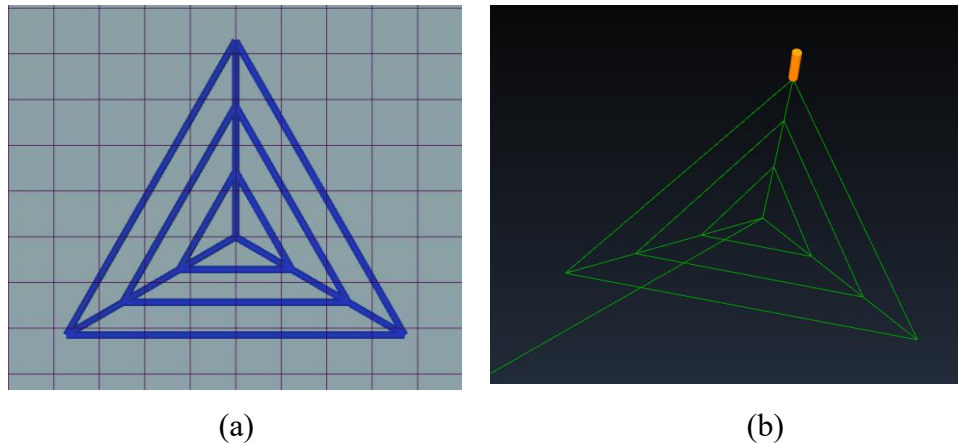
### 3. Resultados e discussão

A Figura 3a ilustra a primeira geometria compilada no *software* Repetier-Host. O código-g foi gerado utilizando o compilador desenvolvido. É possível ver que o arquivo \*.gcode gerado pelo compilador reproduziu com sucesso a geometria dos algoritmos e gerou um código-g interpretável pelo *firmware*. O *software* Repetier-Host também fornece algumas informações adicionais sobre a impressão chamadas “Estatísticas de impressão.” Analisando os dados obtidos, é possível ver que a impressão apresenta o mesmo número de camadas que o selecionado no compilador do Grasshopper®, além do tempo total de impressão, que é pertinente com as dimensões finais da geometria. Posteriormente, analisando a simulação no CAMotics, apresentada na Figura 3b, é possível verificar que o código-g reproduz todos os *toolpaths* parametrizados por meio de *AAD*.



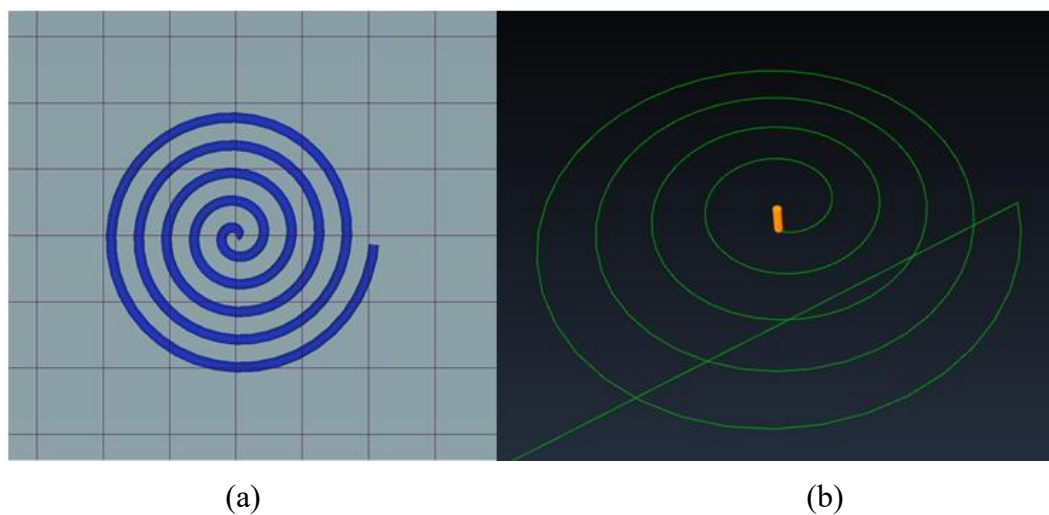
**Figura 3.** Geometria do primeiro algoritmo implementado: (a) Compatibilidade com o *software* Repetier-Host e (b) código-g no *software* CAMotics. Fonte: De autoria própria.

A Figura 4(a,b) retrata a geometria gerada pelo segundo algoritmo desenvolvido no Repetier-Host e CAMotics, respectivamente. Neste caso em específico, é possível ver mais nitidamente o trajeto inicial feito pelo cabeçote de sua posição inicial na impressora 3D até o centro da mesa de impressão, onde as geometrias foram programadas para serem reproduzidas. No entanto, essa movimentação inicial não é feita utilizando a deposição de materiais, portanto, não há material nesse *toolpath* em específico. Adicionalmente, outras geometrias podem ser geradas utilizando o mesmo algoritmo parametrizado. No caso da Figura 4, um polígono de 3 lados e 2 divisões radiais foi regenerado, no entanto, é possível gerar polígonos de  $N$  lados com  $N$  divisões radiais para numerosas opções de geometrias para impressão.



**Figura 4.** Geometria do segundo algoritmo implementado: (a) Compatibilidade com o *software* Repetier-Host e (b) código-g no *software* CAMotics. Fonte: De autoria própria.

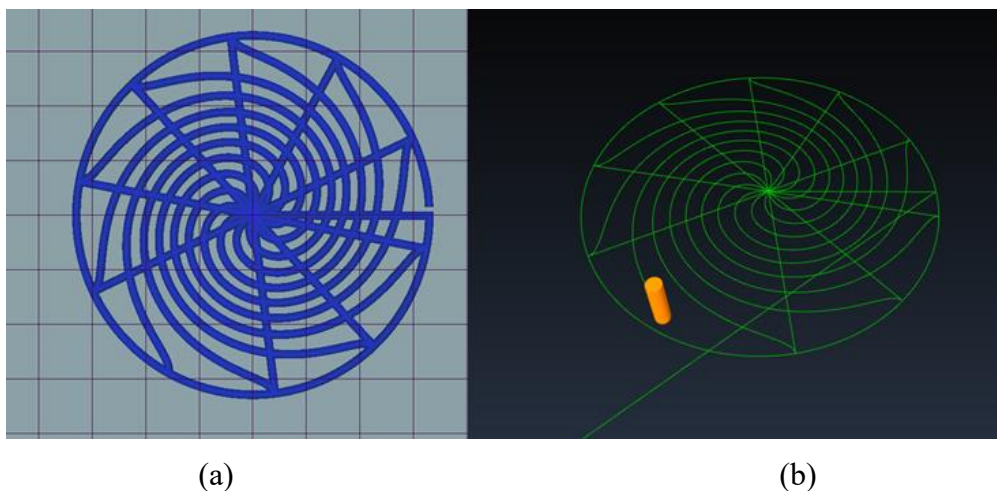
O terceiro algoritmo desenvolvido, apresentado na Figura 5, permite gerar uma geometria com formato de uma espiral decrescente. A velocidade de impressão é constante durante todo o percurso, desta maneira, o modelo matemático para a extrusão é utilizado para garantir que o material depositado seja o mesmo independente do raio instantâneo da espiral.



**Figura 5.** Geometria do terceiro algoritmo implementado: (a) Compatibilidade com o *software* Repetier-Host e (b) código-g no *software* CAMotics. Fonte: De autoria própria.

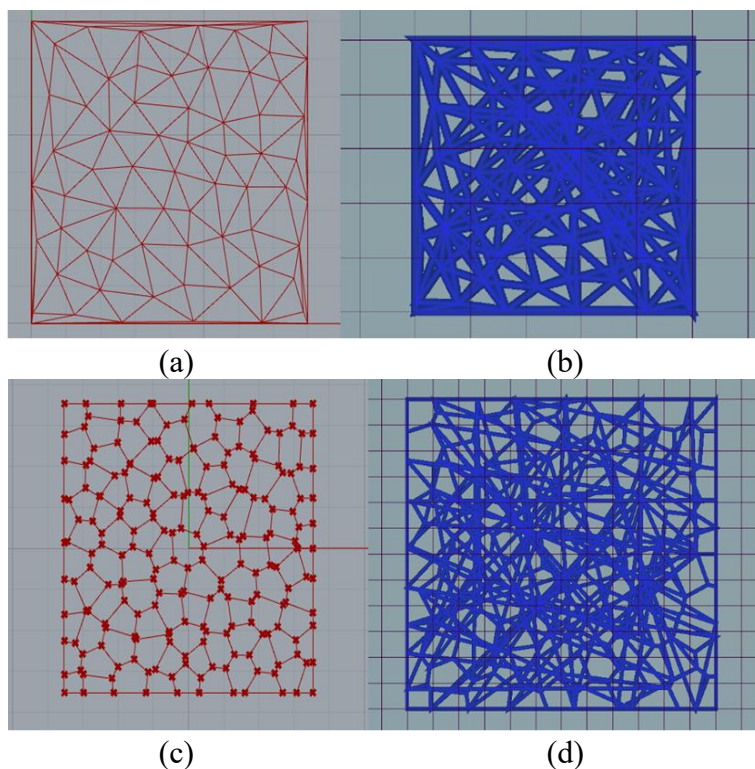
O quarto algoritmo, representado na Figura 6, combina algumas as características da espiral gerada no algoritmo anterior, com um número maior de segmentos, para uma geometria final porosa.





**Figura 6.** Geometria do quarto algoritmo implementado: (a) Compatibilidade com o *software* Repetier-Host e (b) código-g no *software* CAMotics. Fonte: De autoria própria.

Os dois algoritmos restantes utilizam os processos de decomposição de espaços Voronoi e Delaunay, respectivamente. Isto permite criar regiões porosas dentro de uma região retangular predefinida. Essas geometrias possuem uma complexidade maior em comparação com as geradas anteriormente. Na Figura 7, são mostradas as geometrias geradas no *software* AAD e no teste de compatibilidade no *software* Repetier-Host.



**Figura 7.** Decomposição de Delaunay no *software*: (a) Grasshopper e (b) Repetier-Host. Decomposição de Voronoi no *software*: (c) Grasshopper e (d) Repetier-Host. Fonte: De autoria própria.

Para as geometrias geradas partindo de algoritmos de Delaunay e Voronoi, a complexidade geométrica faz com que o código-g precise de uma grande quantidade de movimentos de posicionamento para percorrer o *toolpath*. Dessa maneira, esse tipo de algoritmos podem requerer de algoritmos adicionais para otimização da trajetória e do tempo de processo.

Como observado nos algoritmos implementados, o desenho assistido por algoritmos constitui uma ferramenta que permite programar algoritmos de maneira modular [11]. Assim, algoritmos podem ser implementados para diferentes tarefas específicas, desde a geração de *toolpaths* até a definição de parâmetros de processo e compilação de código-g. A modularidade permite integrar diferentes ferramentas ou combinar varios tipos de algoritmos, o que demonstra a versatilidade dessa metodologia para aplicações em MA, principalmente em processos de extrusão de material, os mesmos que são amplamente aplicados em diferentes áreas de pesquisa.

#### 4. Conclusões

Por meio do trabalho desenvolvido é possível identificar a utilidade do desenho assistido por algoritmos (AAD) para aplicações em manufatura aditiva, desde a geração de geometrias parametrizadas até a criação de algoritmos compiladores de código-g. As principais vantagens notadas foram a flexibilidade e agilidade em gerar várias geometrias rapidamente. Além disso, o compilador de código-g permite um controle maior sobre o processo de geração do arquivo \*.gcode, sendo possível uma maior autonomia, principalmente é aplicações específicas que requerem de um maior controle da deposição do material.

#### 5. Referências

1. ISO/ASTM 52900:2015(E). Standard Terminology for Additive Manufacturing – General Principles – Terminology. 2015;
2. Campbell T, Williams C, Ivanova O, Garrett B. Could 3D Printing Change the World? Technologies, Potential, and Implications of Additive Manufacturing. Atl Counc. 2011;
3. Provaggi E, Kalaskar DM. 3D printing families: Laser, powder, nozzle based techniques [Internet]. 3D Printing in Medicine. Elsevier Ltd; 2017. 21–42 p. Available from: <http://dx.doi.org/10.1016/B978-0-08-100717-4.00003-X>
4. Karakurt I, Lin L. 3D printing technologies: techniques, materials, and post-processing. Curr Opin Chem Eng [Internet]. 2020;28:134–43. Available from: <https://doi.org/10.1016/j.coche.2020.04.001>
5. Baumann F, Bugdayci H, Grunert J, Keller F, Roller D. Influence of slicing tools on quality of 3D printed parts. Comput Aided Des Appl. 2016;13(1):14–31.
6. Jones R, Haufe P, Sells E, Iravani P, Olliver V, Palmer C, et al. Reprap - The replicating rapid prototyper. Robotica. 2011;29(1 SPEC. ISSUE):177–91.
7. Bryła J, Martowicz A. Study on the importance of a slicer selection for the 3d



- printing process parameters via the investigation of g-code readings. *Machines*. 2021;9(8).
8. Dávila JL, Manzini BM, Lopes da Fonsêca JH, Mancilla Corzo IJ, Neto PI, Aparecida de Lima Montalvão S, et al. A parameterized g-code compiler for scaffolds 3D bioprinting. *Bioprinting*. 2022;27(March):e00222.
  9. Coffland J. Open-Source Simulation & Computer Aided Machining [Internet]. Available from: <https://camotics.org/>
  10. Littwin M, Littwin R. Repetier-Host [Internet]. Available from: <https://www.repetier.com/>
  11. Dávila JL, Inforçatti Neto P, Noritomi PY, Coelho RT, Lopes da Silva JV. Algorithms-aided design applied to the tool-paths generation for hybrid manufacturing. *Manuf Lett*. 2021;27:53–7.