Available online at www.sciencedirect.com

ScienceDirect

Procedia Manufacturing 17 (2018) 371–378

Procedia MANUFACTURING

www.elsevier.com/locate/procedia

28th International Conference on Flexible Automation and Intelligent Manufacturing (FAIM2018), June 11-14, 2018, Columbus, OH, USA

Tuning the Center of Gravity of 3D Printed Artifacts

Mert Keles[a], Ulas Yaman[a,*]

[a]*Department of Mechanical Engineering, Middle East Technical University, Ankara 06800, Turkey*

Abstract

In this study, we propose an Algorithms-Aided Design (AAD) approach to shift the center of gravity of 3D printed artifacts to a predefined location by creating a heterogeneous internal structure utilizing the same type of material. When the conventional design and fabrication pipeline of 3D printers and additive manufacturing machinery is employed, information about the interior of the artifacts is lost during the conversion of the design files to the STL file format. This de facto file standard only stores the boundary information of the objects. Even though the designed artifact has heterogeneous interior in the Computer Aided Design (CAD) software, after the conversion it becomes a homogeneous solid. Our method does not require an STL file, since we are using a query-based approach in which the built-in algorithm communicates with CAD software to acquire the necessary information about the design for manufacturing. According to the proposed pipeline, the designed artifact in CAD software is first decomposed into voxels having predefined sizes with AAD add-on software. Then, the desired center of gravity and the amount of extra material available are entered by the user and this additional material is distributed to the voxels by our developed algorithm so that the center of gravity of the final artifact is at the predefined location. At the end of the design process, filling percentages of some voxels is altered which made the structure internally heterogeneous. Then the final structure is directly sliced and the trajectories are converted to G-codes. Using the generated file, artifacts are printed on a desktop FFF printer. With the developed algorithm, we can modify the coordinates of the center of gravity of any shape by adjusting their interior structures and fabricate them on FFF printers.

Keywords: Additive manufacturing; 3D printing, FFF; FDM, center of gravity

10.1016/j.promfg.2018.10.059

1. Introduction

Modifying the mechanical properties of 3D printed artifacts is highly studied in the literature. Many researchers have modified mechanical properties such as density, strength, topology etc. Stave et al. [1] proposed three solutions to improve the strength of the 3D printed artifacts. Those of which are adding struts (supports), thickening weak parts and hollowing some of the parts to decrease the weight at critical areas. Zhou et al. [2] also proposed an optimization method for strengthening the 3D printed artifacts while preserving the shape features. Vanek et al. [3] developed an optimization framework for 3D printing time and material by hollowing the interior and creating a shell mesh. Martinez et al. [4] utilized 3D Voronoi structures to make the 3D printed artifacts lighter and more flexible. Wu et al. [5] proposed a novel method to adjust the interior structure of an object and provided stable mass properties using the ray-reps. Changing the interior to alter mass properties, namely the center of gravity, is similar to the goal of our method. Ying et al. [6] proposed an algorithm to define an anisotropic porous structure inside the 3D printed artifacts to improve the strength to weight ratio of the artifacts. Although the goal includes a change in the gravitational properties, the study does not mainly focus on adjusting the center of gravity but rather focuses on strength to weight ratio efficiency.

Though many researchers focused on improving mechanical properties, changing the center of gravity in particular is a quite popular and a well-addressed issue. Different solutions with different perspectives have been offered in the previous studies. Prévost et al. [7] suggested a similar method to shift the center of gravity of the printed artifacts to the desired (more stable in that case) position interactively. They have the same objective with us, but we follow a completely different approach. They proceeded by modifying the volume and the surface of the objects while our method does no alteration on such features. We focus on the interior of the objects and directly adjust the mass of the heterogeneous interior. Thus, we preserve every visual aspect of the artifacts. Yamanaka et al. [8] created a density aware 3D printing algorithm to adjust the density of 3D artifacts. The method used additional nodes and their thrusts to modify the density along with the center of gravity, moment of inertia etc. Our method has a familiar approach since we both use additional mass to adjust the mass properties. Li et al. [9] also presented a density variable interior structure modeling technique to control the mass properties and mainly focused on the strength of the 3D objects. The solution emphasizes on the weakest part of the model and adjusts the interior' density using single material to achieve the desired conditions. Christiansen et al. [10] offered two methods to shift the center of gravity of a 3D printed artifact to the desired position. The first method is very similar to our method but with an inversed approach. They retracted infill materials from the interior of the object namely created cavities and either filled with a heavier material or left as a cavity to adjust the center of gravity. However, most of the 3D printers cannot print with more than one material. Thus, filling the cavities with another material is not viable. Moreover, leaving cavities empty is not sufficient to alter the center of gravity significantly. As an inverse approach, our method does not create cavities from a full interior, but fills the empty interior of the artifacts with additional mass which makes a big difference. The second method they offered is to rotate the shape to shift the center of gravity to a more stable position. The outer shape of the artifact is deformed as opposed to the first method and our method. Prévost et al. [11] brought attention to a different kind of balance problem and offered a dynamic balancing system utilizing movable masses. This solution allows the designer to provide multiple centers of mass and shift between them without changing the structure. Yaman et al. [12] proposed a method utilizing the coherence of the slices. It makes use of Voronoi cells to create heterogeneous interior. With the leverage of asymmetric distribution of Voronoi cells, center of gravity of the printed artifact is shifted to the side that has higher dense of Voronoi sites as in the example case of the letter *P*. We, in this paper, took the concept of 3D Voronoi cells and added more mass to increase the effect of Voronoi cells.

2. Method

An algorithm for tuning the center of gravity is developed in this paper and in the following subsections a detailed explanation is provided to deeply understand the dynamics of the algorithm.

2.1. Aim of the algorithm

The main objective of the algorithm is to create a secondary shape inside the input geometry so that the overall center of gravity (CoG) of the printed artifact is in the desired position. In order to realize this objective, the algorithm is divided into main steps. Firstly, voxelization of the initial geometry should be done in order to have a base structure to hold the additional mass to be placed inside the initial geometry. Later, the mass and the CoG of the voxelized geometry are calculated. After the amount of additional mass is determined, center coordinates of the additional mass are found. After the formation of the uniform box around the center of additional mass, it is aimed to fit the created box inside the initial geometry so that the integrity and the accuracy of the desired artifact are maintained. Lastly, direct slicing algorithm is run to obtain the G-codes to fabricate the final geometry with the modified interior structure.

2.2. Work flow of the algorithm

The flowchart of our approach is given in Fig. 1. To initialize the procedure, required inputs of the system should be provided. All the inputs must be introduced into the Grasshopper3D since it is the main platform on which the algorithm runs. CAD file of the object is needed to be imported into the Rhinoceros3D CAD software. Then, using the *Brep* component in Grasshopper3D, boundary representation of the input geometry is loaded into the Grasshopper3D platform. Other inputs (center coordinates of the final object and the mass of the additional material) are defined directly within the platform using *Number Slider*.

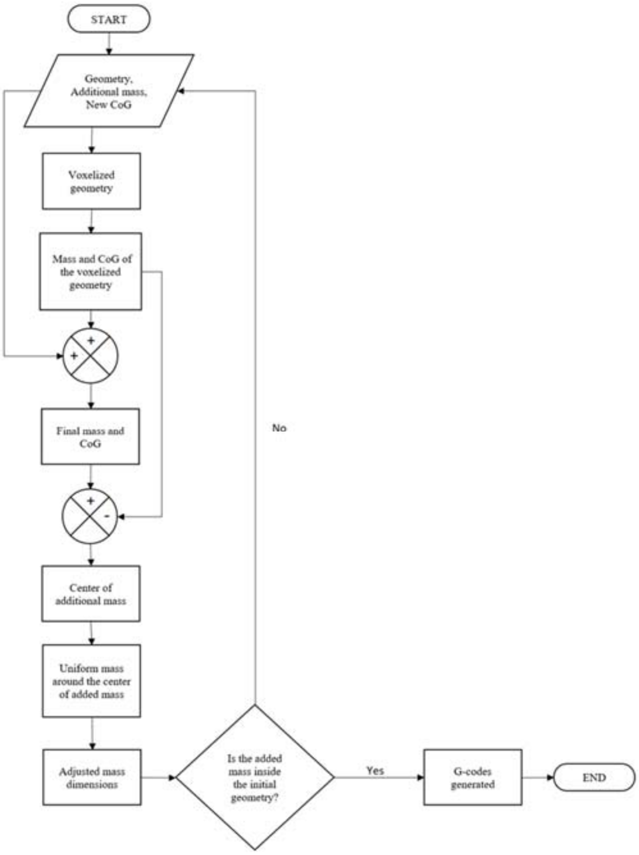

START
Geometry, Additional mass, New CoG
Voxelized geometry
Mass and CoG of the voxelized geometry
+
+
Final mass and CoG
+
-
Center of additional mass
Uniform mass around the center of added mass
Adjusted mass dimensions
Is the added mass inside the initial geometry?
No
Yes
G-codes generated
END

Fig. 1. Flowchart of the algorithm

After the initial process and defining the required inputs, the first process is to divide the geometry into voxels to create a skeleton structure by the walls of these voxels. This is required mainly because when the additional mass is fused to the original geometry, overhanging of the additional mass in a form of a uniform box is inevitable. This situation would result in a decrease in the accuracy of the z-axis and eventually failure of the main objective of the process, which is shifting of the CoG to the desired location with acceptable amount of errors in all axis. Thus, a skeleton structure is important for bridging the additional mass and keep it compact. For this purpose, bounding box of the geometry is divided into voxels by using *Voronoi* components and Voronoi cells are intersected with the boundary representation of the geometry. Intersections of the cells with the geometry gives us a voxelized artifact.

Before the next step, initial gravitational properties of the given geometry should be determined. Obtaining the CoG and the volume of the 3D object is a straightforward process in Grasshopper3D, done simply by using *Volume* component. However, *Volume* does not compute the interior parts of the geometry (voxels in this case). Moreover, since the object is going to be printed using a line-wise approach on an FFF type of 3D printer, it is convenient to calculate every parameter of the process using the same line-wise approach for each layer. The mass of each layer is calculated by simply adding the lengths of all the segments in the layer and the CoG is determined by taking into account the center of these line segments and their lengths. By computing the mean of each layer properties, overall mass and the CoG of the voxelized structure is obtained.

At this point, mass of the initial voxelized geometry and the mass to be added to adjust the center is known in the form of total length of the lines to be printed. By summing up them, total mass of the final geometry is determined. Moreover, center coordinates of the final geometry are also obtained. Thus, all the gravitational properties are obtained so far.

The next step is to obtain the center coordinates of the mass to be added. After this process, all the necessary information to build an additional shape to shift the center of input geometry will be found. To find the coordinates, weighted sum of the coordinates is used. The main idea is that the difference between weighted sum of the coordinates of the final geometry and the initial geometry is due to the effect of additional mass and its regarding position. Equations below are followed to obtain the center coordinates of the added mass in x, y and z-axis.

$$(\mathrm{M}_{\mathrm{Final}} \times X_{\mathrm{Final}}) - (M_{\mathrm{initial}} \times X_{\mathrm{initial}}) = M_{\mathrm{Added}} \times X_{\mathrm{Added}} \quad (1)$$

$$(\mathrm{M}_{\mathrm{Final}} \times Y_{\mathrm{Final}}) - (M_{\mathrm{initial}} \times Y_{\mathrm{initial}}) = M_{\mathrm{Added}} \times Y_{\mathrm{Added}} \quad (2)$$

$$(\mathrm{M}_{\mathrm{Final}} \times Z_{\mathrm{Final}}) - (M_{\mathrm{initial}} \times Z_{\mathrm{initial}}) = M_{\mathrm{Added}} \times Z_{\mathrm{Added}} \quad (3)$$

The only unknown in the above equations are the x, y and z coordinates. Thus, the center coordinates are obtained with the following equation.

$$C_{AddMass} = (X_{\mathrm{Added}}, Y_{\mathrm{Added}}, Z_{\mathrm{Added}}) \quad (4)$$

Now that the amount of mass to be added and the center locations of the regarding mass is known, additional geometry can be created and placed inside the initial geometry. This geometry is created as a uniform box of 100% fill rate so there is no space between the lines and a homogeneous box of added mass is placed at the center. In order to have a full fill rate, 0.27 mm gap is left between the lines at each layer. Considering the nozzle diameter of 0.4 mm is used, 0.27 mm gap is required and sufficient to slightly overlap the lines to have a better grip and completely fill the box. As in the slicing of the overall geometry, 60 µm of layer thickness is applied to the box and all respective calculations and distributions of the lines are completed with this layer thickness. After distributing the additional mass, the CoG of the final geometry is shifted to the desired location.

Since the created box of the additional mass is independent of the initial geometry, it may extend beyond the boundary of the initial geometry. To eliminate this undesired situation up to a certain limit, adjustment tools for changing the proportions of the box in x, y or z dimensions are developed. By adjusting the dimensions of the box, the same volume can fit into a box with a larger height and a smaller cross section or a lower height and a larger cross section. By this adjustment, extensions from the boundary surface can be eliminated to a level.

Although the adjustment tools try to eliminate the extension from the boundary of the geometry, they may not be sufficient to completely prevent those kind of undesired conditions. Then, the designer should go back to the input phase and choose a different amount of additional material. While using more material than before, the weighted effect of the additional mass will be more and smaller deviation from the initial center coordinates will be enough to shift the center of final geometry to the desired position. However, one should note that larger additional mass means that there will be a larger uniform box around center of new mass. Thus, using a larger amount of extra material may cause the box to extend further. If the designer could not find a sweet spot by adjusting the added mass parameters like mass and proportions of the box, one should also consider changing the desired location of the center of the final geometry. This is because of the physical limitations that the geometry may have and no matter the amount material is added, desired center may not exist in real life applications.

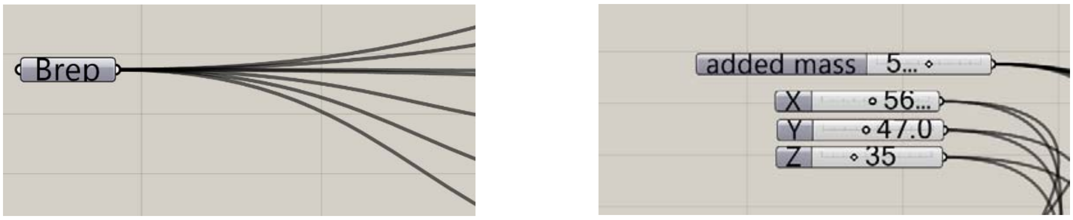
After finalizing the design processes mentioned above, the final geometry must be converted into G-codes to be used for 3D printing. Since the main principle of the algorithm is to create a non-homogeneous interior structure, conventional CAM software cannot be utilized for our method. STL files which are used as a de-facto standard in 3D printing can only have information about the outer shape of the geometry. Thus, they cannot contain information of the voxels and the additional mass inside the geometry. For this reason, a direct slicing method is followed and using a Python script, G-codes of the final geometry including the interior structure is obtained. Note that, the G-code is machine dependent and for different machines little adjustment needs to be made on the script.

2.3. Implementation

The platforms that the algorithm runs are Rhinoceros3D and Grasshopper3D (which is an add-on of Rhinoceros3D). In this section, the implementation of the algorithm utilizing the mentioned software is explained to have a better understanding of the algorithm.

Firstly, input geometry must be imported to Rhinoceros3D or modelled in there. If the geometry is imported, then it should be converted to a boundary representation (*Brep* in Rhinoceros3D and Grasshopper3D). Other important inputs can be defined via *Number* Slider blocks in Grasshopper3D as numerical inputs (Fig. 2).

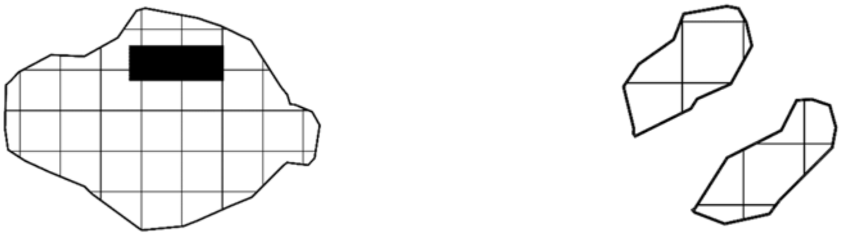
Grasshopper3D uses component blocks and connections to run the algorithms, which is an easier way to implement such algorithms. Obtained geometrical data in Grasshopper3D is presented in Rhinoceros3D. This helps us see all the layers (Fig. 3) of the artifact to be printed. If there are no awkward layers, then the G-codes can be generated and conveyed to the 3D printer via different means. Preview of the components can be deactivated since it increases the computational work of the processor and makes it harder to update the parameters when the algorithm is running.


Brep

added mass 5...
X 56...
Y 47.0
Z 35

(a) (b)

Fig. 2. Input components in the interface of the Grasshopper3D (a) Boundary representation (*Brep*); (b) *Number slider* components for center coordinates and additional mass



(a) (b)

Fig. 3. Cross sectional image of the model after voxelization and addition of mass (a) Body with additional mass; (b) Ears of the model *Bunny*

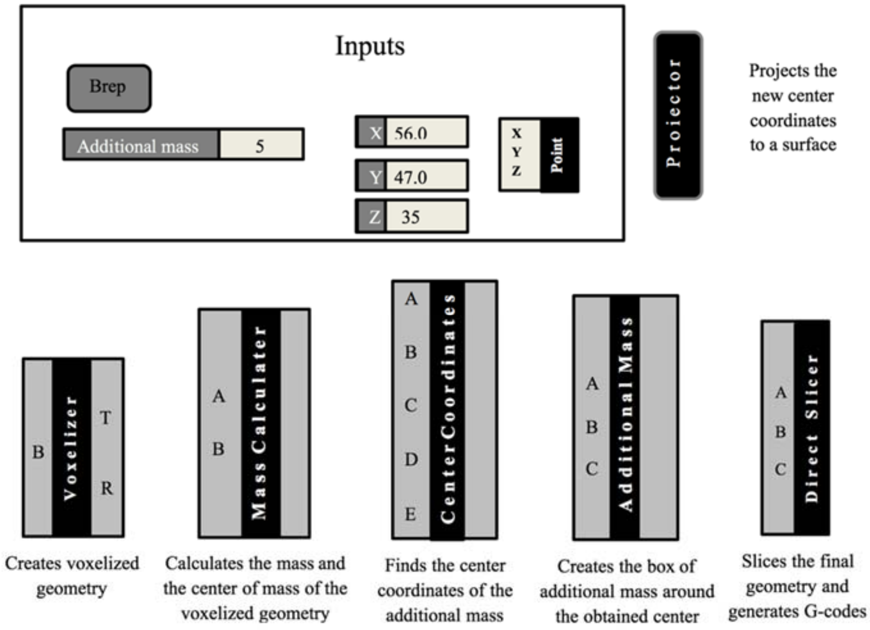

Inputs
Brep
Additional mass
5
X 56.0
Y 47.0
Z 35
X
Y
Z
Point
Projector
Projects the new center coordinates to a surface
B
Voxelizer
T
R
A
B
Mass Calculater
A
B
C
D
E
CenterCoordinates
A
B
C
Additional Mass
A
B
C
Direct Slicer
Creates voxelized geometry
Calculates the mass and the center of mass of the voxelized geometry
Finds the center coordinates of the additional mass
Creates the box of additional mass around the obtained center
Slices the final geometry and generates G-codes

Fig. 4. Schematic representation of the general algorithm in Grasshopper3D

Simplified version of the whole algorithm as a network of clusters can be seen in Fig. 4. Note that this image is a schematic representation of the cluster components in the Grasshopper3D interface, rather than the actual image.

2.4. Slicing and fabrication

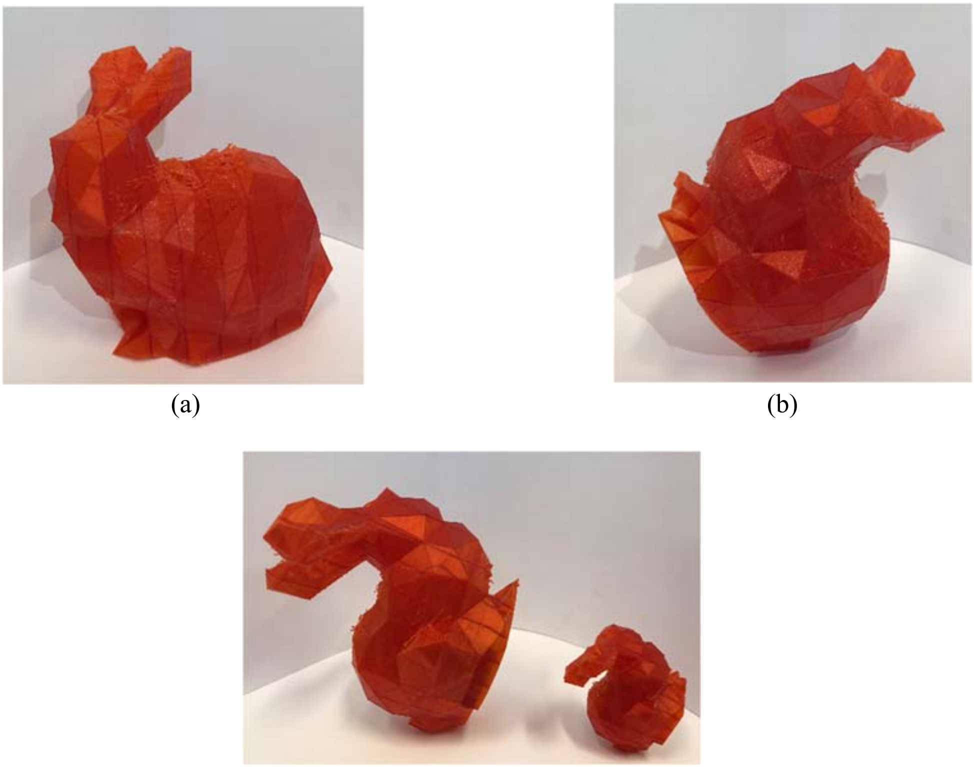
After the part is designed, it should be sliced and G-codes for fabrications should be generated. Since the interior of the artifact has become inhomogeneous in the design phase, conventional way of slicing has also become invalid for this job since the de facto standard STL files can only store the exterior information. Therefore, by utilizing a Python module inside Grasshopper3D, heterogeneous model can be sliced and the G-codes are directly generated inside the module by the line-wise approach. Inside the Python module, initial and final G-codes are taken from a standard CAM software and pasted on the G-code file that is being generated by the module. Note that, the machine specific codes should be carefully embedded inside the Python module. Thus, the generated G-code can be properly utilized in the machine selected. Even though the G-codes are properly generated, in practice it is quite difficult to obtain a standard printing quality.

3. Test case

In order to prove the validity of the solution, a test case is prepared where the famous *Stanford Bunny* model is tried to be manipulated in terms of gravitational properties. Note that the model is chosen to be a low-poly version of the *Bunny* since it is possible for it to stand on the surfaces along its body once the center is adjusted. Target CoG is set so the model can stand on its surface on the tail. In order to do this, CoG should shift 5.92% in x direction, 8.42% in y direction, 10.06% in z direction.

There are two models with different sizes to demonstrate the effect of size on the printing quality of the generated commands. It is expected to observe more overhanging, disturbed regions in the larger model with larger voxels. Since in the model with smaller voxels bridging occurs denser than the larger model, disturbances occurs more lightly.

Although the electronic model was successfully designed by following the steps mentioned earlier, obtaining the printed version was not quite a straightforward task. Because of the direct slicing method utilized in the solution,



(a) (b)

(c)

Fig. 5. (a) Printed bunny standing on the regular surface; (b) Bunny is standing on the tail surface as the goal of the test (c) Larger and smaller bunny models are standing on the tail surface

generating the proper G-code was tried in a somewhat iterative fashion so that it can fit the specific format of the machine that is used in this operation, which is Ultimaker^{2}GO. Fig. 5 shows the printed models and the validity of the assumption that the Bunny would stand on its tail as stated in the beginning of the test case. The overall goal is accomplished and the algorithm is proven to be a valid solution.

Even after generating the proper G-code, one can clearly see the defects on the surface of the model. Those defects are occurred mainly because those surfaces do not have supporting structures below them. Thus, it is inevitable for them to dangle from the edges. This was the original reason why voxels inside the model were created in the first place, which is to have a base structure for the initial model and the mass to be added after the gravitational adjustment. However, the test case clearly shows that there is still room for improvement on the manufacturing part of the solution.

4. Conclusion and discussions

In this paper, a novel approach is proposed to adjust the center of gravity of the 3D printed artifacts. Validity of the algorithm is tested and found to be agreeable with the goals of the algorithm.

- No design changes are made in the outer shape of the model.
- No extra type of material is utilized.
- Required shift in the CoG is provided.

Although we have accomplished the goals of the study, we need to further improve the approach. The first thing we should focus is to increase the quality of printing. The current direct slicing algorithm will be improved so that the printer can be adjusted more accurately. This software improvement will optimize the printer trajectory and deposition of the material, optimizing the overall sharpness and accuracy of the printed artifacts. Alongside the software optimization, design criteria should also be checked to improve the quality of the prints. As in the case of the test model, overhanging surfaces should be taken into consideration and the proper grit size for the voxels inside the model must be selected at the beginning of the process. Another thing to be improved in the future is the placement of the additional mass. Currently, a 3D box with rectangular cross section can be added inside the model. However, in all cases the complexity of the model requires more complex dimensions of the additional mass to be placed inside. This process is aimed to be automated and built inside the main algorithm. Thus, the designer can easily accomplish the tuning of the model.

The method proposed in the paper may be employed on static and dynamic applications. Static applications include basically any structural element for which the CoG can be altered in order to have a more stable stance or even stand on special surfaces which were not possible to stand upon before. In civil engineering, especially with the arriving of additive concrete manufacturing, walls, columns, bridges, all hanging features etc. can be modified such that they can be safely and stably stand on their intended surfaces in extreme cases.

Dynamic applications include moving (especially rotating) parts whose CoG and moment of inertia interfere with the resulting forces and moments on those parts. These forces can create distortions for the rotational balance of the dynamic parts. Fine tuning the rotational unbalance by adjusting the CoG can solve many problems in industries such as automotive and aerospace. In the following years, this technique that helps adjusting the CoG without making any alterations on the outer dimensions of the pieces will certainly have a major impact on industries that seek for an out of the box solution with the help of using original designs.

References

[1] Stava, O., Vanek, J., Benes, B., Carr, N., & Měch, R. (2012). Stress relief. ACM Transactions on Graphics, 31(4), 1-11.

[2] Zhou, Y., Kalogerakis, E., Wang, R., & Grosse, I. R. (2016). Direct shape optimization for strengthening 3D printable objects. Computer Graphics Forum, 35(7), 333-342.

[3] Vanek, J., Galicia, J. A., Benes, B., Měch, R., Carr, N., Stava, O., & Miller, G. S. (2014). PackMerger: A 3D Print Volume Optimizer. Computer Graphics Forum, 33(6), 322-332.

[4] Martínez, J., Dumas, J., & Lefebvre, S. (2016). Procedural voronoi foams for additive manufacturing. ACM Transactions on Graphics, 35(4), 1-12.

[5] Wu, J., Kramer, L., & Westermann, R. (2016). Shape interior modeling and mass property optimization using ray-reps. Computers & Graphics, 58, 66-72.

[6] Ying, J., Lu, L., Tian, L., Yan, X., & Chen, B. (2018). Anisotropic porous structure modeling for 3D printed objects. Computers & Graphics, 70, 157-164.

[7] Prévost, R., Whiting, E., Lefebvre, S., & Sorkine-Hornung, O. (2013). Make it stand. ACM Transactions on Graphics, 32(4), 1.

[8] Yamanaka, D., Suzuki, H., & Ohtake, Y. (2014). Density aware shape modeling to control mass properties of 3D printed objects. SIGGRAPH Asia 2014 Technical Briefs on - SIGGRAPH ASIA 14.

[9] Li, D., Dai, N., Jiang, X., & Chen, X. (2015). Interior structural optimization based on the density-variable shape modeling of 3D printed objects. The International Journal of Advanced Manufacturing Technology, 83(9-12), 1627-1635.

[10] Christiansen, A. N., Schmidt, R., & Bærentzen, J. A. (2015). Automatic balancing of 3D models. Computer-Aided Design, 58, 236-241.

[11] Prévost, R., Bächer, M., Jarosz, W., & Sorkine-Hornung, O. (2016, October). Balancing 3D Models with Movable Masses. In *VMV*.

[12] Yaman, U., Butt, N., Sacks, E., & Hoffmann, C. (2016). Slice coherence in a query-based architecture for 3D heterogeneous printing. Computer-Aided Design, 75-76, 27-38.