

## A STANDARDIZED FRAMEWORK FOR COMMUNICATING AND MODELLING PARAMETRICALLY DEFINED MESOSTRUCTURE PATTERNS

G.T. Williams\*, J. F. O'Brien†, A. Mezghani\*, M. J. Eachus\*, N. A. Meisel‡, C. C. McComb‡

\*Department of Mechanical Engineering, †Department of Materials Science and Engineering,  
‡School of Engineering Design, Technology, and Professional Programs

The Pennsylvania State University,  
University Park, PA 16802

### Abstract

Intricate mesostructures in additive manufacturing (AM) designs can offer enhanced strength-to-weight performance. However, complex mesostructures can also hinder designers, often resulting in unpalatably large digital files that are difficult to modify. Similarly, existing methods for defining and representing complex mesostructures are highly variable, which further increases the challenge in realizing such structures for AM. To address these gaps, we propose a standardized framework for designing and representing mesostructured components tailored to AM. Our method uses a parametric language to describe complex patterns, defined by a combination of macrostructural, mesostructural, and vector field information. We show how various mesostructures, ranging from simple rectilinear patterns to complex, vector field-driven cellular cutouts can be represented using few parameters (unit cell dimensions, orientation, and spacing). Our proposed framework has the potential to significantly reduce file size, while its extensible nature enables it to be expanded in the future.

### 1. Introduction

Additive manufacturing (AM) has fundamentally altered the landscape of manufacturable geometries by giving designers powerful abilities to manipulate shape, functional, material, and hierarchical complexity of their designs [1,2]. Respectively, these abilities allow for unique AM-enabled design advantages, including components with strategically-defined macro, meso, and micro structures [3]. In general, a macrostructure describes the overall shape of a geometry [4]. Mesostructures and microstructures refer to medium and smaller scale geometric features, respectively [4]. Designing with hierarchical complexity in mind, which involves the strategic definition of structures across scales, is becoming increasingly useful due to advancements in manufacturing capability, especially in examples such as lightweighting parts to achieve higher strength-to-weight ratio [5]. These mesostructurally-complex designs can also be useful for heat exchangers, thermal insulators [6], and metamaterial parts with controlled properties [7]. The diversity of different mesostructures that can be tried for even trivial applications is expansive [4,8], causing designers to rethink their traditional, macrostructurally-centered design processes.

Although the benefits of adding hierarchical complexity through the use of mesostructures are well-established, a major shortcoming exists in the AM community that leaves it unprepared to extract the full potential of such designs. Simply put, computer aided design (CAD) packages have

not kept pace with capabilities of manufacturing processes [9]. Since many industries evolved to predominantly use CAD only as a drafting tool to reproduce designs that were once completed by hand [10], commercial CAD tools excel at standard modeling tasks. Unfortunately, they often fail to provide efficient and scalable techniques to design, save, communicate, and redesign the diversity of mesostructurally complex designs that researchers have explored. Standard file formats superlinearly scale in size when the complexity of a mesostructure increases, limiting designers [11]. Furthermore, although many CAD standards exist [12], such as ISO 10303 [13] and ISO/ASTM 52915 [14], none sufficiently provide a unified framework to allow designers to share hierarchically complex designs in a platform-agnostic manner.

As the understanding of hierarchically complex parts matures, further work must be completed to bring those pieces together into useful end-user tools and systems. In this study, we propose and demonstrate a standardized framework for representing mesostructurally complex parts. The framework uses hierarchically-complex entities and scalable relationships to provide generalized, platform-independent representation of macrostructures, mesostructural unit cells, coordinate systems, and patterns. Our framework both meets the demands of mesostructure strategies available today, such as truss-based, cellular, and functionally-graded patterns, and allows for expansion to meet future needs. We showcase the versatility of this framework by modeling six different case studies and characterizing our framework's effect on the file size scalability. Through this study, we aim to answer the following research questions:

1. What is the capacity of a single framework to describe the diverse types of mesostructures previously explored in literature and beyond?
2. If using such a framework, to what extent does doing so influence file sizes when compared to STEP and STL formats?

For our first research question, we hypothesized that our framework will be able represent diverse categories of mesostructures, which must be shown through rigorous case study testing. For our second research question, we hypothesized that the framework would reduce file sizes for all case studies and that the benefit will be maximized with the most complex mesostructures because the existing CAD formats must duplicate geometry definition for every patterned unit cell instance.

## **2. Background**

### *2.1. Mesostructural Complexity in Additive Manufacturing*

Mesostructures are made up of many unit cells and can either be stochastic, as is the case in foams [15]; unique, if each unit cell is different from the others; or patterned, as is most common with AM parts [4]. Three-dimensional, patterned mesostructured parts are difficult or impossible to manufacture using traditional methods, but can be designed to exhibit unique mechanical properties [16–18]. Two of the most common approaches for designing parts with patterned mesostructures are truss-based and cellular-based patterns [19]. Truss-based unit cells define repeating truss elements which connect nodes [4]. Cellular-based unit cells pattern a shape and subtract it from the macrostructure [19]. While regular unit cells are common in AM, the design freedom provided by the technology allows the manufacture of non-uniform mesostructural elements. A clear example of this is the creation of a graded lattice structure, with uniform variation

in member thickness [20]. However, these lattice structures are often laborious to create, or created using proprietary tools.

Although proprietary CAD applications such as SolidWorks, NetFabb, and nTopology can assist the creation of complex lattice structures, they introduce challenges of their own. For instance, CAD tools do not follow uniform standards or workflows [21], which means that advancements in one tool will only immediately enhance the capabilities of its specific users. Furthermore, users typically must convert CAD files to neutral formats before switching proprietary applications or advancing through the design-to-manufacture process [22]. While this conversion, although tedious, does not normally cause issue, neutral CAD formats are poorly suited for preserving fine details and design intent [23], two attributes that are often essential to mesostructurally-complex designs. Similarly, the available features, such as unit cell type and patterning methodology, differs in each proprietary application, hindering engineers' collaborative efforts. A diverse array of mesostructure synthesis techniques have been developed by researchers, sometimes using third-party software such as MATLAB or C++ [24–26]. This trend suggests that maximizing interoperability of mesostructure data representation and design workflows rather than relying on available proprietary methods would help foster advancement across the industry.

## 2.2. Additive Manufacturing Digital File Formats

Despite the benefits designers have at their disposal when manufacturing a mesostructured part using AM, they often face challenges when navigating the digital thread of file formats that is inherent to AM processes [27]. For the purposes of this study, we have organized the filetypes employed along the digital thread into four groups: input, design, transfer, and machine (see Figure 1).

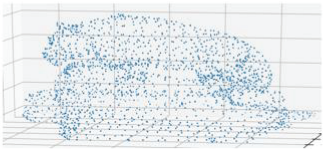


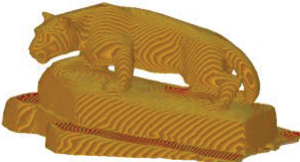
	Example File Formats	Example Image		Example File Formats	Example Image
INPUT	.OBJ .DICOM .BMP		TRANSFER	.STL .AMF .3MF	
DESIGN	.STEP .IGES .SLDPRT .F3D		MACHINE	G-code	

Figure 1 Listing of the groups of CAD files used in the AM digital thread, as they pertain to this study, with example formats and images.

“Input” files are the results of scanning operations used to bring objects from the physical world into the digital world, and include surface point cloud file formats [28], such as OBJ, voxelized representations [29], DICOM, and photogrammetry image sets [30]. “Design” files are CAD files which are typically characterized by dimensional accuracy, describing geometries as mathematically defined boundary surface representations. Generally, they include neutral data interchange files, such as STEP [13] and IGES [31], and proprietary CAD files, such as Autodesk Fusion 360 Archive Files (.f3d) and SolidWorks Part Files (.SLDPRT). “Design” files can also

contain abstract information, such as parameterized modeling schemes, allowing designers to quickly redesign parts by only modifying a few values [32]. “Transfer” files are used to convert “input” or “design” files into a consistent form prior to translation into layer slices and machine commands [33]. “Transfer” files are usually a repetition of a simple tessellation pattern, such as the Stereolithography (STL), which patterns triangles across the entire surface [33]. Despite being computationally lightweight to process, “transfer” files can require very large file sizes, especially for curved or repetitive geometry. Additionally, these files can vary based on the CAD packages used, and aren't produced in a uniform way [34]. We included STEP and IGES files, which are commonly referred to as “interchange” formats in the “design” category instead of the “transfer” category because one of their primary purposes is to interchange between other CAD file formats, and not to transfer geometric information to slices. Finally, “machine” files are the actual machine position and temperature commands, and are often text commands in the form of G-code [35].

Despite the prevalence of the STL file format in the AM workflow [32], STL lacks several features that can be crucial to AM geometries. For instance, tessellated polygons patterned in an STL cannot perfectly model curved surfaces [36]. Another example is the poor scalability of STL file sizes with increases in geometric complexity [37]. In response to these and other shortcomings, newer AM file formats have been developed [12]. Two examples are the AMF and 3MF formats. The AMF file format, ISO/ASTM 52915, preserves the tessellated surface geometry data from the STL file format and adds additional features through an extensible system [14]. 3MF was developed by Microsoft as a standardized AM data format [38], and uses a similar XML scheme to AMF [14]. Although not an ISO standard, 3MF is backed by an industry-sponsored consortium [38]. While these extensible AM file formats offer potential solutions to many issues faced by modern AM designers, they do not fully capitalize on the complexity AM has to offer. For instance, they are lacking in representation of voxel-based geometries, tolerancing and testing specification, and efficient representation of generalized mesostructures. We predict that as design tools in these areas become more advanced, greater need for such capabilities will emerge.

In response to the gap in standard AM file format mesostructure representation, we propose a standardized framework for representing complex mesostructures in an efficient, scalable, and platform-agnostic manner. By defining the data structure for complex mesostructures outside of proprietary CAD applications, many different CAD applications can be upgraded to behave identically and predictably build parts defined by the data structure. This framework exists in the “design” domain of AM digital representations. As mentioned earlier, it does not intend replace other standard file formats, such as STEP or AMF, but rather to coexist with them and be used as a part of the digital thread when a design requires it.

### **3. Methodology**

The framework presented in this study is not a file format which can be used with CAD applications today, but rather a generalized blueprint upon which such a file format can be developed. This framework could then be standardized as a file format and the next generation of CAD software upgrades and add-ins could be designed to utilize it. We generate a prototype of such an implementation using the Fusion 360 API [39] and apply it to six case studies to demonstrate the framework's flexibility and usefulness in a practical setting. This section describes

1) our proposed framework, 2) our prototype CAD implementation of the framework, and 3) the case-studies to demonstrate our framework.

### 3.1. Proposed Mesostructure Framework Design

Our framework is inspired by the design strategy of first defining macrostructures and then applying mesostructural elements to them, which may be represented as parent-child relationships that may interact with each other (see Figure 2). The four operations handled by our framework are 1) coordinate system definition, 2) macrostructure definition, 3) mesostructure unit cell definition, and 4) unit cell patterning. These operations were selected with three target framework attributes in mind: generalizability, flexibility, and scalability. In order to be general, our framework must allow many potential design paradigms to be used. Allowing multiple types of macrostructure representations, custom coordinate systems, unit cells, and patterns enables this goal to be maximized. Flexibility is achieved by allowing the user to employ as few or as many of the framework's features as needed. For instance, if a custom coordinate system is not defined, the industry-standard cartesian coordinate system would be used. Lastly, the framework promotes scalability by allowing reuse of the four operation definitions wherever possible. Scalable designs must preserve memory-efficiency with increased complexity, and can be achieved through parametric definitions and abstract data concepts, such as mathematically-defined patterns and gradients.

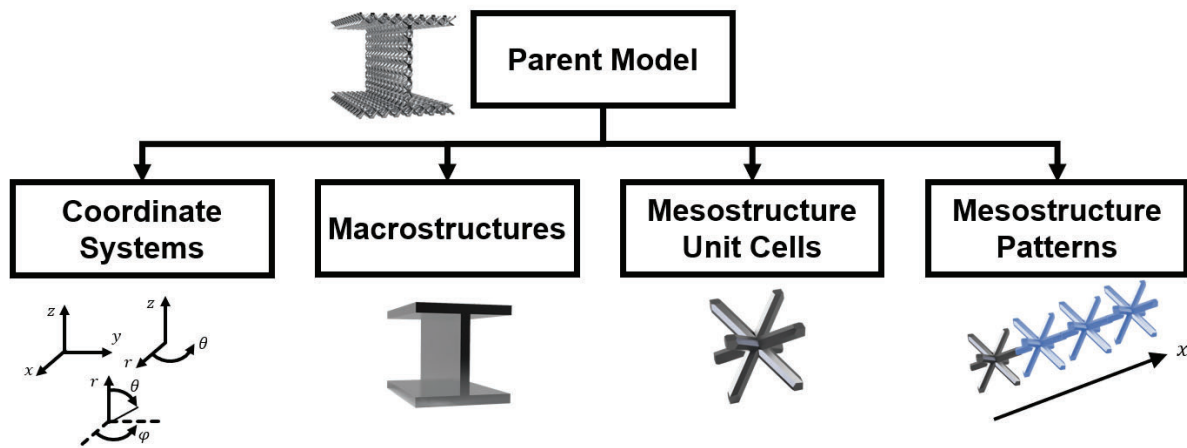


Figure 2 Hierarchical relationship between some of the information required to pattern a unit cell across a macrostructure

We have created an entity-relationship model as a generalized description of the framework from which programmatic implementations can be built (see Figure 3). This entity-relationship model defines the relationship between our major operations. The arrows extending outward from the parent model indicate which entities are to have many-to-one relationship allowance, a key differentiating feature for our framework from other file formats, such as AMF, which uses a “constellation” feature that merely patterns a single object to specified positions in cartesian space [14]. Conversely, our framework enables scalable inclusion of parametric design in all of its major operations. For example, unit cells can be defined by parametric values using constructive solid geometry (CSG) operations. These values can then be scaled, stretched, and morphed through space by changing their parameter values using functions defined as unit cell patterning entities. This scheme allows each individual geometric definition to determine many more complex patterns than simple patterning of a single shape by employing more abstraction in the file format.

Although doing so may increase the complexity of the parsing program which reads and manipulates the files, we believe this is a necessary trade-off for advancing the art in the next generation design applications.

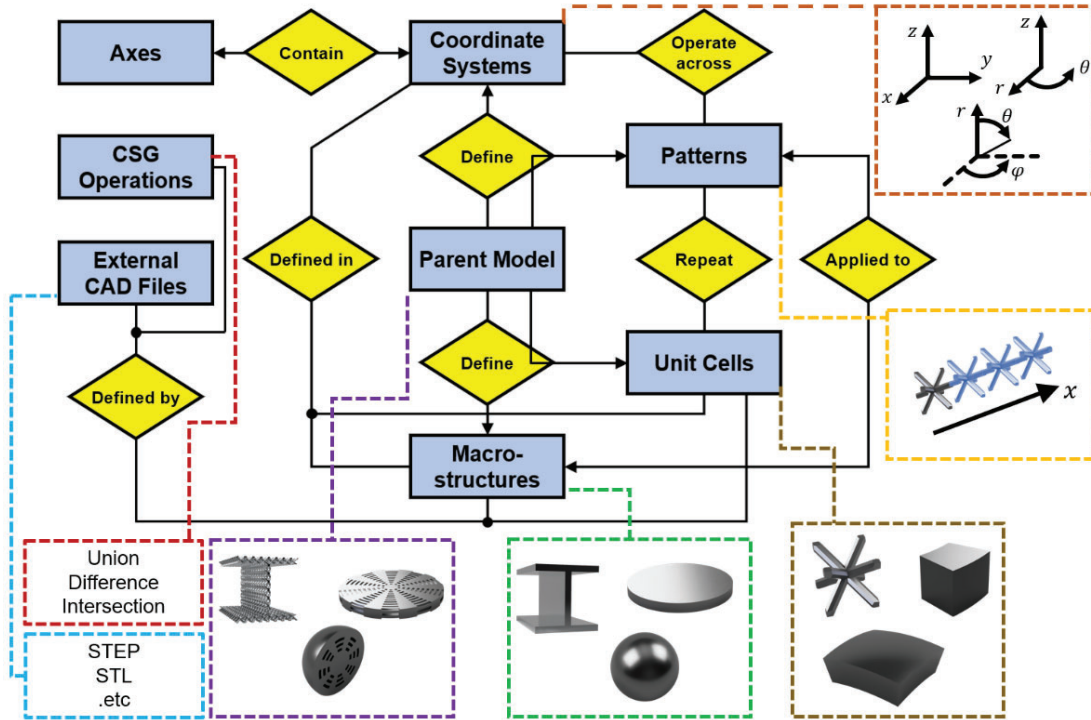


Figure 3 Entity-relationship diagram for the proposed mesostructure framework with examples of each entity.

### 3.1.1. Coordinate System Definition

Our framework augments the global cartesian coordinate systems used by CAD applications by allowing users to define their own coordinate systems in one, two, three, or even more dimensions. More than three dimensions could be useful in designs that specify conditions based on both spatial and other physical quantities, such as time and or density. Additionally, these coordinate systems can have nonlinear axes, as is the case in polar and spherical coordinate systems. Allowing for different coordinate system representations can simplify the amount of information needed to represent some geometries and facilitate new ways of thinking about patterns beyond rudimentary rectilinear and circular templates. We handle these complex coordinate systems by allowing users to specify the origin, unit vector direction, and axis type (linear, distance, or circular) for each axis of the coordinate system in the global cartesian space. Linear axes record position as a distance magnitude to the origin and do not change direction. Distance axes record position as a distance vector to the origin that can change direction based on another coordinate axes. Circular axes record position as an angular distance from an origin angle.

The coordinate system examples we explore in this initial study are cartesian, polar, and spherical (see Figure 4). For custom 3D cartesian coordinate systems, three perpendicular, linear unit vectors are defined. Although seemingly trivial, this feature allows users to locate and rotate structures relative to one another within the CAD global default coordinate system in a platform-agnostic manner. For 3D polar, a distance axis ( $r$ ), a circular axis ( $\theta$ ), and a linear axis ( $z$ ) can be

used. Finally, a 3D spherical coordinate system uses a distance axis ( $r$ ) and two circular axes ( $\theta$  and  $\varphi$ ).

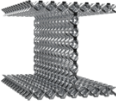
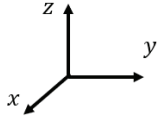

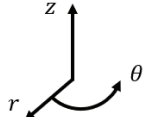

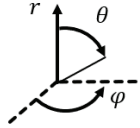
Example Case Study	Axes Diagram	Axes Types
CARTESIAN 		$x$ : Linear $y$ : Linear $z$ : Linear
POLAR 		$r$ : Distance $\theta$ : Circular $z$ : Linear
SPHERICAL 		$r$ : Distance $\theta$ : Circular $\varphi$ : Circular

Figure 4 The three coordinate systems investigated in this study

### 3.1.2. Macrostructure Definition

Once coordinate systems are defined, macrostructure entities may be created, positioned, and oriented. Since our file format does not seek to replace existing geometric definition standards, we chose to use a flexible approach to how macrostructures may be defined. In our approach they can either be directly created through CSG or imported as a reference to an external standard CAD format file. CSG was chosen for direct model definition because it is supported by many CAD applications and allows for many AM-relevant geometries to be defined [40]. Once the macrostructure is defined, it must be positioned and oriented within the global coordinate system of the CAD application attempting to read it. Our framework enables these actions by allowing transformation matrices to be applied to macrostructures. To support scalability, more than one macrostructure may be added to each file, allowing assemblies or arranged build platforms to be represented.

### 3.1.3. Mesostructure Unit Cell Definition

After defining macrostructures, the mesostructure unit cell entities may be defined and applied to them. Like macrostructures, mesostructure unit cells can either be directly defined through CSG or imported from an existing model. The mesostructure unit cells are defined once and then patterned many times through separate patterning entities. Unlike macrostructures, mesostructures can be defined either by constant values or based on parameters that can be changed through functions. When defining the mesostructure, the file specifies these parameters with unique identifiers. For instance, the thickness of a truss element could be parametrically defined, and its actual size values would be computed as a result of a pattern entity's operations later in the file. At this time, imported geometries used as mesostructures may be parametrically morphed through transformation matrices, with the elements of those transformation matrices serving as the defining parameters.

### 3.1.4. Mesostructure Pattern Definition

Once mesostructure entities are ready for use, pattern entities can be used to duplicate the unit cells across the macrostructures. The actual application of the mesostructures is achieved through

CSG. Unit cells can be subtracted from the macrostructure using a “difference” operation, added to the macrostructure using a “union” operation, and combined with the macrostructure using an “intersection” operation. This approach means that hierarchically complex parts can be defined in two divergent ways: Either as patterned mesostructures alone, or as a combination of both macrostructural and mesostructural components. This dual approach is flexible and mimics paradigms that designers are familiar with in current CAD applications. Once the method of application is prescribed, the unit cells are then patterned through looping commands. The number, spacing, and direction of the patterns may be defined as constant values or mathematical functions (see Figure 5). As shown in Figure 5a, unit cells are designed just like any other macrostructure, a parametric ellipsoid in this case. This unit cell shape can then be oriented or morphed based on additional parameters. Additionally, the initial values and functional gradients of unit cell parameter values are defined alongside the ellipsoid as well, represented as numeric attributes of the unit cell entity definition. Doing so allows a CAD program to rebuild the geometry exactly as intended by the designer while maintaining the freedom to make major design changes with small changes in the parametric equations.

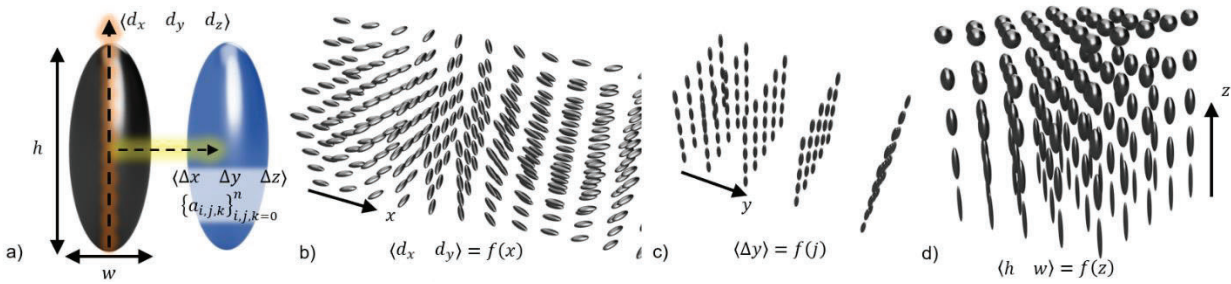


Figure 5 Images of a) an ellipsoidal unit cell positive form, b) functionally grading the orientation of the ellipsoid as a function of position, c) functionally grading the spacing of the ellipsoid as a function of iteration number in a series, and d) functionally grading the shape of the ellipsoid as a function of space

In addition to patterning based on functions, our framework also allows for exotic unit cell placement and spacing. For example, rather than to simply pattern unit cell position and parametric size blindly across a coordinate system, a different design strategy could be to define those unit cells based on a vector field. Our framework allows vector fields to be imported from an external file or embedded directly as binary values. First, the structure of the vector field file is defined. Next, the vector field is made available to the file as an array of IEEE double-precision floating point values that meets the organizational definition. Finally, the patterning instructions detail how each vector field value manipulates the parametric transformations of the unit cell. We have chosen this generalized method so that user may input large amounts of vector field information without reliance on a particular proprietary format for that information. This technique could be used to apply the results of numerical analysis to create unique patterns or vectors based on the curvature of a part to create conformal lattice structures.

### 3.2. Prototype File Format and CAD Implementation

We have implemented a prototype file format and CAD design tool that demonstrates this framework in action. We have chosen an Extensible Markup Language (XML) file format that supports the features in our entity-relation model. We chose XML because it is human-readable and used by other modern AM-oriented file formats, such as AMF and 3MF, and extensible [37]. This file format is not intended to be a finished product ready for deployment to commercial CAD applications as-is, but rather a viable demonstration to support our research questions. Future file

formats could either be an extension of this effort or a restarted effort adopting a different strategy, such as a binary file specification. For the prototype CAD implementation, we created a Fusion 360 Add-In. This Add-In was programmed in C++ using the Fusion 360 API. To showcase the benefits of our framework to the design process, we enabled our Add-In to use a form-based dialog window, allowing users to specify their mesostructure through simple inputs, such as drop-downs, checkboxes, and number entry fields (see Figure 6).

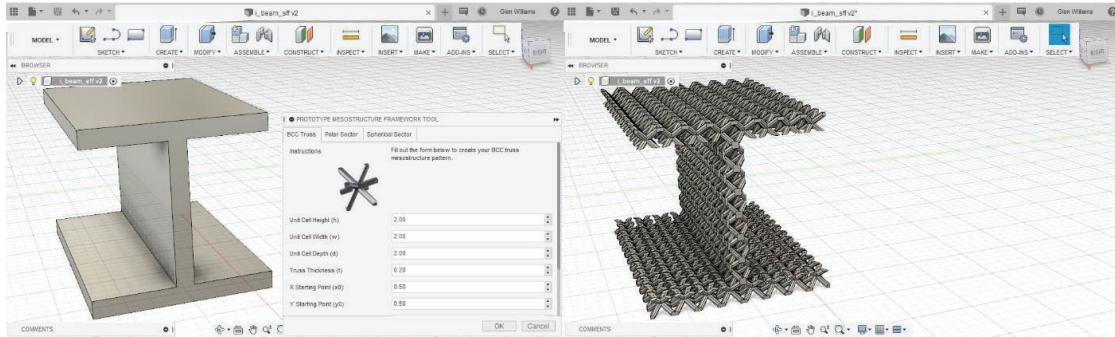


Figure 6 Screenshot of the prototype Fusion 360 CAD Add-In being used to create a BCC truss I-beam part

#### 4. Case Studies

After developing our framework and a prototype implementation we tested the framework with six diverse case-studies in order to examine the framework's versatility with respect to arbitrary, mesostructurally-complex designs (see Figure 7). We designed these case study geometries using our prototype framework implementation then compared the resulting file size to STEP and STL files, two common file formats used in the AM community.



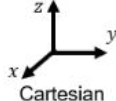
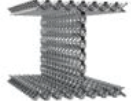


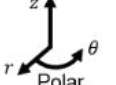



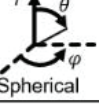


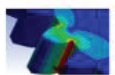

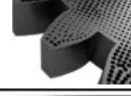

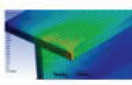

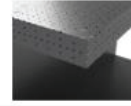
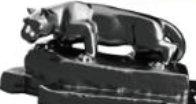

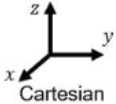

Case Study Name	Macro	Meso	Pattern Coordinate System	CSG Pattern	Applied Geometry
I-beam		 BCC Truss	 Cartesian	Intersection	
Disk		 Polar Sector	 Polar	Difference	
Sphere		 Spherical Sector	 Spherical	Difference	 *Section View
Gear	 	 Cylinder	$\langle x, y, z, \sigma \rangle \times n$ Mesh Node von Mises Stress Field	Difference	
Direction I-beam	 	 Ellipsoid	$\left\langle x, y, z, \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{bmatrix} \right\rangle \times n$ Mesh Node Principal Stress Vector Field	Difference	
Nittany Lion		 Nittany Lion	 Cartesian	Difference	

Figure 7 Images of macrostructures and mesostructures for all case studies investigated

These case-studies were intended to capture increasingly complex examples of 1) truss-based and cellular patterning mesostructures that are commonly used today (“common”) and 2) some potential design concepts that are not commonly encountered (“uncommon”). For the “common” examples we explored an I-beam with a truss-based unit cell (“I-beam”), a disk with a polar sector cutout pattern (“disk”), and a sphere with a spherical sector cutout pattern (“sphere”). For the “uncommon” examples we explored a gear with vector field-driven, varying size cutouts (“gear”), an I-beam with vector field-driven, varying direction cutouts (“direction beam”), and a patterned macrostructure used as a mesostructure in a Nittany Lion statue (“Nittany Lion”).

All of the “common” case-studies used base macrostructures defined by the STEP file format and mesostructures defined with CSG operations. The I-beam case study uses an I-beam-shaped macrostructure and a body-centered cubic (BCC), truss-based mesostructure. All entities in the I-beam were designed using a cartesian coordinate system. The parameters of the mesostructure included truss thickness, unit cell height, unit cell width, and unit cell depth. It was applied using a static pattern spacing in all three dimensions and intersection CSG operation with the macrostructure. Unlike the I-beam, the disk case study used a difference CSG operation, subtracting a sector-shaped volume element unit cell from a disk-shaped macrostructure. The sector was patterned along the three polar directions. The spherical case study was similar to the polar geometry but used a spherical macrostructure, spherical sector unit cell, and used one distance axis and two curved axes.

For the “uncommon” case studies, the base macrostructures were also defined by the STEP file format. In the gear case study, a basic involute gear was modeled and meshed for finite element analysis (FEA) using ANSYS. A force was then applied to a single tooth of the gear. The purpose of this analysis was not to accurately model the performance of a gear, but rather to generate an interesting scalar field to test the mesostructure framework with. We used the FEA locations and their corresponding von Mises stress magnitudes as input scalar field data to define the locations and sizes of patterned cylinder cutouts. The direction I-beam case study also used an FEA analysis. An asymmetric bending force was applied to the corner of the I-beam flange. In this case, the principal stress direction vectors were output from the analysis and used to control the position and orientation of constant-size, ellipsoid cutout features. Finally, the Nittany Lion case study demonstrated how both the macrostructure and the mesostructure can be loaded from external CAD files, in this case patterning a structure within itself along a cartesian coordinate system.

To evaluate the performance of the prototype file format and CAD implementation we analyzed the file sizes of each of the case studies. First, we modeled the macrostructure of each case study geometry and saved that file in the STEP format. Next, we used our prototype framework Fusion 360 CAD add-in to apply the appropriate mesostructure. We then saved this geometry as a prototype framework file, STEP file, and ASCII STL file. We then compared the size of these files to determine how much the file size was affected when using our framework. Additionally, we modeled the I-beam case study for different unit cell densities and similarly saved all files for size comparison to investigate the effect of the number of unit cells on the file size reduction benefits of our framework. For all case studies we set Fusion 360 to export the STL with the “High” quality setting in order to maximize fidelity to the original geometries. For the framework file size, the sizes of all source files needed for the prototype Fusion 360 add-in to build the part were summed. These files included macrostructure STEP files, XML framework files, and

FEA vector field data files. This summation was done to ensure that all data necessary to rebuild the framework file was accounted for since, unlike the STEP and STL files, some of the data existed in multiple files. Other implementations of our framework could choose to bundle all data into a single file.




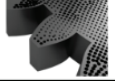
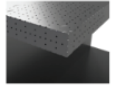

Case Study Name	Applied Geometry	Macro Structure	Mesostructured Part			% Reduction	
		STEP File Size (bytes)	STEP File Size (bytes)	ASCII STL File Size (bytes)	Framework File Size (bytes)	vs STEP	vs STL
I-beam		19,209	27,250,688	13,700,044	20,344	99.925	99.851
Disk		6,582	2,056,580	5,150,111	7,594	99.680	99.872
Sphere		4,619	935,422	53,564,192	5,280	99.378	99.989
Gear		535,036	1,560,926	111,224,295	710,259	54.498	99.361
Direction I-beam		19,209	5,243,525	136,158,680	73,422	98.600	99.946
Nittany Lion		4,718,339	235,916,953	11,104,200	4,719,345	98.000	57.499

Figure 8 Results of the file size comparison for each study, showing the percent reduction when using the framework

Our framework successfully reduced the file size compared to STEP and STL for all case studies (see Figure 8). Nearly all case studies reduced the file size by 98% or more, except for the gear STEP file and the Nittany Lion STL file cases. These cases were 54.498% and 57.499%, respectively. We suspect that the gear STEP file was not reduced as much as the STL because STEP is capable of some abstractions, such as representing circular edges, with short commands. Since the gear mesostructure incorporated cylindrical cutouts, a circular edged feature, it was thus able to be represented in STEP with less penalization compared to our framework. However, the STL, which cannot represent curved features without many triangles, still exhibited large file sizes for the gear. For the Nittany Lion, we surmise that the decreased benefit of the framework was due to the original method of modeling the Nittany Lion geometry and our method of comparison. The Nittany Lion was originally obtained as an STL that was produced from a digital scan of a statue. This STL was then converted to a STEP file. We suspect the Fusion 360 conversion operation introduced inefficient overhead because the macrostructure STEP file was significantly larger than the STL. Resaving the mesostructured model as an STL removed this overhead. Had we used the original STL for the framework file, the reduction would have been over 99% compared to STL.

To investigate our second research question, the influence of our framework on file size, we further investigated variants of the I-beam case study to study the effect of unit cell density on file size. By varying the number of unit cells in the mesostructure, we were able to more clearly identify a trend in our framework's effectiveness in relation to design complexity. We observed that the file size increased exponentially with respect to the number of unit cells (see Figure 9). The increased number of unit cells only negligibly altered the prototype framework file sizes.

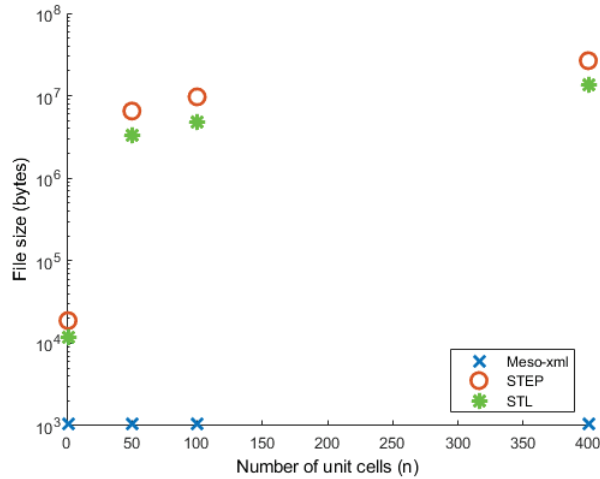


Figure 9 Plot of the I-beam case study file size versus the number of BCC unit cells used in the mesostructure

These results indicate that our framework successfully reduces file sizes for all case studies examined, offering real potential to improve efficiency in the design process for complex mesostructures. Based on our findings, file size reduction benefits of our framework over “design” CAD files, such as STEP, are maximized when there are few abstract features, such as circular edges, in the model. Additionally, file size reduction benefits over STL are maximized when the source geometry is not already an STL. In general, file size reduction was most dramatic when the number and complexity of unit cells was high.

## 5. Conclusions and Future Work

This study proposed and demonstrated a standardized framework for designing and representing mesostructured components tailored to AM. A prototype implementation of our framework, incorporating both an extensible file format based on our framework and a Fusion 360 Add-In design tool to manipulate the files, was generated to explore six different case studies. The case studies, which represented varied mesostructure strategies, were then analyzed for file size reduction compared to standard STEP and STL file formats.

In terms of our first research question, we believe our framework is capable of representing the diverse array of mesostructures found in literature. The framework’s ability to parametrically define truss-based and cellular mesostructures in terms of arbitrary coordinate systems give it a strong advantage over standard file formats alone, which do not retain parametric knowledge of the patterning scheme and are typically limited to cartesian coordinate systems only. Regarding our second research question, the framework resulted in smaller file sizes for all case studies observed. The file size does not scale with increases in the number of unit cells, giving a stronger advantage for increasingly complex models.

Although this initial study has presented the framework, more work still needs to be done before it may be deployed in CAD applications. A full file format should be created, documented, and rigorously tested. Additionally, using a framework like the one presented introduces abstractions into the digital thread, trading off data storage for computational cost. The computational cost of this trade off should be characterized to confirm how complex of

mesostructures can be handled by today's hardware. This study could be accompanied by a more in-depth look at how the number and complexity of unit cells affects these computational performance metrics. Furthermore, more improved definitions for how vector field data may be imported into a model should be developed, so as to maximize compatibility with software that may be used to generate such vector fields. Finally, end-use CAD Add-Ins should be developed for a variety of platforms to test how using the framework enhances the human experience when designing complex mesostructures and modifications or extensions should be made to maximize its benefit. Our prototype add-in was only capable of modeling geometries similar to the ones explored in this study, and must be expanded before being able to handle the full range of geometries found in literature and industry.

## **6. References**

- [1] Bloesch-Paidosh, A., and Shea, K., 2018, "Design Heuristics for Additive Manufacturing Validated Through a User Study," *J. Mech. Des.*, (c), pp. 1–40.
- [2] Gao, W., Zhang, Y., Ramanujan, D., Ramani, K., Chen, Y., Williams, C. B., Wang, C. C. L., Shin, Y. C., Zhang, S., and Zavattieri, P. D., 2015, "The Status, Challenges, and Future of Additive Manufacturing in Engineering," *Comput. Des.*, **69**, pp. 65–89.
- [3] Yang, S., and Zhao, Y. F., 2015, "Additive Manufacturing-Enabled Design Theory and Methodology: A Critical Review," *Int. J. Adv. Manuf. Technol.*, **80**(1–4), pp. 327–342.
- [4] Chu, C., Graf, G., and Rosen, D. W., 2008, "Design for Additive Manufacturing of Cellular Structures," *Comput. Aided. Des. Appl.*, **5**(5), pp. 686–696 Additive Manufacturing technologies enable.
- [5] Maskery, I., Sturm, L., Aremu, A. O., Panesar, A., Williams, C. B., Tuck, C. J., Wildman, R. D., Ashcroft, I. A., and Hague, R. J. M., 2018, "Insights into the Mechanical Properties of Several Triply Periodic Minimal Surface Lattice Structures Made by Polymer Additive Manufacturing," *Polymer (Guildf.)*, **152**, pp. 62–71.
- [6] Xiong, J., Mines, R., Ghosh, R., Vaziri, A., Ma, L., Ohrndorf, A., Christ, H. J., and Wu, L., 2015, "Advanced Micro-Lattice Materials," *Adv. Eng. Mater.*, **17**(9), pp. 1253–1264.
- [7] Xu, S., Shen, J., Zhou, S., Huang, X., and Xie, Y. M., 2016, "Design of Lattice Structures with Controlled Anisotropy," *Mater. Des.*, **93**, pp. 443–447.
- [8] Hussein, A., Hao, L., Yan, C., Everson, R., and Young, P., 2013, "Advanced Lattice Support Structures for Metal Additive Manufacturing," *J. Mater. Process. Technol.*, **213**(7), pp. 1019–1026.
- [9] Thompson, M. K., Moroni, G., Vaneker, T., Fadel, G., Campbell, R. I., Gibson, I., Bernard, A., Schulz, J., Graf, P., Ahuja, B., and Martina, F., 2016, "Design for Additive Manufacturing: Trends, Opportunities, Considerations, and Constraints," *CIRP Ann.*, **65**(2), pp. 737–760.
- [10] Monedero, J., 2000, "Parametric Design: A Review and Some Experiences," *Autom. Constr.*, **9**(4), pp. 369–377.
- [11] Zha, W., and Anand, S., 2015, "Geometric Approaches to Input File Modification for Part Quality Improvement in Additive Manufacturing," *J. Manuf. Process.*, **20**, pp. 465–477.
- [12] Lu, Y., Morris, K., and Frechette, S., 2016, *Current Standards Landscape for Smart Manufacturing Systems*.
- [13] Pratt, M. J., 2001, "Introduction to ISO 10303—the STEP Standard for Product Data Exchange," *J. Comput. Inf. Sci. Eng.*, **1**(1), p. 102.

- [14] Lipson, H., 2012, "Standard Specification for Additive Manufacturing File Format (AFM) Version 1.2," ASTM Int., **2013**, pp. 1–15.
- [15] Davies, G. J., and Zhen, S., 1983, "Metallic Foams: Their Production, Properties and Applications," *J. Mater. Sci.*, **18**(7), pp. 1899–1911.
- [16] Tancogne-Dejean, T., Diamantopoulou, M., Gorji, M. B., Bonatti, C., and Mohr, D., 2018, "3D Plate-Lattices: An Emerging Class of Low-Density Metamaterial Exhibiting Optimal Isotropic Stiffness," *Adv. Mater.*, **30**(45), p. 1803334.
- [17] Wang, Q., Jackson, J. A., Ge, Q., Hopkins, J. B., Spadaccini, C. M., and Fang, N. X., 2016, "Lightweight Mechanical Metamaterials with Tunable Negative Thermal Expansion," *Phys. Rev. Lett.*, **117**(17), pp. 1–6.
- [18] Pa, P., Mirotznik, M. S., McCauley, R., Yarlagadda, S., and Duncan, K., 2012, "Integrating Metamaterials within a Structural Composite Using Additive Manufacturing Methods," *Proceedings of the 2012 IEEE International Symposium on Antennas and Propagation*, IEEE, pp. 1–2.
- [19] Evans, A. G., Hutchinson, J. W., Fleck, N. A., Ashby, M. F., and Wadley, H. N. G., 2001, "The Topological Design of Multifunctional Cellular Metals," *Prog. Mater. Sci.*, **46**(3–4), pp. 309–327.
- [20] Hussey, A., Hague, R., Tuck, C., Panesar, A., Maskery, I., Ashcroft, I., and Aremu, A., 2016, "An Investigation into Reinforced and Functionally Graded Lattice Structures," *J. Cell. Plast.*, **53**(2), pp. 151–165.
- [21] Rappoport, A., 2003, "An Architecture for Universal CAD Data Exchange," *Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications - SM '03*, ACM Press, New York, New York, USA, p. 266.
- [22] Fiorentino, M., Radkowski, R., Stritzke, C., Uva, A. E., and Monno, G., 2013, "Design Review of CAD Assemblies Using Bimanual Natural Interface," *Int. J. Interact. Des. Manuf.*, **7**(4), pp. 249–260.
- [23] Bianconi, F., Conti, P., and Di Angelo, L., 2006, "Interoperability among CAD/CAM/CAE Systems: A Review of Current Research Trends," *Geom. Model. Imaging New Trends*, 2006, **2006**(January), pp. 83–89.
- [24] Engelbrecht, S., and Folgar, L., 2009, "Cellular Structures for Optimal Performance," *Proc. SFF ...*, pp. 831–842.
- [25] Aremu, A. O., Brennan-Craddock, J. P. J., Panesar, A., Ashcroft, I. A., Hague, R. J. M., Wildman, R. D., and Tuck, C., 2017, "A Voxel-Based Method of Constructing and Skinning Conformal and Functionally Graded Lattice Structures Suitable for Additive Manufacturing," *Addit. Manuf.*, **13**, pp. 1–13.
- [26] Panesar, A., Abdi, M., Hickman, D., and Ashcroft, I., 2018, "Strategies for Functionally Graded Lattice Structures Derived Using Topology Optimisation for Additive Manufacturing," *Addit. Manuf.*, **19**, pp. 81–94.
- [27] Kim, D. B., Witherell, P., Lipman, R., and Feng, S. C., 2015, "Streamlining the Additive Manufacturing Digital Spectrum: A Systems Approach," *Addit. Manuf.*, **5**, pp. 20–30.
- [28] Algorri, M. E., and Schmitt, F., 1996, "Surface Reconstruction from Unstructured 3D Data," *Comput. Graph. Forum*, **15**(1), pp. 47–60.
- [29] Kaufman, A., and Shimony, E., 1986, "Scan-Conversion Algorithms for Graphics," *Interact. 3D Graph.*, pp. 45–75.
- [30] Emmanuel P., B., 1999, "A Comparison between Photogrammetry and Laser Scanning," *ISPRS J. Photogramm. Remote Sens.*, **54**(2–3), pp. 83–94.

- [31] Nasr, E. A., and Kamrani, A., "IGES Standard Protocol for Feature Recognition CAD System," *Rapid Prototyping*, Kluwer Academic Publishers, Boston, pp. 25–62.
- [32] Janssen, P., and Stouffs, R., 2015, "Types of Parametric Modelling," CAADRIA 2015 - 20th Int. Conf. Comput. Archit. Des. Res. Asia Emerg. Exp. Past, Present Futur. Digit. Archit., (May), pp. 157–166.
- [33] Zegard, T., and Paulino, G. H., 2016, "Bridging Topology Optimization and Additive Manufacturing," *Struct. Multidiscip. Optim.*, **53**(1), pp. 175–192.
- [34] Hällgren, S., Pejryd, L., and Ekengren, J., 2016, "3D Data Export for Additive Manufacturing-Improving Geometric Accuracy," *Procedia CIRP*, **50**, pp. 518–523.
- [35] Frederick, T., Kramer, T., Proctor, F., and Messina, E., 2000, "The NIST RS274NGC Interpreter-Version 3."
- [36] Barequet, G., and Kumar, S., 2002, "Repairing CAD Models," *Proceedings. Visualization '97 (Cat. No. 97CB36155)*, IEEE, pp. 363-370,.
- [37] Mies, D., Marsden, W., and Warde, S., 2016, "Overview of Additive Manufacturing Informatics: 'A Digital Thread,'" *Integr. Mater. Manuf. Innov.*, **5**(1), pp. 114–142.
- [38] 2019, "3MF Website" [Online]. Available: <https://3mf.io/>.
- [39] 2019, "Cloud Powered 3D CAD/CAM Software for Product Design | Fusion 360" [Online]. Available: <https://www.autodesk.com/products/fusion-360>.
- [40] Pilz, M., and Kamel, H. A., 1989, "Creation and Boundary Evaluation of CSG-Models," *Eng. Comput.*, **5**(2), pp. 105–118.