Detection and Resolution of Interpenetrations of Woven Tows

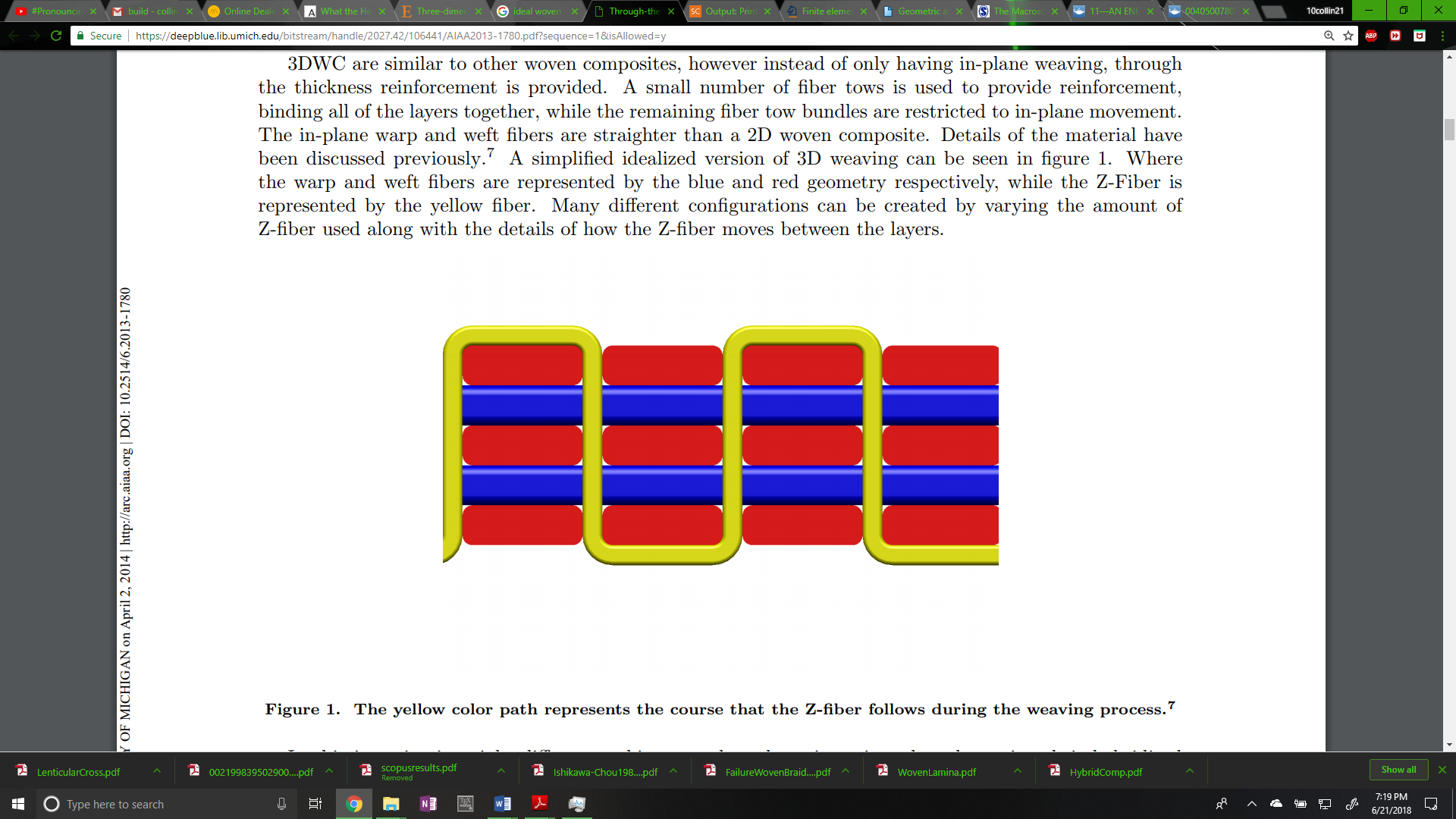
Collin Blake

# Introduction

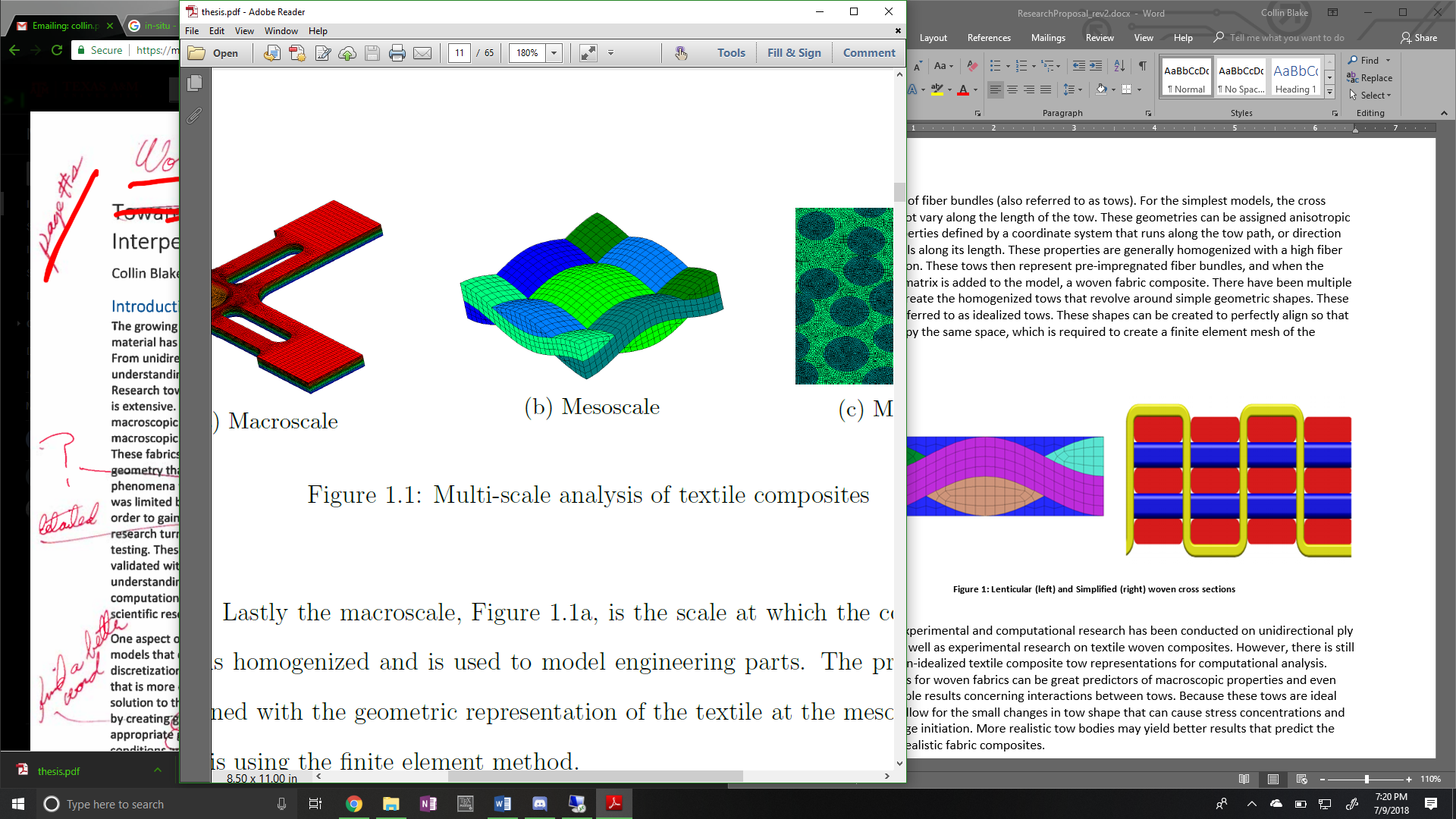
The growing use of fiber-matrix composite materials as both a functional and non-functional material has also increased the need to better understand these materials in all of their forms. From unidirectional ply composites to intricate woven textile composites, the need for understanding material properties and mechanics for these composites has never been higher. Research towards mechanical properties of these geometries in a physical testing environment is extensive. (reference) Many of these tests focused on the idea of discovering the different macroscopic properties and responses of composites. Further complexity was added with the introduction of woven fiber composites. These fabrics brought new challenges in the form of phenomena caused tow interactions that had previously not been explored. During physical testing, equipment was used to measure various aspects, such as strain fields, energy dissipation, etc., of the materials. This is commonly known as in position or in-situ data collection. However, only a limited amount of insight was available during these tests because of the technological limitation. In order to gain a more detailed understanding of the mechanical response for these composites, research turned towards computational modeling of woven fabrics. Equations such as Hooke’s Law and other theories were used in computational models and simulations that were validated with experimental data. (reference) A well-known example is finite element analysis. As these models became more accurate, understanding of both mechanical response and damage initiation increased. The use of computational models and analysis is now a fundamental aspect of most engineering studies and scientific research.

One aspect of computational research for composite analysis is the use of finite element models that can simulate material responses to mechanical loads. Finite element analysis is the discretization of a large, complex problem into smaller, simpler pieces. The result is a problem that is more easily solved mathematically in exchange for accuracy compared to the exact solution to the problem, which is rarely known. This method can be used in composite research by creating two and three-dimensional bodies that mimic the actual geometry of a composite and assigning the appropriate bodies accurate material properties. The model is then given certain boundary conditions and a result can be computed. These results can be insightful to stress concentrations, deformation responses, energy absorption and other attributes that may be of interest. These results are, among other things, affected by how accurately the physical shape of the object can be modeled. Although a finite element mesh (the discretized version of the physical body) is an approximation of a continuous shape, accurate results can be achieved concerning deformation of the body.

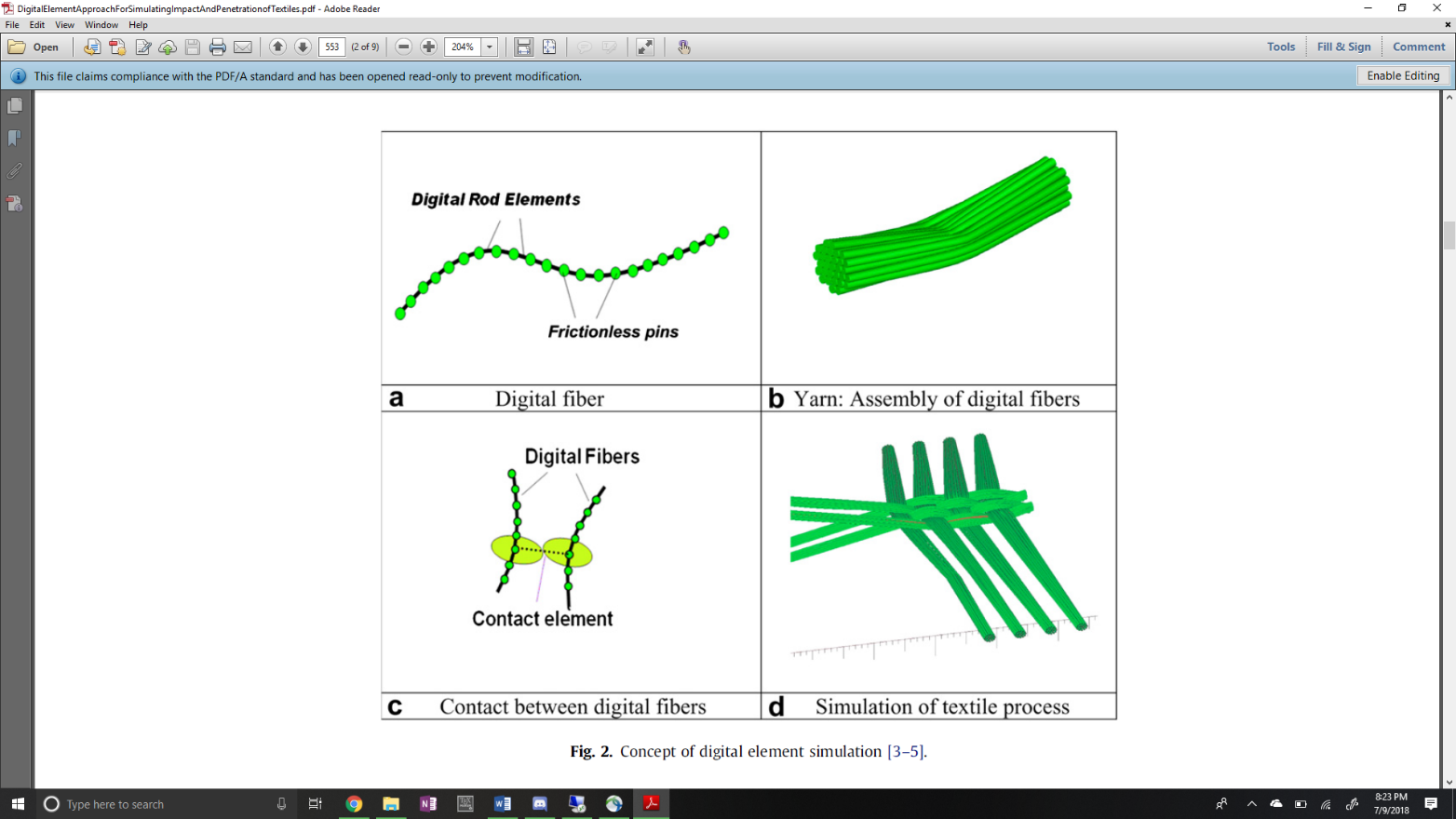
At first, modeling of textile composites using finite element made simplified geometric models in place of the more complex body. These models consisted rounded rectangles and lenticular cross-sections (Figure 1) made by overlapping circles to define the cross-section of fiber bundles (also referred to as tows). For the simplest models, the cross sections do not vary along the length of the tow. These geometries can be assigned anisotropic material properties defined by a coordinate system that runs along the tow path, or direction the tow travels along its length. These properties are generally homogenized with a high fiber volume fraction. These tows then represent pre-impregnated fiber bundles, and when the surrounding matrix is added to the model, a woven fabric composite. There have been multiple methods to create the homogenized tows that revolve around simple geometric shapes. These are usually referred to as idealized tows. These shapes can be created to perfectly align so that no tows occupy the same space, which is required to create a finite element mesh of the shapes.



**Figure 1: Lenticular (left) and Simplified (right) woven cross sections**

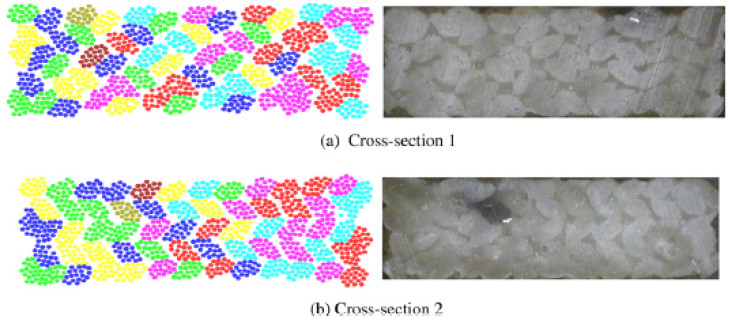


Substantial experimental and computational research has been conducted on unidirectional ply composite as well as experimental research on textile woven composites. However, there is still a need for non-idealized textile composite tow representations for computational analysis. Idealized tows for woven fabrics can be great predictors of macroscopic properties and even yield acceptable results concerning interactions between tows. Because these tows are ideal they do not allow for the small changes in tow shape that can cause stress concentrations and lead to damage initiation. More realistic tow bodies may yield better results that predict the response of realistic fabric composites.



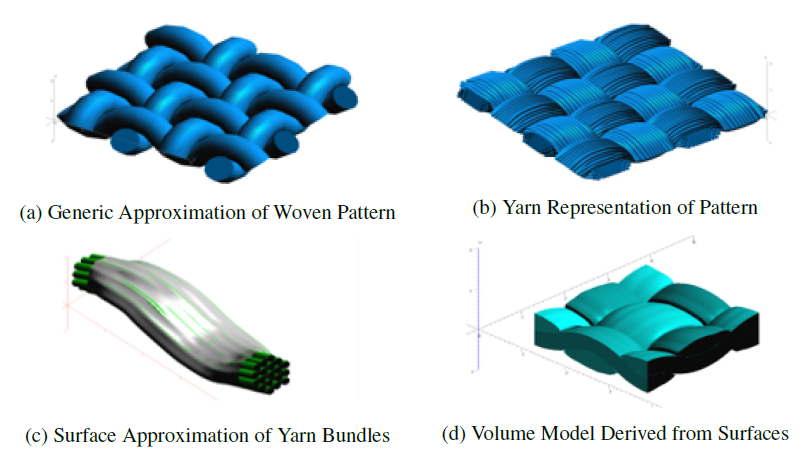
**Figure 2:Digital chain simulation process [s]**

One method to creating more realistic woven geometries is to simulate the process that manufactures employ to create the fabrics (Wang\_Sun). The process begins by simulating bundles of fibers as "yarns". These yarns are made up of digital elements (cylindrical bars connected by friction-less pins) chained together (Figure 2.a). Each bar is given a stiffness in the longitudinal direction that amounts to a large value which eliminates yarn stretching. Then, a contact problem is solved where pins between two digital fibers can create contact forces between each other as well as friction forces through a contact element (Figure 2.c) resulting in realistic interactions between the digital chains. (Wang\_Sun1) The result is fiber bundle cross sections that are similar to micro-CT scans from actual woven specimens, shown in Figure 3, from Wang\_Sun1.



**Figure 3: Simulated vs. Actual Fiber Bundle Cross Sections**

While there is possibly other software that can accomplish this level of similarity between simulation and reality, only two were explored. The first is Digital Fabric Mechanics Analyzer (DFMA) from Kansas State, overseen by Youqi Wang and students. The other is Virtual Textile Morphology Suite (VTMS), developed by Eric Zhou at AFRL. It should be noted that Eric Zhou is a former student of Youqi Wang and is cited in a previous paper (Wang\_Zhou1).

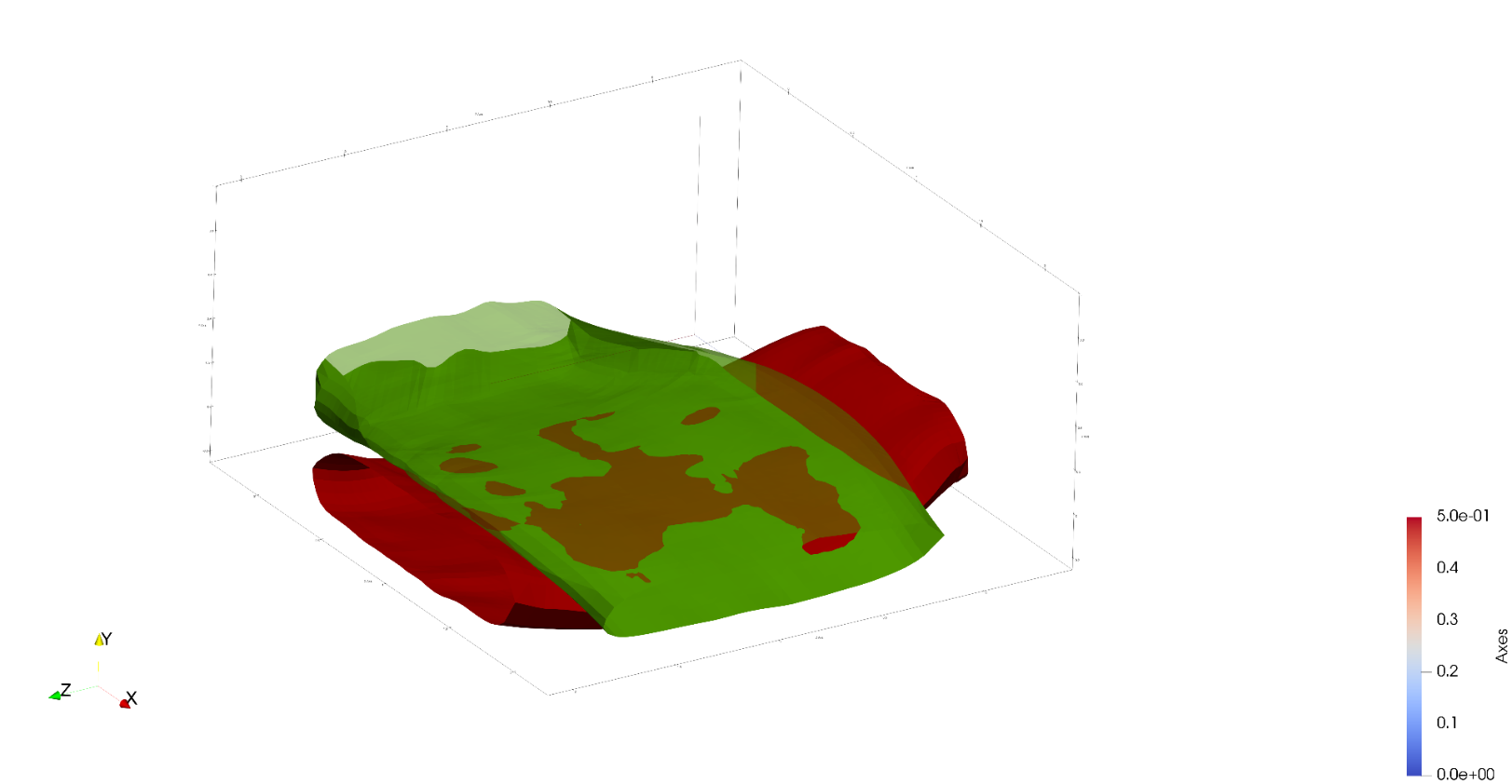
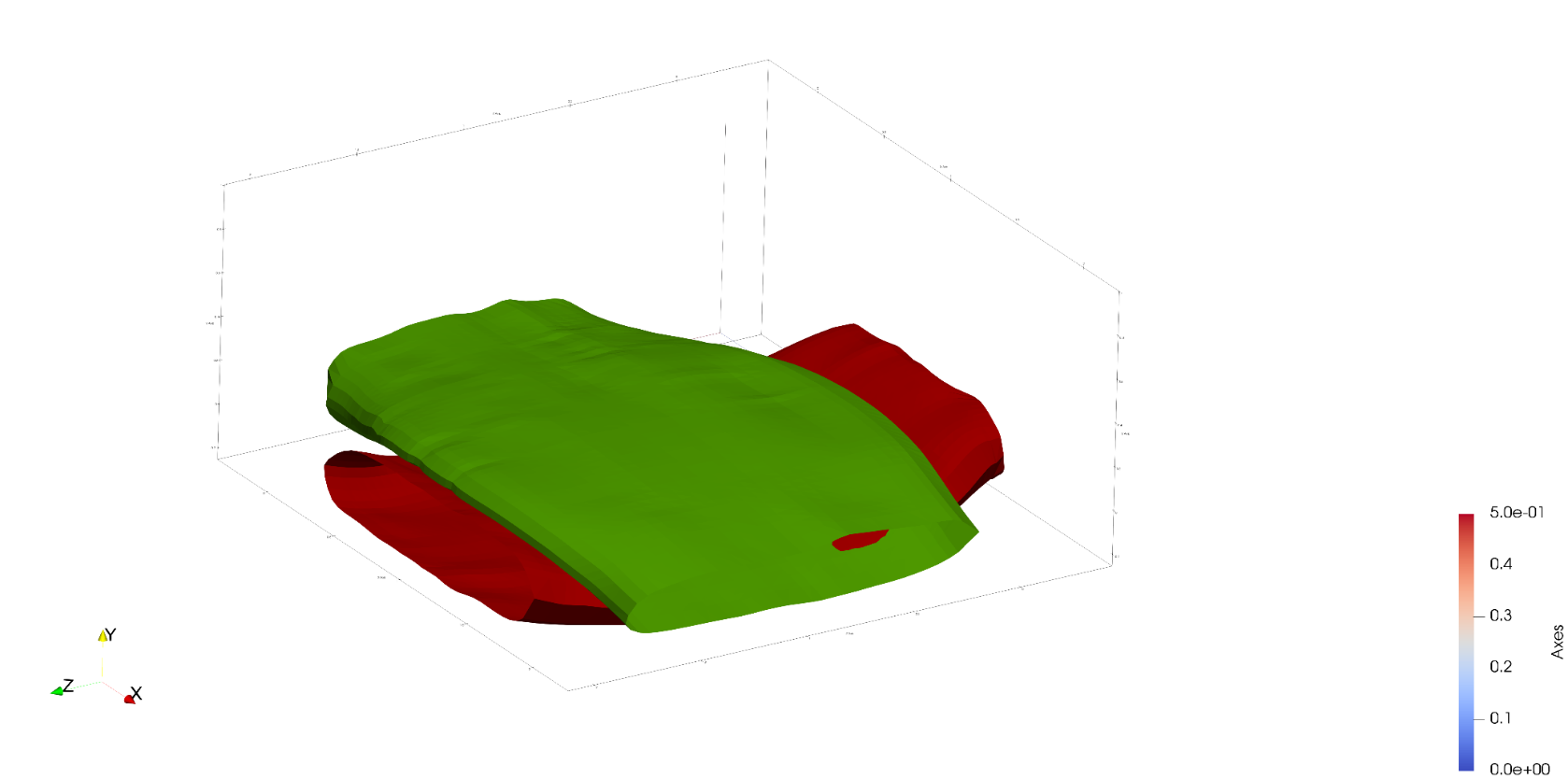


**Figure 4: Evolution of Weave Textile Geometry**

It is from VTMS that the base geometry and surface mesh that is used in this study originates. Figure 4 shows the process visually. The surface inter-penetrations come as a result from the geometries shown in Figure 4.d.

The reasoning behind creating surface and volume approximations (Figure 4.c and d) is that the computational cost of analyzing many bundles that represent woven fibers is very high. Instead, researchers currently are content with using a surface or volume approximations and applying material properties found in experiments.

1. **Transparent upper tow showing interpenetrations**
2. **Tows in close proximity**



**Figure 5: Region of close tow geometries with interpenetrations**

Once a surface representation is created, surfaces in close proximity have the ability to penetrate into each other, as shown in Figure 5. Physically, the two surfaces would come into contact and create some form of surface. This reaction is not represented here because the surfaces are created after the simulation process is done. These interpenetrations represent the error in approximating the yarn bundles as a surface to apply homogenized properties to for analysis. Here in lies the focus of this study. Traditional finite element software requires that two geometries cannot occupy the same space and must have compatible meshes along any boundaries that they may share. These regions must be fixed if a traditional FEA is to be conducted.

# Literature Review

The idea of solving the penetration (also know as intersection) issue is one documented well in computer aided modelling. Various approaches have been used to detect whether two shapes, in both two and three dimensions, occupy the same space at any given point. Jimenez, Thomas, and Torras asserted that intersection scenarios can be static and time dependent. For both cases, a static intersection step must be calculated. For surfaces that are polyhedral (defined as having multiple flat faces and can be open or closed), Dobkin and Kirkpatrick (source) state that a hierarchical representation of the polyhedral can be used. This representation reduces the computation time required to detect an intersection. They also state that the representation of an intersection is embodied in the hierarchies of the two parent polyhedra. During this method, the minimum distance between the shapes is calculated, and said to be null if the shapes intersect. This framework is useful because once a hierarchy is established for a shape, it can be used for every query involving the shape. Its limitation is that it requires the polyhedrals to be convex.

Canny (source) discusses in his book a method for the more general case of a polyhedral with convex faces. He states that two intersection cases exist for polyhedrals, face-to-node contact (Type-A) and edge-edge (Type-B). By associating a predicate that is true or false for each case, a series of tests can be run on a polyhedra and if either predicate remains true at the end of the tests, the shapes are said to have intersected. This method is useful because it requires simple vector math to run the tests. The method itself does not directly identify the case of containment (one shape lying completely in another). However, a simple ray intersection algorithm can determine if containment is occurring.

The most general case for a polyhedral shape is a non-convex shape. For this case there are two trains of thought. The most popular response is to subdivide the domain into convex sub-domains and then perform similar intersection operations that apply to convex polyhedrals. Two popular methods are decomposition into smaller convex polyhedral (source) and decomposition of only the surface into convex surfaces (source). After sub-division, the smaller, convex shapes can then use a multitude of intersection algorithms that apply to convex shapes. The main drawback to this sub-division method is the increase in number of operations and intersection checks required. The more complex and less used method is a direct approach to calculating the intersection. This usually involves a two-step process to identify edge-face intersections (source) involving a ray-intersection algorithm to determine if edge end points lie on opposing sides of any face of a polyhedron. By counting the number of intersections an edge has with faces on the polyhedron, it can be determined if the edge intersects with the polyhedron. Another method that does not require computing these intersection tests involves computing the signs of the determinants of a set of linear equations. Suppose there exists a linear equation that determines where a surface node lies in space. These equations can be set up so that they calculate the location of certain polyhedron surface nodes. The equations are set up to quantitatively calculate the predicates mentioned previously (source). They can be assembled in matrix form and by calculating the sign of certain determinants it can be determined if a certain case of intersection occurs. In this way the method does not care about the convexity of the shape and can be applied to the most general of cases. The main drawback of this method is that it requires extensive setup of the shape vertex equations as well as the framework for solving the linear equations.

Although detection algorithms for discretized surfaces are well documented, they are not the only method for detection. There are a variety of methods to translate polyhedral surfaces into non-polyhedral descriptions of these surfaces. With these surfaces there are also methods to detect intersections between surfaces of similar description type. Two common types of non-polygon surfaces are implicit surfaces and parametric surfaces. Implicit surfaces are in three-dimensional space and are defined by a function that, when the function is evaluated at a point on the surface in three-dimensional space, the function is equal to zero. If the function is a polynomial in *x*, *y*, and *z*, it is considered algebraic (source). These functions may also be quadric, which are second degree polynomials in *x*, *y*, and *z*. The other typically used non-polygon surface is a parametric surface. These surfaces in three dimensions are described by functions that have two input parameters. As a result, they are generally not closed but easier to polygonalize and render. A special class labeled Non-Uniform Rational B-Spline (NURBS) have gained traction in computer aided design software (source) and possess some ideal properties that make them easier to use. For each of these non-polygon surfaces, there are algorithms for detecting intersections.

For implicit surfaces, the available algorithms are limited. Pentland and Williams (source) discuss the implementation of an “inside-outside” functions that use the object’s canonical frame (no rotation, centered on origin) and current location. Once the function is formed the surface to be tested has its points tested against another surfaces inside-outside functions. If a point is determined to be inside, it is intersecting. One main advantage of this algorithm over any polygon intersection detection algorithms is that it can obtain a good closed form solution that approximates interpenetration region depth, area, and shape. This is very valuable when the shapes are static and simply detecting intersections is not enough. However, this method is only applicable to implicit functions and has drawbacks in terms of robustness as it relies on point samples. Lin and Manocha (source) have discussed algorithms that extend their previously mentioned hierarchical representation algorithm that used curved models made of splines and algebraic surfaces, which work best on low degree curves.

Parametric surfaces have a larger set of explored algorithms for intersection detection. There are four main methods: lattice, subdivision, tracing, and analytic methods. This review will cover the latter three as they are the most relevant to this research. The subdivision method works by subdividing both surfaces in parallel. By recursively subdividing and testing for intersections of the subdomains, the domain of the intersection region can be approximated. The intersected subdomains can be further subdivided to more accurately describe the intersection region (source). A method very similar to this is used by Drach et al (source) to determine if surface nodes interpenetrate a surface. They then use another technique to remove interpenetrating surface nodes until they are all corrected. The main drawbacks to this approach is that the desired level of refinement of the intersection region negatively affects the computation time. As the desired level of refinement increases, so does the computation time.

A second method used is tracing. This method starts by first finding a known point of intersection, of which there are multiple methods to choose from (sources). Then, the intersection curve is traced along by starting at the previously calculated point of intersection and a moving along a determined vector by a set distance. The vector is found by intersecting the tangent planes of the two surfaces at this point and calculating the direction of the line that defines the intersection of these planes. The distances along this vector is predetermined and is the determining factor in the amount of “refinement” the curve has. One issue the method faces is determining if a curve has reached its starting position. This is usually posed as a system of algebraic equations (source) or a differential equation problem (source). This method can yield very good results when trying to identify a boundary curve for the interpenetration regions.

A third method is the analytic method. Generally, one surface is made into an implicit representation of the surface (source) and creates a scalar function in the two parametric variables. The root locus of these functions in the parametric variable plane is the preimages of the intersection curve (sources). In other words, this method creates a series of algebraic equations that describe where one surface lies on another. In the case that they intersect, the equations can be solved and the result is a curve that defines where and how the two surfaces intersect. This method can be difficult to implement as it requires knowledge of how to accomplish the parametric-implicit conversion as well as the frame work for multiplying polynomials and solving multi-basis functions.

Drach et al have used a couple of these techniques to solve a very similar problem to the one posed for this research (source). They have used a variety of software to produce realistic woven fabric geometries and have also encountered the tow interpenetration problem. Their first attempt was using a variation of the subdivision method where they create voxels (or bounding boxes) that collectively encompass the tow volume for the host tows. The tow being checked against the host is still in its polygon form and they check the host voxels against the surface nodes of the other tow. This allows them to quickly identify interpenetrating nodes. They then move the penetrating node in the mean normal direction of all the interpenetrating surface elements inside the host. When they detect no more interpenetrations they consider them fixed. In paper published shortly after, they updated their method to also account for edge-edge intersections as well. This method accomplishes the task of fixing interpenetrations however results in two tows not in contact. During the removal of interpenetrating nodes, the nodes are moved until they are a minimum distance away from the host tow. This allows for small matrix pockets between tows that are not present in actual CT. This can cause minor yet important inaccuracies when observing the interaction between tows. It is the goal of this research to further reduce these inaccuracies.

There are a number of possible methods for detecting interpenetrations between polyhedral surface representations. Many use the same polygon representation that is similar to the standard output from VTMS while others depend on mathematical (parametric and implicit) representations. Using a method that uses the polygon form will be quicker but less accurate than its mathematical counterparts, which should have a similar but inverted trade off. Both will be explored for their potential in solving this problem.

# Research Problems

This research can be summarized into three main goals. The goals of this research are:

1. Determine if the default VTMS data is enough to solve the interpenetrations or if another surface data type (such as NURBS or parametric surfaces) is more useful.
2. Develop methods for each representation type that will accurately identify interpenetration regions.
3. Develop methods that resolve interpenetrations for the chosen surface representation type.

To understand the reasoning for these goals, they will be discussed individually.

## Surface Representation Data Types

The first objective will be to explore possible representations of the data from VTMS and how they relate to the default types given from this software. VTMS can export a tow surface as a surface represented as polygons that result in a faceted surface. VTMS also has another data type that can be easily made into a faceted surface but is more structured than the other export type. Other types previous mentioned such as a NURBS, parametric, or implicit surface also can be used to identify interpenetration of surfaces. This objective is a prerequisite to the remaining objectives as it is important to us the best suited data type for identifying and resolving interpenetration regions between surfaces.

The origin software VTMS is written in C++ and it is the goal of this research to create a set of software that can implemented in not just VTMS but other software as well. Therefore, the methods developed will be written in C++. Completion of this objective will result in an easy to use software that can translate the tow surfaces given from VTMS to other surface types.

## Identification of Interpenetration Regions

The second objective will determine an accurate way to identify interpenetration regions for the representation types chosen in the first objective. It is important that the detection algorithm correctly identify the regions interpenetrating so that all incompatibilities may be fixed. The results of completing this objective will give all the information needed to correctly fix the interpenetrations for the respective representation type for the geometries.

## Resolution of Interpenetration Regions

The third objective is to develop a method that can resolve the issue of interpenetrations for each representation type identified in the first objective. This method will fix the interpenetration regions and also will export the interpenetration region data in a form that is descriptive enough that the user may implement their own solution. A user may wish to implement a solution that is complex and outside the scope of this research.

# Research Plan

A short summary of methods and expected results are provided in Table 1 along with a tentative outline.

Table 1. ….

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Task** | **Description** | **Time** |
| 1 | Determine best surface representation candidates | Analyze multiple surface data description types to determine one or more ideal candidates. An in-depth study will occur that resembles the previous literature review except that the ideal candidate(s) will be chosen based on their being previously used in successful attempts at identifying interpenetrations. The expected results are one or more representation types that show promise in solving interpenetrations. | 2 months |
| 2 | Identify the region of interpenetrations for representations chosen | Identify methods that can accurately identify the regions of interpenetration between the tow geometries. The ideal methods will not only identify interpenetration regions but also return data that will have the potential to be useful for solving the interpenetrations. This could include boundary curves, element sets, node sets, and other data that will be useful in solving the interpenetrations. | 3 months |
| 3 | Resolve the interpenetrations | Use the representations and methods previously identified to correct interpenetrations. Once the best method(s) are chosen, a resolution will be developed. The resolution will result in a compatible (in terms of meshing) contact surface. This surface could then have a cohesive zone (or other scheme for modelling the contact properties at the surface) to model the effect of the surfaces being in contact with each other. Ideally the surface would be an average of the interpenetrating region from each tow. However, it is easier to start with a Boolean type mesh operation of selecting one region as the master. Other resolution types could prove useful with more research. | 5 months |

# References

To be added.