

CAD and constraint-based geometric modelling algorithms for 2D and 3D woven textile structures

Martin A. Smith and Xiaogang Chen ⁺

School of Materials, University of Manchester, Manchester, UK

(Received April 25, 2008, accepted July 12, 2008)

Abstract. Geometric models of woven textile structures form the basis for mechanical analysis of properties of textile assemblies and for components made from such textiles. Existing geometric models often ignore yarn interpenetration and compression, yet those models are frequently combined with mechanical models for property simulation and prediction. Meshes generated from geometric models that disregard yarn interpenetration contain overlapping elements and are generally unsuitable for finite element analysis. In contrast to many existing modelling approaches, the new algorithms guarantee to eliminate yarn interpenetration, do not assume yarns are incompressible and generate meshes with no overlapping elements, thus resulting in more accurate modelling of woven textile structures. In this paper, efficient, robust and novel algorithms are described and evaluated for geometric modelling of 2D and 3D woven textile structures. Hermite curves and Catmull-Rom splines are used to model yarn path centrelines. Existing 2D yarn cross-sectional shapes are extended from 2D to 3D modelling. Additionally, a new approach to modelling yarn cross-sections is explored that captures a wide variety of observed irregular yarn cross-sectional shapes. The constraint-based geometric modelling algorithms use a hierarchical binary tree data structure for yarn collision detection and determination. Collision response eliminates yarn interpenetration. The algorithms apply to all woven structures regardless of dimension, interlacement or yarn cross-sectional shape. The results show visualisations of woven structure geometries are a qualitative improvement over existing techniques. Furthermore, the proposed geometric modelling approach is extensible to include the energy-based approach and finite element simulation.

Keywords: woven structure, yarn, geometric modelling, CAD, collision.

Nomenclature:

a	semi-major ellipse axis
b	semi-minor ellipse axis
$\mathbf{b}(t)$	yarn centreline unit binormal
$\mathbf{c}(t, \theta)$	circular parametric yarn cross-section
\mathbf{d}_i	Hermite curve tangent vector
$\mathbf{e}(t, \theta)$	elliptical parametric yarn cross-section
EI	flexural rigidity
H	height of oriented-bounding rectangle enclosing racetrack/lenticular cross-section
I	collision interval
$\mathbf{l}(t, \theta)$	lenticular parametric yarn cross-section
M	bending moment
$\mathbf{n}(t)$	yarn centreline principal unit normal
\mathbf{p}_i	Hermite curve endpoint
$\mathbf{p}(t)$	Hermite interpolant

⁺ Corresponding author. Tel.: +44-161-306 4113.
E-mail address: xiaogang.chen@manchester.ac.uk

r_{circ}	circular cross-section radius
r_{lent}	lenticular cross-section radius
r_{race}	racetrack cross-section radius
$\mathbf{r}(t)$	Catmull-Rom interpolant
$\mathbf{r}(t, \theta)$	racetrack parametric yarn cross-section
s	Cardinal spline tension
$\mathbf{t}(t)$	yarn centreline unit tangent
u	tangent vector magnitude for Hermite approximation of a circular arc
w	width of oriented-bounding rectangle enclosing racetrack/lenticular cross-section
y	$y = f(x)$ where y is the beam deflection as a function of the x -coordinate
α	half-angle subtended by Hermite approximation of a circular arc
k	yarn centreline curvature
θ	yarn cross-section angle
θ_{lent}	lenticular cross-section cut-off angle
τ	bracketing subinterval tolerance

1. Introduction and Survey

Technical textiles are becoming prevalent in high performance and safety-critical industries such as automotive and aerospace. As a result, modelling woven structures for descriptive and predictive properties has received much attention from the textile research community. Modelling can be broadly classified as geometric (descriptive) or mechanical (predictive) and the two categories are not mutually exclusive as the distinction between the two is somewhat blurred. In practice, researchers may simultaneously adopt both approaches [1]. It is evident that in order to maximise the predictive capabilities of woven structure models, mechanical modelling such as force-equilibrium modelling [2, 3] or the energy-based approach [4-9] is essential and can lead to useful predictions such as biaxial load deformation and yarn compression.

Despite the predictive benefits of mechanical modelling, improving the fidelity of purely geometrical woven structure models independent of the underlying structural mechanics is nevertheless challenging and worthwhile and has many practical benefits: Firstly, geometric models are integral to mechanical models as mechanical modelling frequently relies on geometric models to formulate predictions. Improving geometric models can improve the predictive capability of mechanical models, many of which rely on the dated and highly idealised Peirce [10] model and its derivatives [11,12]. Furthermore, yarn interpenetration is not possible in actual woven structures although it is frequently overlooked and often appears in software-generated geometric definitions [13]. Secondly, geometric modelling is essential in computer-aided design (CAD) applications that allow designers to create, manipulate and visualise the spatial layout, shape and appearance of woven structures. Of particular interest is the capability afforded by CAD in prototyping woven structures prior to costly testing, rework and manufacture. Thirdly, geometric representation and visualisation of woven structures are useful for textile teaching and research. Finally, geometric woven structure models can facilitate discussion and collaboration between textile designers, researchers, technologists and other project stakeholders.

Predictive mechanical modelling is often too complex to understand or apply (particularly for complex textile assemblies) and is inaccessible to users and specialists alike [14]. The descriptive geometrical approach is mathematically simpler and is to be preferred whenever it is adequate for the purpose at hand [7].

The seminal work by Peirce [10] described a systematic study of woven fabric geometry. In his most quoted work, yarn paths are modelled with circular arcs connected by straight line segments and yarn cross-sections are assumed circular and incompressible. Consequently, the model is a highly idealised approximation. Lin and Newton [15] generated yarn path centrelines using B-splines. Specifically, control points were placed at locations where warp and weft yarns interlace and at midpoints between points of interlacement. The authors concede the yarn path does not pass through the control points. As a result there is no precise control over yarn path direction. Consequently, user intervention may be required to manually adjust control points to ensure yarn arrangements reflect the geometry of the weave. Liao and Adanur [16]

used 2D parametric curves to represent yarn path centrelines and cross-sections. Yarn cross-sections were assumed elliptical [10], racetrack [11] as well as circular [10] forms. Liao and Adanur presented example equations for a sinusoidal yarn path centreline. Description of how to generate particular yarn path centrelines for the visualised 2D fabric models was omitted. A variety of 2D fabric models (plain, twill, sateen, backed, pile and braid) were visualised using Mathematica. Compared with bespoke CAD, the use of Mathematica for model visualisation is restricted to tool-specific functionality and, dependent on choice of interface, may also be unsuitable for end-users due to its learning curve and arcane syntax. Adanur and Liao [17] reported a very similar technique to include 3D fabric structure models. The authors referred to yarn path centreline generation as described by Ma et al. [18]. However, Ma et al. approximated yarn paths with line segments. Furthermore, the models used explicit polygon lists that hinder graphics performance and should therefore be avoided. Jiang and Chen [19] modelled the yarn path and geometry using a quadratic (second-degree) polynomial. A quadratic representation does not allow a curve segment to interpolate two specified endpoints with specified derivatives at each endpoint [20]. Furthermore, a quadratic polynomial is restricted to plane curves [20] and is therefore unsuitable for representing out of plane yarns that may result from downstream mechanical modelling. The adopted B-spline form requires interpolation points to be computed and may produce round-off [22]. In addition, the computational overhead required by the convergence algorithm may be prohibitive for large weaves or repeats, particularly in an interactive setting where real-time response is desirable. Furthermore, there is no apparent merit in the proposed interpolating curve algorithm over existing interpolating curves in the literature [20]. User manipulation of control points is required to deform the yarn cross-section. Moreover, the yarn cross-section was assumed constant along the yarn length.

This paper focuses on the descriptive over the predictive and presents novel constraint-based geometric modelling algorithms implemented as part of the WeaveStudio CAD/CAM software tool [21] aiming for more accurate fabric geometry. This aim will be achieved through the following objectives: to model yarn paths derived from beam theory, to extend existing 2D models to 3D visualisation; to empirically derive parametric yarn cross-sectional modelling based on existing literature; to geometrically account for yarn bending and flattening at points of interlacement; to devise constraint-based algorithms to eliminate yarn interference irrespective of woven structure type and to briefly evaluate algorithm efficiency and accuracy.

2. Yarn Path Centreline

Yarns are fundamental components of woven structures. Yarn paths in real woven fabrics are complicated and are dependent upon fabric construction, yarn properties, weaving condition and mechanical loading on the fabric which causes the yarns to assume positions of minimum energy subject to external constraints and frictional effects [15]. A mathematical model of the yarn path centreline can be constructed using beam theory [23]. The model starts with expressing the centreline curvature in terms of the beam bending moment and is known as the moment-curvature equation as shown below.

$$\kappa = \frac{M}{EI} \quad (1)$$

It can be seen from Eq. (1) that the curvature is directly proportional to the bending moment M and inversely proportional to the flexural rigidity EI . From the calculus of vector-valued functions, the curvature of a space curve is the magnitude of the rate of change of the direction of the unit tangent vector with respect to arc length. The curvature of a smooth curve given by the vector-valued function \mathbf{r} is

$$\kappa = \frac{|\mathbf{r}' \times \mathbf{r}''|}{|\mathbf{r}'|^3} \quad (2)$$

For the special case of a plane curve with equation $y = f(x)$, curvature is expressed as

$$\kappa = \frac{|y''|}{[1 + (y')^2]^{3/2}} \quad (3)$$

Combining Eq. (1) with Eq. (3) gives

$$\frac{|y''|}{[1+(y')^2]^{3/2}} = \frac{M}{EI} \tag{4}$$

For small deflections it can be assumed $(y')^2$ is negligible compared to unity and therefore Eq. (4) can be simplified to give the differential equation of the deflection curve

$$y'' = \frac{M}{EI} \tag{5}$$

Gere [23] describes the assumptions that need to be satisfied to apply Eq. (5). In particular, for a linear equation of the bending moment $M(x) = Ax + B$ and integrating Eq. (5) twice results in a cubic polynomial solution of the deflection curve. As Eq. (5) is a second order equation, the first integration gives the slope y' and the second integration gives the deflection y .

$$y(x) = Ax^3 + Bx^2 + Cx + D \tag{6}$$

Based on Eq. (6) a section of yarn path centreline between two successive points of yarn interlacement can be approximated using cubic Hermite interpolation. It can be shown [29] that equations of structural mechanics describing the elastic deformation of a simply supported beam are analogous to the second derivate form of a cubic Hermite Curve. Despite the simplifying assumptions of applying beam theory to modelling yarn path centrelines, the resulting cubic polynomial is an improvement on line segments, sinusoidal curves and planar curves frequently found in the literature.

A Hermite curve segment is defined by two endpoints (p_i, p_{i+1}) and one tangent vector at each end point (d_i, d_{i+1}) . The Hermite interpolant where $t \in [0,1]$ is

$$\mathbf{p}(t) = \begin{pmatrix} t^3 & t^2 & t & 1 \end{pmatrix} \begin{pmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{p}_i \\ \mathbf{p}_{i+1} \\ \mathbf{d}_i \\ \mathbf{d}_{i+1} \end{pmatrix} = (2t^3 - 3t^2 + 1)\mathbf{p}_i + (t^3 - 2t^2 + t)\mathbf{d}_i + (-2t^3 + 3t^2)\mathbf{p}_{i+1} + (t^3 - t^2)\mathbf{d}_{i+1} \tag{7}$$

From basic set theory, if $U = \{t | 0 \leq t \leq 1\}$ and $S = \{t | 0 < t < 1\}$ then the complement of S is $S' = \{t | t \in U \text{ and } t \notin S\}$ and generates endpoints and S generates interior points on the Hermite curve segment. In order to extend Hermite interpolation to accommodate an entire yarn path centreline, the Catmull-Rom spline [24] features exact interpolation and controls shared tangent vectors at the join points. Furthermore, it overcomes the main disadvantages of cubic splines, namely the lack of local control and the need to solve a potentially large system of linear equations [25] the size of which is dependent upon the number of yarn interlacement points. The shared tangent vectors d_i and d_{i+1} at points p_i and p_{i+1} respectively are

$$\mathbf{d}_i = \frac{(\mathbf{p}_{i+1} - \mathbf{p}_{i-1})}{2} \tag{8}$$

$$\mathbf{d}_{i+1} = \frac{(\mathbf{p}_{i+2} - \mathbf{p}_i)}{2} \tag{9}$$

The Catmull-Rom interpolant where $t \in [0,1]$ is obtained by substituting Eqs. (8-9) into Eq. (7) to obtain

$$\mathbf{r}(t) = \frac{1}{2}(-t^3 + 2t^2 - t)\mathbf{p}_{i-1} + \frac{1}{2}(-t^3 + t^2)\mathbf{p}_i + (2t^3 - 3t^2 + 1)\mathbf{p}_i + \frac{1}{2}(t^3 - 2t^2 + t)\mathbf{p}_{i+1} + (-2t^3 + 3t^2)\mathbf{p}_{i+1} + \frac{1}{2}(t^3 - t^2)\mathbf{p}_{i+2} \tag{10}$$

As with the Hermite interpolant, S' generates endpoints and S generates interior points on the Catmull-Rom spline. A variant on the Catmull-Rom spline is the Cardinal spline that introduces a tension parameter where $s \in (-\infty,1)$ to modify the magnitude of a shared tangent vector \mathbf{d}_i

$$\mathbf{d}_i = \frac{1-s}{2}(\mathbf{p}_{i+1} - \mathbf{p}_{i-1}) \tag{11}$$

Fig. 1 shows the variation in curvature and tangent vector magnitude at a point of interlacement as a function of s . Low and high tension parameters are useful for geometrical modelling of tight and loose weave structures respectively.

3. Yarn Cross-section

In addition to modelling the yarn path centreline, yarns have volume and occupy a region of space. Yarns are composed of an assembly of fibres that vary considerably in material, arrangement, mechanical properties and geometry. Inherent in the Catmull-Rom spline is the loss of second-order continuity and this is potentially restrictive when specifying the orientation of the yarn cross-section along the yarn path centreline. As a result, the Frenet frame is unsuitable because it requires $r(t)$ be twice differentiable so that the normal vector is well defined. However, the Catmull-Rom spline is C^1 continuous and therefore has a well defined tangent vector. The orthonormal basis vectors for the yarn cross-section are

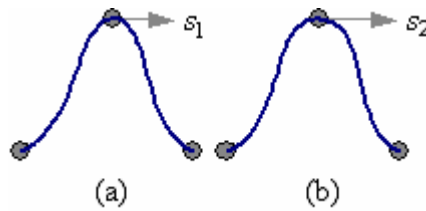


Fig. 1. Tension parameter $s_1 > s_2$ results in a smaller tangent magnitude and larger curvature (a) and a larger tangent magnitude and smaller curvature (b).

$$\mathbf{t}(t) = \frac{\mathbf{r}'(t)}{|\mathbf{r}'(t)|} \tag{12}$$

$$\mathbf{n}(t) = \frac{\mathbf{t}(t) \times \mathbf{j}}{|\mathbf{t}(t) \times \mathbf{j}|} \tag{13}$$

$$\mathbf{b}(t) = \mathbf{n}(t) \times \mathbf{t}(t) \tag{14}$$

Vector \mathbf{j} must be chosen so as not to be parallel to $\mathbf{t}(t)$ otherwise $\mathbf{n}(t) = 0$ and the coordinate system cannot be constructed. $\mathbf{j} = \langle 0, 1, 0 \rangle$ is chosen because the yarn path centreline is never parallel to \mathbf{j} for the actual geometric models.

The original circular cross-section geometry [10] was extended to include racetrack [11] and lenticular geometry [12] to account for non-circularity of yarn cross-sections, extensions to the non-plain weave and the mechanics of yarn flattening deformation. In order to represent woven structures as true 3D models, the original 2D cross-sections are extended to 3D geometric modelling. For Eqs. (15-18) $\{(t, \theta) | t \in [0, 1], \theta \in [0, 2\pi)\}$ unless otherwise stated.

$$\mathbf{c}(t, \theta) = \mathbf{r}(t) + r_{circ}(\cos(\theta)\mathbf{n} + \sin(\theta)\mathbf{b}) \tag{15}$$

$$\mathbf{e}(t, \theta) = \mathbf{r}(t) + a \cos(\theta)\mathbf{n} + b \sin(\theta)\mathbf{b} \tag{16}$$

$$r_{race} = \frac{h}{2} \tag{17a}$$

$$\mathbf{r}(t, \theta) = \begin{cases} \mathbf{r}(t) + (\frac{w}{2} - r_{race})\mathbf{n} + r_{race}(\cos(\theta)\mathbf{n} + \sin(\theta)\mathbf{b}) & -\pi/2 \leq \theta \leq \pi/2 \\ \mathbf{r}(t) - (\frac{w}{2} - r_{race})\mathbf{n} + r_{race}(\cos(\theta)\mathbf{n} + \sin(\theta)\mathbf{b}) & \pi/2 \leq \theta \leq 3\pi/2 \end{cases} \tag{17b}$$

$$r_{lent} = \frac{w^2 + h^2}{4h} \tag{18a}$$

$$\theta_{lent} = \cos^{-1}\left(\frac{2wh}{w^2 + h^2}\right) \tag{18b}$$

$$\mathbf{l}(t, \theta) = \begin{cases} \mathbf{r}(t) + (\frac{h}{2} - r)\mathbf{b} + r_{lent}(\cos(\theta)\mathbf{n} + \sin(\theta)\mathbf{b}) & \theta_{lent} \leq \theta < \pi - \theta_{lent} \\ \mathbf{r}(t) - (\frac{h}{2} - r)\mathbf{b} + r_{lent}(\cos(\theta)\mathbf{n} + \sin(\theta)\mathbf{b}) & \pi + \theta_{lent} \leq \theta < 2\pi - \theta_{lent} \end{cases} \tag{18c}$$

Eq. (15), Eq. (16), Eq. (17b) and Eq. (18c) parameterise circular, elliptical, racetrack and lenticular yarn cross-sections respectively. The parameterisations ensure cross-sections are plane, symmetrical about the unit binormal vector \mathbf{b} and orthogonal to the unit tangent vector \mathbf{t} and are therefore consistent with the beam theory adopted for the yarn path centreline. Notwithstanding the consistency of parameterisations with beam theory, the reality is more complex as shown in Fig. 2.

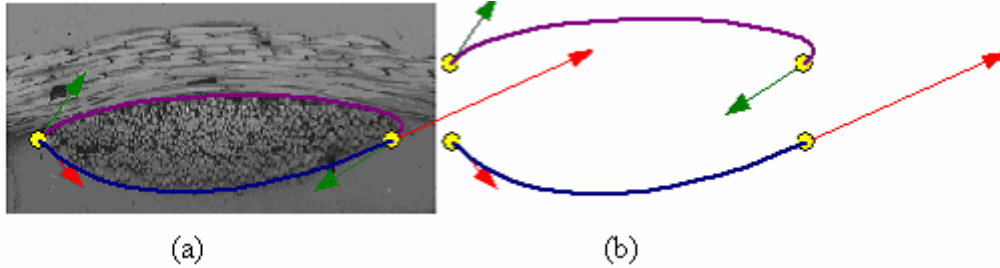


Fig. 2. Photomicrograph showing irregularity of yarn cross-section. Hermitian geometry superimposed on yarn cross-section (a) and exploded view of constituent Hermite curves (b).

The yarn cross-section in Fig. 2 is neither lenticular nor the frequently observed bow shape [14]. A composite curve formed by joining together two or more Hermite curve segments can capture a wide variety of observed yarn cross-sectional shapes. The parametric nature of Hermite curves is also useful in the context of yarn flattening and collision response. Further elaboration is described in Section 4.3 Bounds Update for Collision Response.

4. Constraint-based Geometric Modelling

Solving geometric constraints ensures yarns only just touch at the regions of overlap and eliminates interpenetration. The top-level algorithm drives the constraint-based geometric modelling and is described in the first subsection. Subsequent subsections describe the hierarchical collision, bounds updating and weave discretisation algorithms called by the top-level algorithm.

4.1. Top-level algorithm

The iterative top-level constraint-based modelling algorithm uses a numerical bracketing method analogous to interval bisection.

```

ConstraintIteration()
1 collision = HierarchicalCollision()
2 if (collision = false and status = FIRSTITERATION)
3     AdvanceToNextInterlacementPoint()
4 else if (status = BISECTION)
5     UpdateBisectionBounds()
6     DiscretiseWeave()
7 else if (collision = true)
8     UpdateCollisionBounds()
9     status = INITIALCOLLISION
10    DiscretiseWeave()
11 else if (status = INITIALCOLLISION)
12    status = BISECTION
13    DiscretiseWeave()
    
```

The ConstraintIteration algorithm is called for each step of geometric constraint resolution for the entire woven structure. The status variable is initialised to FIRSTITERATION for each point of interlacement. After line 1 detects the presence or absence of collision, there are four cases to consider. For case 1 (lines 2-3) there is no collision on the first iteration and therefore the algorithm advances to the next interlacement point. Fig. 3 (a) shows an example of no interpenetration at an interlacement point. Cases 2 and 3 (lines 4-10) perform collision response. Elimination of yarn interpenetration (case 3 lines 7-10) flattens yarns prior to interval bisection. Interval bisection ensures yarns are just touching (case 2 lines 4-6) and are neither interpenetrating nor free-floating. Fig. 3 (b) shows an example of yarn interpenetration at an interlacement point. Overlapping (i.e. intersecting) triangles at the contact zone of the weft yarn (running from lower-left to upper-right) are shown. Case 4 (lines 11-13) occurs when interpenetration at the point of interlacement is eliminated. Subsequent calls to ConstraintIteration ensure calls to UpdateCollisionBounds (case 3) are replaced with calls to UpdateBisectionBounds (case 2).

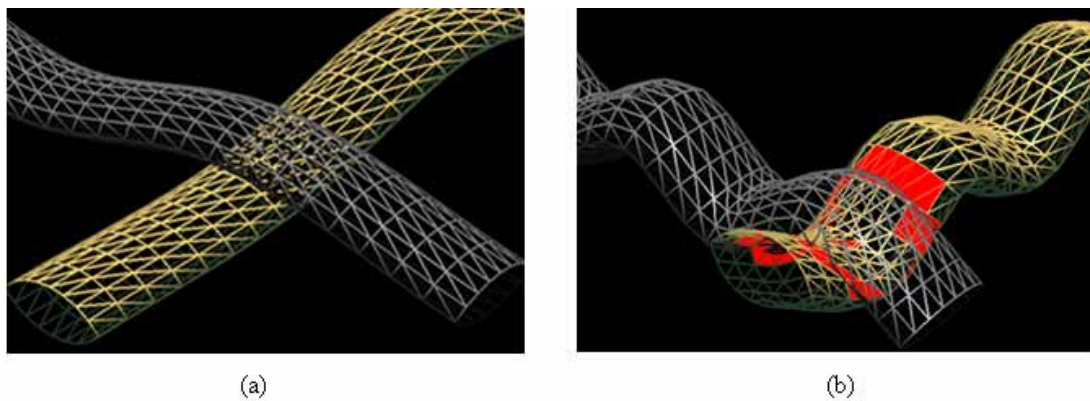


Fig. 3. Screenshots from WeaveStudio CAD/CAM software. Low crimp resulting in no yarn interpenetration (a) and high crimp resulting in yarn interpenetration (b).

4.2. Hierarchical Collision

The algorithm to resolve yarn interpenetration consists of three main phases: collision detection, collision determination and collision response. The first two phases are described in this subsection and address if and where a collision occurs, respectively. The collision detection and determination phases check for collision at the points of interlacement in the weave. Yarns are assumed not to self-intersect and distances between adjacent yarns are such that warp yarns do not collide with warp yarns and weft yarns do not collide with weft yarns. The final collision response phase eliminates yarn interpenetration and is described in the next subsection.

The geometric model of each yarn in the weave is enclosed by a hierarchical binary tree structure. Each node in the tree has a bounding volume that fully encloses the geometry in its subtree (node_volume in the pseudocode where node is either warp or weft). The root node is the topmost node and refers to a left child and a right child (node_handednesschild in the pseudocode where handedness is either left or right). Internal nodes also refer to left and right child nodes. Each (childless) leaf node contains the yarn triangle geometry (node_triangle in the pseudocode). For a binary axis-aligned bounding box (AABB) tree used in the implementation, each node's bounding volume is an AABB and is defined by two 3D position vectors to represent the AABB extremities.

$$\mathbf{v}_i^{\min} \leq \mathbf{v}_i^{\max}, \forall \in \{x, y, z\} \quad (19)$$

The AABB for the root node is formed by iterating over all the triangle vertices for the yarn and adjusting \mathbf{v}_i^{\min} and \mathbf{v}_i^{\max} to enclose the extents of the vertices. The root node's AABB is then subdivided into two and the two new nodes are referred to as the left and right child of the root node. The tree construction continues recursively until a stopping criterion is reached. There are a variety of schemes for deciding on when to terminate subdivision and where to divide the AABB [26]. The implementation uses common heuristics for AABB subdivision [26] by subdividing the AABB along its longest axis, and choosing the midpoint as the split point. After subdivision, each triangle is assigned to one of the AABBs of the two new nodes depending on which AABB contains its centroid. Subdivision terminates when the number of triangles in a (leaf) node falls below an implementation defined threshold.

```

bool HierarchicalCollision()
1     trianglelist =  $\emptyset$ 
2     if (status == INITIALISE)
3         InitialiseToFirstInterlacementPoint()
4         status = FIRSTITERATION
5     TestHierarchies(warpi, weftj)
6     if (trianglelist =  $\emptyset$ ) return false else return true

TestHierarchies(warp, weft)
1     if (IsLeaf(warp) and IsLeaf(weft))
2         for each warp_trianglei  $i \in \{1, 2, \dots, m\}$ 
3             for each weft_trianglej  $j \in \{1, 2, \dots, n\}$ 
4                 if (overlap(warp_trianglei, weft_trianglej))
5                     Insert(trianglelist, warp_trianglei, weft_trianglej)
6     else if (IsNotLeaf(warp) and IsNotLeaf(weft))
7         if (overlap(warp_volume, weft_volume))
8             TestHierarchies(warp_leftchild, weft)
9             TestHierarchies(warp_rightchild, weft)
10    else if (IsLeaf(warp) and IsNotLeaf(weft))
11        TestHierarchies(warp, weft_leftchild)
12        TestHierarchies(warp, weft_rightchild)
13    else
14        TestHierarchies(warp_leftchild, weft)
15        TestHierarchies(warp_rightchild, weft)

```

The HierarchicalCollision algorithm returns a Boolean value. True if trianglelist is populated with any triangles resulting from collision at the yarn interlacement point. An empty trianglelist generates a false return value. The return value from HierarchicalCollision indicates the status of collision detection, the trianglelist contains triangles pertaining to collision determination. HierarchicalCollision calls the recursive TestHierarchies algorithm with the root nodes of the AABB trees for a single warp_i and a single weft_j yarn. It should be noted that the use of an underscore in TestHierarchies denotes access to tree node attributes (i.e. access to a node's child node, bounding volume or triangles). TestHierarchies populates trianglelist with triangles from yarn intersections. There are four cases to consider. For case 1 (lines 1-5) both warp and weft nodes are leaf nodes. Overlapping triangles are inserted into trianglelist. Case 2 (lines 6-9) both warp and weft nodes are non-leaf (i.e. internal) nodes. If the volumes (i.e. AABBs) for warp and weft nodes overlap, TestHierarchies is called recursively. Case 3 (lines 10-12) and case 4 (lines (13-15) call TestHierarchies recursively when one node is a leaf node and the other node is a non-leaf node. For the triangle overlap test the efficient interval overlap method is used [26]. The overlap test between two AABBs is trivial. From Eq. (19) each vector component $warp_i^{\min}$ of the warp node's AABB is compared with the corresponding vector component $weft_i^{\max}$ of the weft node's AABB. If $warp_i^{\min} > weft_i^{\max}$ or $warp_i^{\max} < weft_i^{\min}$ the AABBs are disjoint, otherwise they overlap.

4.3. Bounds Update for Collision Response

Collision response as part of the bounds update is empirically derived. Experimental observations show yarn cross-sections may be lenticular [27]. However, during weaving the yarn is deformed. Inter-thread pressures set-up during weaving cause considerable yarn flattening normal to the plane of the woven structure [14]. At the points of interlacement, compressive and bending stresses contribute to deformation. Drasarova [28] conducted image analysis of textile structures for finished weaves in a relaxed state. Many parameters were varied including weave type, staple and multifilament yarns, material, fineness, twist and sett. Yarn flattening occurred in all weaves at points of interlacement. In particular, a compression and lateral expansion were observed relative to the original yarn dimensions. To geometrically model and simulate yarn deformation, the yarn cross-section is deformed at the contact zone to simulate yarn flattening and to

eliminate yarn interpenetration. Analogous to Poisson’s ratio [23], the deformation can be modelled to affect compression along the binormal vector \mathbf{b} (Eq. (14)) and a proportional lateral expansion along the normal vector \mathbf{n} (Eq. (13)). In addition to the yarn path centreline, piecewise Hermite curves can also be used to model irregular yarn cross-sections (see Fig. 2). From experimental observations [27] lenticular geometry is initially assumed prior to deformation. The circular arc used in lenticular geometry can be approximated with a Hermite curve.

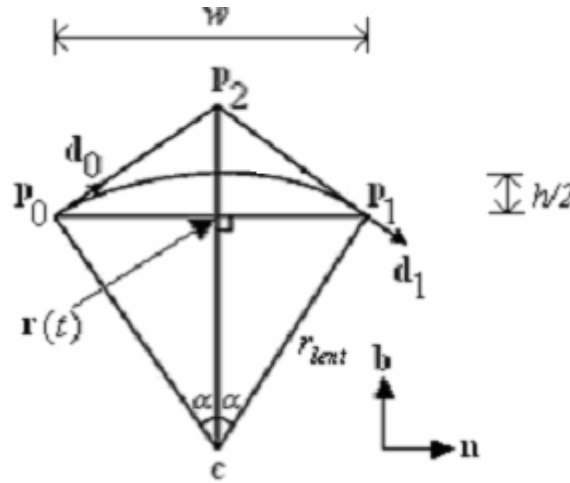


Fig. 4. Vector geometry of Hermite approximation to a circular arc.

Fig. 4 shows the vector geometry of a circular arc. p_0 and p_1 are position vectors of points on the circular arc with radius r_{lent} and centre c . d_0 and d_1 are unit tangent vectors at p_0 and p_1 respectively. p_2 is the point of intersection of the tangent lines passing through p_0 and p_1 and parallel to their respective unit tangent vectors d_0 and d_1 . Angle 2α in radians is the angle subtended by the circular arc. It can be shown [29] that the magnitude of the tangent vectors for Hermite approximation of a circular arc is

$$u = \frac{4r_{lent}(1 - \cos(\alpha))}{\sin(\alpha)} \tag{20}$$

Reducing the magnitude u of the tangent vectors reduces the arc curvature and flattens the yarn cross-sectional shape. The equations of the intersection point p_2 , the Hermite curve endpoints p_0 and p_1 and unit tangent vectors d_0 and d_1 are

$$\mathbf{p}_2 = \mathbf{r}(t) + (r_{lent} \sin(\alpha) \tan(\alpha))\mathbf{b} \tag{21}$$

$$\mathbf{p}_0 = \mathbf{r}(t) - \left(\frac{w}{2}\right)\mathbf{n} \tag{22}$$

$$\mathbf{p}_1 = \mathbf{r}(t) + \left(\frac{w}{2}\right)\mathbf{n} \tag{23}$$

$$\mathbf{d}_0 = u \left(\frac{\mathbf{p}_2 - \mathbf{p}_0}{|\mathbf{p}_2 - \mathbf{p}_0|} \right) \tag{24}$$

$$\mathbf{d}_1 = u \left(\frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|} \right) \tag{25}$$

UpdateCollisionBounds()

```

1   for each yarnidi i ∈ Y where Y is the set of yarn indices at the interlacement point. |Y| = 2.
2       for each segmentidj j ∈ A where A is the set of colliding segment indices for yarnidi;
3           if (status = FIRSTITERATION)
4               cap_upperbound = u
    
```

```

5         cap_lowerbound = u × (1 - ε)
6         insert(yarnidxi, segmentidxj, cap)
7         insert(yarnidxi, segmentidxj+1, cap)
8     else
9         cap = find(yarnidxi, segmentidxj)
10        if (cap_updated = false)
11            cap_lowerbound = cap_lowerbound - u × ε
12            cap_updated = true
13        cap = find(yarnidxi, segmentidxj+1)
14        if (cap_updated = false)
15            cap_lowerbound = cap_lowerbound - u × ε
16            cap_updated = true

```

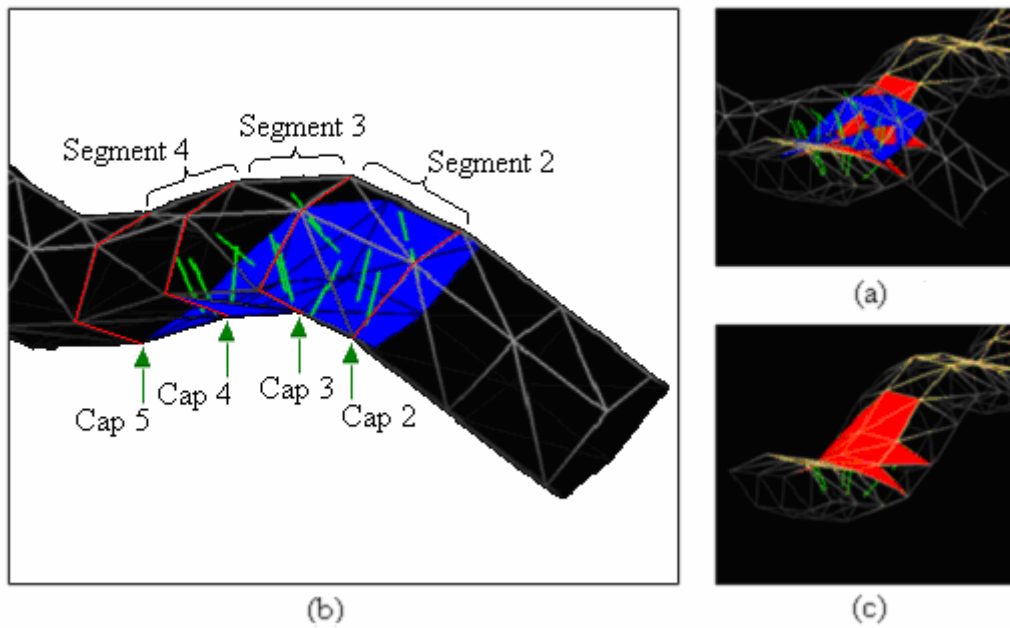


Fig. 5. Screenshots from WeaveStudio CAD/CAM software. Discrete yarn geometry of interpenetration at an interlacement point (a) and enlarged warp yarn (b) and weft yarn (c).

Line 2 of UpdateCollisionBounds iterates over the set A of all the colliding segment indices at the point of interlacement for yarnidx_i . UpdateCollisionBounds is effectively a hunting phase to determine the lower bounds of collision intervals prior to bisection. At a lower level of abstraction (not shown in the pseudocode) the colliding yarn segment indices are obtained from trianglelist. The trianglelist contains all the overlapping triangles at the interlacement point and all triangles are assigned with the index of both the yarn and yarn segment (yarnidx and segmentidx respectively) they are associated with. Fig. 5 (a) visualises trianglelist for interpenetration at an interlacement point. Fig. 5 shows a low-resolution mesh for illustrative purposes. Fig. 5 (b) shows a magnified view of the warp yarn and associated colliding segments, caps and triangles. Fig. 5 (c) shows the weft yarn and associated colliding triangles. Also shown in Fig. 5 are inward facing surface normal vectors for colliding triangles. Positive and negative y -components of the normals result in yarn flattening at the bottom and top respectively. There are two cases to consider. For case 1 (lines 3-7) the first iteration of the algorithm ensures the pair of caps at both ends of each colliding segment is inserted into a data structure. Fig. 5 (b) shows caps 2-5 for insertion associated with segments 2-4. The underscore convention in the pseudocode is maintained and denotes access to cap attributes. The upper bound of the cap (cap_upperbound) is assigned the value u from Eq. (20). The lower bound (cap_lowerbound) is assigned a value fractionally smaller than cap_upperbound . The data structure used for cap insertion and retrieval is the Standard Template Library map [30]. Each yarnidx_i has a map that permits storage of key-value pairs with $O(\log n)$ insertion and retrieval via the unique key segmentidx_j or segmentidx_{j+1} . The map requires unique

keys hence lines 6-7 do not insert duplicate caps. For subsequent iterations, Case 2 (lines 8-16) decrements the lower bound (`cap_lowerbound`) for each cap. Each cap's update status (`cap_updated`) is tested to ensure each cap's lower bound (`cap_lowerbound`) is decremented once per (subsequent) iteration. The update status for all caps is initially false. When interpenetration at the point of interlacement is eliminated (lines 11-13 of `ConstraintIteration`) the bounds of each pair of caps corresponding to each segment bracket the collision intervals. Each collision interval is $I = [l, u]$ where l (`cap_lowerbound`) is the lower bound obtained from `UpdateCollisionBounds` and u (`cap_upperbound`) is from Eq. (20). The interval contains a real number $v \in I$ that is substituted for u in Eqs. (24-25) to ensure yarns are just touching. An approximation to v is obtained by the numerical bracketing algorithm `UpdateBisectionBounds`.

Fig. 6 shows the `yarnidxi` indexing scheme for a plain weave with eight yarns and sixteen interlacement points. An example indexing order to iterate over all sixteen points of interlacement vertically from lower-left to upper-right is (0, 4)(0, 5)(0, 6)(0, 7)(1, 4)(1, 5)(1, 6)(1, 7)(2, 4)(2, 5)(2, 6)(2, 7)(3, 4)(3, 5)(3, 6)(3, 7).

`UpdateBisectionBounds()`

```

1      for each yarnidxi  $i \in Y$  where  $Y$  is the set of yarn indices at the interlacement point.  $|Y| = 2$ .
2          for each segmentidxj  $j \in B$ .  $B$  is the set of initial colliding segment indices for yarnidxi
3              segment = find(yarnidxi, segmentidxj)
4              cap = find(yarnidxi, segmentidxj)
5              if (cap_updated = false)
6                  if (segment_collision)
7                      cap_upperbound = (cap_lowerbound + cap_upperbound) / 2
8                  else
9                      cap_lowerbound = (cap_lowerbound + cap_upperbound) / 2
10                     cap_updated = true
11                 cap = find(yarnidxi, segmentidxj+1)
12                 if (cap_updated = false)
13                     if (segment_collision)
14                         cap_upperbound = (cap_lowerbound + cap_upperbound) / 2
15                     else
16                         cap_lowerbound = (cap_lowerbound + cap_upperbound) / 2
17                     cap_updated = true

```

Line 2 of `UpdateBisectionBounds` iterates over the set B of all the initial colliding segment indices at the point of interlacement for `yarnidxi`. Let $B = A$ when status is `FIRSTITERATION` in `UpdateCollisionBounds`. Each collision interval is successively bisected (lines 7, 9, 14 and 16) resulting in a new subinterval. Calls to `UpdateBisectionBounds` terminate when the magnitude of the bracketing subinterval is less than a user specified tolerance $|b_k - a_k| < \tau$ where $[a_k, b_k]$ is the corresponding closed subinterval for the k th iteration and $\tau > 0$ is the tolerance. On termination, yarn segments in `yarnidxi` (warp) are just touching yarn segments in `yarnidxi` (weft). The notion of 'just touching' is subject to the limitations of floating-point precision and approximation of the numerical bracketing method.

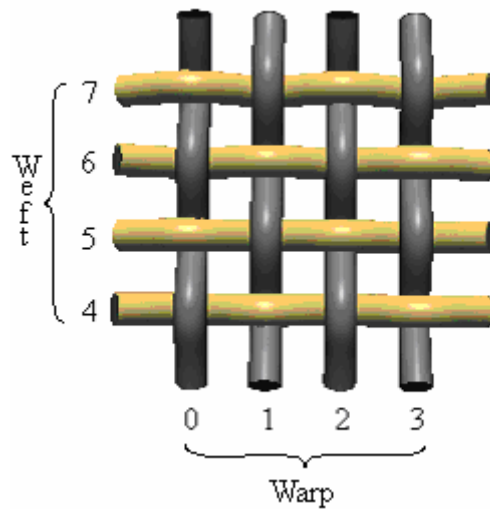


Fig. 6. Screenshot from WeaveStudio CAD/CAM software showing yarn indexing scheme.

4.4. Weave Discretisation

The region of space occupied by the yarn surfaces is discretised into finite triangular elements for several reasons: Firstly, a mesh of finite triangular elements simplifies the task of collision detection and determination due to known and efficient triangle intersection testing algorithms [26]. Secondly, computations for collision detection and determination are independent of the underlying representation of yarn path centreline or cross-sectional shape. As a consequence, this phase of the collision handling is widely applicable to existing triangular geometric models. Furthermore, specialised and complex collision handling specific to various surface representations is avoided. Thirdly, a finite discrete approximation is essential for the finite element method. As a result, descriptive geometric modelling described in this paper is extensible to incorporate predictive finite element analysis. Finally, triangle meshes are used for graphical rendering and visualisation. Graphics performance is particularly important for the interactive visualisation of large, complex high-resolution woven structure models. The trend is for graphics accelerators to take advantage of meshes as much as possible. Essentially, it is more efficient to transform a vertex only once. Triangle strips and fans allow some data sharing, but triangle meshes allow full sharing [26].

Fig. 7 shows a low-resolution mesh for illustrative purposes. A zero-based indexing scheme is employed to identify each yarn and constituent yarn segments, triangles and caps. All indices are unique except for the dual triangle index at the start of each successive cap (for ease of implementation). For clarity, only the first nine vertices are labelled and only one Hermite curve for one cap is shown. For collision handling, each triangle is assigned with the indices of its containing yarn and yarn segment. For example, the filled triangle (indices 2, 9 and 8) in Fig. 7 has indices of zero and zero for both the yarn index and segment index.

DiscretiseWeave iterates over all yarns, yarn segments and vertices to tessellate the yarn surfaces. The pseudocode generates a triangular mesh composed of indices into a vertex array. Each set of three consecutive indices represents a triangle. Much of the pseudocode is book-keeping to drive the mesh construction. The computations deserving particular attention are lines 7, 15 and 17. Eqs. (12-14) represent the construction of the orthonormal basis (line 7). Assignment of vertex (line 15) is achieved by Substituting Eqs. (22-25) into Eq. (7) to generate 3D position vectors for the triangles. BuildBinaryTree (line 17) is described in Section 4.2 Hierarchical Collision.

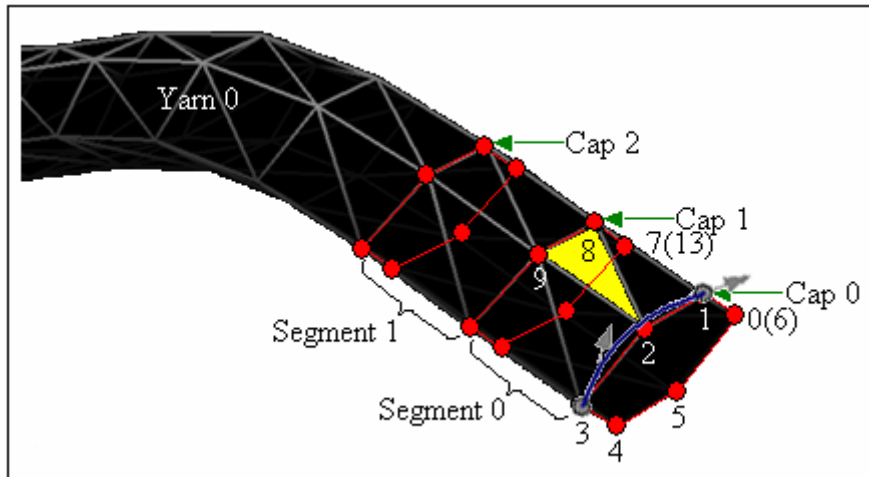


Fig. 7. Screenshot from WeaveStudio CAD/CAM software showing yarn geometry after discretisation.

```

DiscretiseWeave()
1  for each warpi  $i \in \{1, 2, \dots, m\}$ 
2      for each weftj  $j \in \{1, 2, \dots, n\}$ 
3           $ibaseidx = 0; vbaseidx = 0$ 
4          for each segmentk  $k \in \{1, 2, \dots, p\}$ 
5               $i0 = ibaseidx; i1 = i0 + 1; vbaseidxcpy = vbaseidx$ 
6               $ibaseidx = ibaseidx + r + 1; i2 = ibaseidx; i3 = i2 + 1$ 
7              BuildOrthonormalBasis(t, n, b)
8              for each vertexl  $l \in \{1, 2, \dots, r\}$ 
9                  if ( $k \neq p$ )
10                      $indices[ibaseidx] = i0; indices[ibaseidx + 1] = i1$ 
11                      $indices[ibaseidx + 2] = i2; indices[ibaseidx + 3] = i3$ 
12                      $indices[ibaseidx + 4] = i4; indices[ibaseidx + 5] = i5$ 
13                      $i0 = i0 + 1; i1 = i1 + 1; i2 = i2 + 1; i3 = i3 + 1$ 
14                      $ibaseidx = ibaseidx + 6$ 
15                      $vertices[vbaseidx] = \mathbf{vertex}; vbaseidx = vbaseidx + 1$ 
16                      $vertices[vbaseidx] = vertices[vbaseidxcpy]; vbaseidx = vbaseidx + 1$ 
17             BuildBinaryTree()

```

5. Efficiency and Accuracy of Algorithms

The hierarchical collision algorithm has been validated by substituting a brute-force collision algorithm. Test data confirm that the triangle list (i.e. triangle list) contains the same triangles independent of the collision algorithm (brute-force or hierarchical) thus validating the accuracy of the hierarchical collision detection and determination. The empirical collision response phase is a geometric approximation of reality and is based on image analysis of textile structures [28]. The collision handling algorithms described in this paper perform more efficiently than those proposed by other researchers. The binary AABB tree benefits from $O(\log n)$ search time where n is the number of AABBs. In addition, the overlap test for an AABB is simple and consists of up to six coordinate comparisons. Furthermore, the AABB provides a tight fit for the axis-aligned geometry of the 2D weaves. In contrast, Sherburn [31] uses brute-force collision tests between all n triangles in the model and the performance is $O(n^2)$. Furthermore, the interval overlap method outperforms the overlap test [26] used by Sherburn [31]. Barauskas and Kuprys [32] improve on Sherburn's performance by using oriented-bounding boxes (OBBs). However, no hierarchy was used and the performance cost is $O(n^2)$ where n is the number of OBBs. In addition, the performance cost of the OBB overlap test is more expensive than the AABB overlap test [26].

6. Results and Discussion

Fig. 8 shows comparisons between 3D visualisations of woven textile structure geometry.

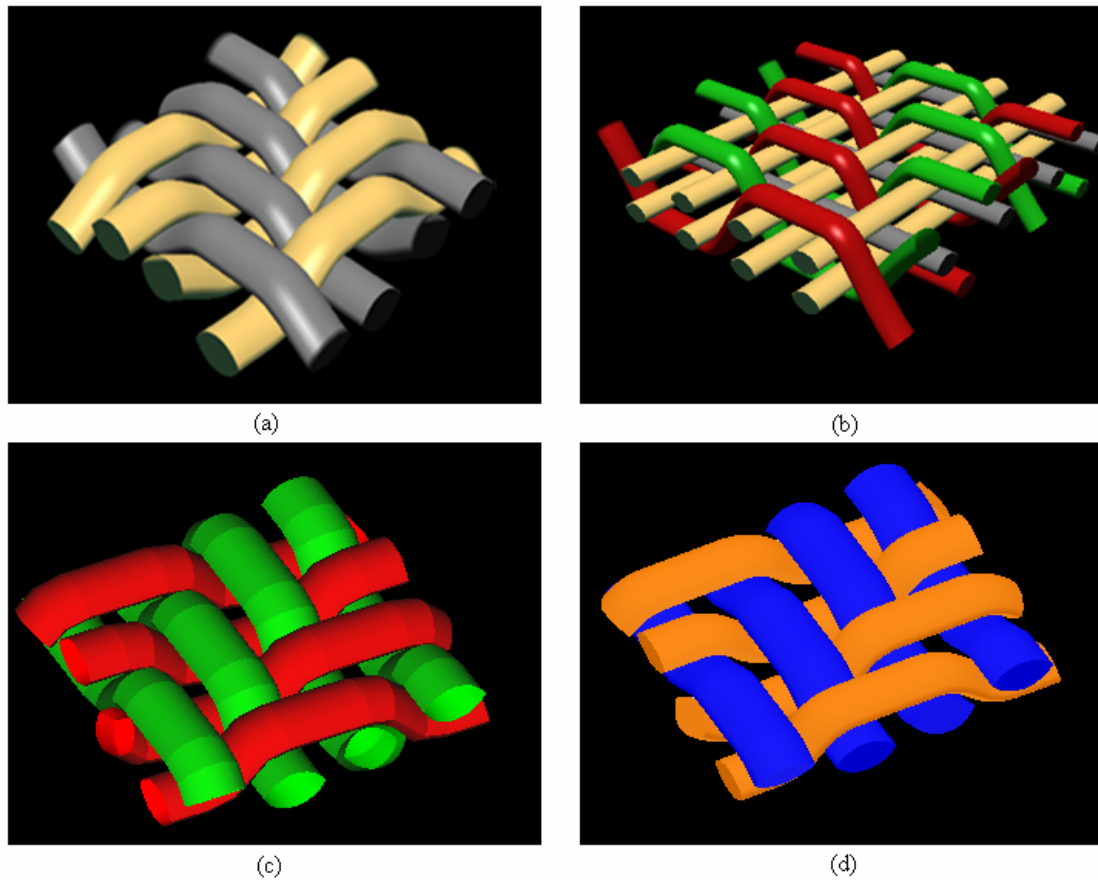


Fig. 8. Screenshots from WeaveStudio CAD/CAM software (a) and (b), Jiang and Chen model (c) and Weave GeoModeller (d).

Fig. 8 (a) and Fig. 8 (b) are screenshots from the WeaveStudio CAD/CAM software tool [21]. Fig. 8 (c) and Fig. 8 (d) are screenshots showing the Jiang and Chen [19] model and Weave GeoModeller [33] respectively. Fig. 8 (a), Fig. 8 (c) and Fig. 8 (d) show a 2/2 twill weave with lenticular yarn cross-sections and warp and weft density of four ends/cm and four picks/cm respectively. It can be seen that the modelling techniques described in this paper – in particular yarn compression and interference resolution – result in improved fabric geometry. Fig. 8 (b) shows an enhanced 3D orthogonal woven structure with lenticular cross-sections. The dual binding warps are in a 2/2 twill weave formation and the structure includes four straight warps, eight straight wefts and warp and weft density of 14 ends/cm and 6 picks/cm respectively. 3D woven structures are not supported by the Jiang and Chen [19] model or Weave GeoModeller [33] and therefore comparisons were not possible for 3D weaves.

7. Conclusions

The constraint-based geometric modelling has been implemented in the C++ programming language as part of the bespoke WeaveStudio CAD/CAM software tool [21]. The general nature of the modelling approach is capable of parametrically describing a wide variety of woven structures irrespective of dimension, interlacement or yarn cross-sectional shape. Flexibility to model a wide variety of yarn cross-sections in addition to that shown in Fig. 2 is inherent in the parametric representation. The computationally efficient collision handling is an improvement over the performance of existing approaches. In particular, the geometric constraint algorithms eliminate yarn interpenetration resulting in a truer representation of weave geometry. Consequently, improved accuracy in predictive mechanical modelling is expected based on the new geometric approach. The described geometrical modelling is extensible to include two existing mechanical modelling approaches. Firstly, the energy-based approach can assume various shapes for yarn path and cross-sections as long as it is possible to calculate geometric parameters such as curvature and arc

length etc. [9]. Geometric parameters for the described approach are readily available. Secondly, the finite elements used in the constraint-based modelling can be extended to allow finite element simulation. Investigations are underway to implement and evaluate a unified geometric-mechanical approach based on the geometric modelling described and the compatible mechanical models.

8. References

- [1] B. Olofsson. A general model of fabric as a geometric-mechanical structure, *J.Text. Inst.* 1964 , **55**: 541-557.
- [2] S. Kawabata, M. Niwa, H. Kawai. The finite-deformation theory of plain weave fabrics-Part I: The biaxial deformation theory, *J. Text. Inst.* 1973, **64**: 21-46.
- [3] S. Kawabata, M. Niwa, H. Kawai. The finite-deformation theory of plain weave fabrics-Part II: The uniaxial deformation theory, *J. Text. Inst.* 1973, **64** : 47-61.
- [4] P. Grosberg, S. Kedia. The mechanical properties of woven fabrics-Part I: The initial load extension modulus of woven fabrics, *Text. Res. J.* 1966, **36** : 71-79.
- [5] S. De Jong, R. Postle. An energy analysis of woven fabric mechanics by means of optimal control theory-Part I: Tensile properties, *J. Text. Inst.* 1978,**68** : 350-361.
- [6] S. De Jong, R. Postle. An energy analysis of woven fabric mechanics by means of optimal control theory-Part II: Pure bending properties, *J. Text. Inst.* 1978, **68**.
- [7] J. W.S. Hearle, W. J. Shanahan. An energy method for calculations in fabric mechanics-Part I: principles and methods, *J. Text. Inst.* 1978, **69** .
- [8] J. W.S. Hearle, P. Potluri. V. S. Thammandra. Modelling fabric mechanics, *J. Text. Inst.* 2001, **92** .
- [9] T. V. Sagar, P. Potluri, J. W. S. Hearle. Mesoscale modelling of interlaced fibre assemblies using energy method, *Comput. Mater. Sci.* 2003, **28**: 49-62.
- [10] F. T. Peirce. The geometry of cloth structure, *J. Text. Inst.* 1937, **28**: 45-96.
- [11] A. Kemp. An extension of Peirce's cloth geometry to the treatment of nonlinear threads, *J. Text. Inst.* 1958, **49**: 44-48.
- [12] W. J. Shanahan, J. W. S. Hearle. An energy method for calculations in fabric mechanics-Part II: Examples of applications of the method to woven fabrics, *J. Text. Inst.* 1978, **69**: 92-100.
- [13] F. Robitaille, A. C. Long, I. A. Jones, C. D. Rudd. Automatically generated geometric descriptions of textile and composite unit cells, *Compos. Part A.* 2003, **34**: 303-312.
- [14] J. Hu. *Structure and mechanics of woven fabrics*. Cambridge: Woodhead Publishing. 2004.
- [15] H. Y. Lin, A. Newton. Computer representation of woven fabric by using B-splines, *J. Text. Inst.* 1999, **90**: 59-72.
- [16] T. Liao, S. Adanur. A novel approach to three-dimensional modeling of interlaced fabric structures, *Text. Res. J.* 1998, **68**: 841-847.
- [17] S. Adanur, T. Liao. 3D modelling of textile composite performs, *Compos. Part B* 1998, **29B**: 787-793.
- [18] C. L. Ma, J. M. Yang, T.W. Chou. Elastic Stiffness of three-dimensional braided textile structural composites, *Composite Materials: Testing and Design (7th Conference)*, J.M. Whitney, Ed., ASTM STP 893, Philadelphia 1986: 404-421.
- [19] Y. Jiang, X. Chen. Geometric and algebraic algorithms for modelling yarn in woven fabrics, *J. Text. Inst.* 2005 , **96**: 237-245.
- [20] J. Foley, A. van Dam, S. Feiner, J. Hughes J. *Computer Graphics: Principles and Practice, 2nd Ed.*, Addison-Wesley, 1996.
- [21] M. Smith, X. Chen. WeaveStudio CAD/CAM Software for 3D Woven Structures, EPSRC funded project, University of Manchester, UK, 2005-2008.
- [22] G. Farin. *Curves and Surfaces for Computer Aided Geometric Design, 4th Ed.*, Academic Press, 1997.
- [23] J. Gere. *Mechanics of Materials, 5th SI Ed.*, Nelson Thornes, Cheltenham, 2002.

- [24] E. Catmull, R. Rom. A class of local interpolating splines, *Computer Aided Geometric Design*, Academic Press, San Francisco, 1974.
- [25] D. Salomon. *Computer Graphics and Geometric Modeling*, Springer, New York, 1999.
- [26] T. Akenine-Moller, E. Haines. *Real-Time Rendering, 2nd Ed.* A K Peters, Massachusetts, 2002.
- [27] G. Hivet, P. Boisse. Consistent 3D geometrical model of fabric elementary cell. Application to a meshing preprocessor for 3D finite element analysis. *Finite Elements in Analysis and Design*. 2005, **42**: 25-49.
- [28] J. Drasarova. Application of image analysis of investigation of textile structures, Technical University of Liberec, Dept. of Textile Structures, Czech Republic, 2003.
- [29] M. E. Mortenson. *Geometric Modeling, 3rd Ed.*, Industrial Press Inc., New York, 2006.
- [30] N. M. Josuttis. *The C++ Standard Library: A Tutorial and Reference*, Addison-Wesley Longman Inc., 1999.
- [31] M. Sherburn, F. Robitaille, A. Long, C. Rudd. Geometric pre-processor for the calculation of physical properties of textiles, *Industrial Simulation Conference Proceedings*, Malaga, Spain, 2004, 479-486.
- [32] R. Barauskas, M. Kuprys. Collision handling of fabric yarns in woven structures, *Information Technology and Control*, 2005, **34**: 318-326.
- [33] Weave GeoModeller. TexEng Software Ltd., University of Manchester, UK, <http://www.texeng.co.uk/weavegeomodeller.html>.