3D Printable Vascular Networks Generated by Accelerated Constrained Constructive Optimization for Tissue Engineering

Andrew A. Guy, Alexander W. Justin, Dulce M. Aguilar-Garza and Athina E. Markaki

***Abstract*— One of the greatest challenges in fabricating artificial tissues and organs is the incorporation of vascular networks to support the biological requirements of the embedded cells, encouraging tissue formation and maturation. With the advent of 3D printing technology, significant progress has been made with respect to generating vascularized artificial tissues. Current algorithms to generate arterial/venous trees are computationally expensive and offer limited freedom to optimize the resulting structures. Furthermore, there is no method for algorithmic generation of vascular networks that can recapitulate the complexity of the native vasculature for more than two trees, and export directly to a 3D printing format. Here, we report such a method, using an accelerated constructive constrained optimization approach, by decomposing the process into construction, optimization and collision resolution stages. The new approach reduces computation time to minutes at problem sizes where previous implementations have reported days. With the optimality criterion of maximizing the volume of useful tissue which could be grown around such a network, an approach of alternating stages of construction and batch optimization of all node positions is introduced and shown to yield consistently more optimal networks. The approach does not place a limit on the number of interpenetrating networks that can be constructed in a given space; indeed we demonstrate a biomimetic, liver-like tissue model. Methods to account for the limitations of 3D printing are provided, notably the minimum feature size and infill at sharp angles, through padding and angle reduction, respectively.**

***Index Terms*— Constrained Constructive Optimization, vascular networks, tissue engineering, 3D printing**

AA Guy is supported by an EPSRC Doctoral Training Partnership Award (EP/N509620/1). AW Justin is supported by EPSRC (EP/R511675/1 & EP/N509620/1), the Isaac Newton Trust and the Rosetrees Trust (M787). DM Aguilar-Garza is supported by The Cambridge Trust, CONACyT (Mexico) and the EPSRC Cambridge & Cranfield Doctoral Training Centre in Ultra Precision (EP/K503241/1). Corresponding author: Athina E. Markaki (email: am253@cam.ac.uk).

AA Guy, AW Justin, DM Aguilar-Garza and AE Markaki are with the Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, U.K.

AAG conceived of and executed the development and analysis of the code.

I. Introduction

VASCULARIZATION is one of the key challenges for engineering new tissues and organs in the lab [1]. Tissue constructs without a perfusable vascular network lack a mechanism for supplying cells with nutrients and removing CO_2 and cellular waste; for this supply mechanism to be efficient, it is important that the network displays hierarchy [2], [3]. By incorporating a vascular network into a cell-ladened scaffold, tissues with clinically relevant dimensions can be developed and maintained, which could then be used as a replacement for a diseased tissue. Furthermore, tissues and organs consist of multiple interpenetrating tubular systems, including those specific to the tissue function, such as the biliary system in the liver. Until such complex systems are constructed in the lab, fabrication of engineered tissue equivalents for complex organs (e.g. liver, kidney) will remain a formidable challenge.

By algorithmically generating biomimetic and architecturally complex vascular network models, one can rapidly produce vascular networks capable of uniformly supporting high densities of metabolically active cells in a 3D environment. Such models can be used to simulate fluid flow (e.g. wall shear stresses, velocity field), metabolite diffusion, and the cellular environment. Furthermore, template structures from these models can be fabricated via 3D printing techniques (e.g. inkjet, extrusion-based, stereolithography) and incorporated into tissue engineered scaffolds to yield perfusable channels [4]. Such approaches involve a solid sacrificial template around which biomaterials (frequently loaded with cells) are cast. These sacrificial structures are removed (e.g. chemically, thermally, mechanically) to produce a vascular network in the biomaterial with channels of the same diameter as the 3D printed feature.

Since manually constructing vascular structures using computer aided design (CAD) packages is unfeasible for the level of complexity required, procedural generation of vascular networks is greatly preferable. Other applications for such models include comparisons of diseased and physiological tissues [5], surgical planning [6], [7], and the validation of vessel segmentation algorithms [8]–[10].

A number of algorithms have been developed over the past 25 years for arterial tree representation, considering the physiological constraints associated with blood flow and the hierarchical vasculature in the human body. The most common

approach employed to produce realistic vascular networks involves iterative growth [8], [10], [11]. Constrained Constructive Optimization (CCO) is a widely accepted method for the generation of vascular trees [12]–[17]. Other approaches include the Lindenmayer systems (L-systems) [9], [18], Global Constructive Optimization [19] and the Random Walk Algorithm [20]. In order to build a physiologically relevant arterial tree, an optimality criterion must be fulfilled. The principle of minimum intra-vascular volume is most commonly used to ensure an optimal build [8], [11]–[14], [16]–[19]. In addition, several assumptions are frequently applied to simplify these models. The most common include laminar (steady-state) flow, flow conservation, equal terminal flow and pressure, Newtonian fluid flow, and Murray's law for optimal bifurcations; networks that conform to this law maximize flow conductance per unit volume [2]. Some algorithms simulate additional phenomena such as the Fåhræus-Lindqvist effect (viscosity changes with vessel diameter) [15], [20] and vascular growth driven by oxygen demand [8], [20]. Most algorithms are used to produce single vascular trees [8]–[10], [12], [13], [17]–[19], [21], while only a few simulate full vascular networks [11], [15] or include capillaries [16], [20].

Here, we describe a new CCO approach, capable of producing multiple and independent interpenetrating networks (e.g. arterial, venous, epithelial, lymphatic). The networks bifurcate from large vessels down to fine vessels and anastomose back to large-sized vessels. The model allows for any volumetric region to be uniformly supported with vessels and the spacing between the vessels is also controllable. Thus, multiple independent networks can be packed into the space. Further, our approach is computationally efficient, and engages with vessel collision detection after the vascular tree has been formed. Alongside physical flow constraints, such as pressure and flow rate, our approach incorporates a number of biomimetic principles, including Murray's law, which relates the parent and daughter branch diameters, and minimal network volume considerations.

Section II gives a brief overview of the fluid mechanics and physiological laws underlying CCO, and the essential inputs required to create a network. Section III outlines the novel methods implemented, and presents an argument for their efficiency; this is investigated in detail in Section IV, alongside some demonstrative examples. The mathematical notation used in this paper and parameters included in the model are summarised in Table I.

II. Constrained Constructive Optimization

The required input parameters (those which do not have default values) are listed in Table II. We follow the established terminology of Schreiner [13] and refer to nodes in a tree with no children as *terminals*, and the node which starts the root vessel as the *source*. We used a constant-viscosity model, as the vessel sizes we are currently able to manufacture ($r \geq 125$ µm) are above the radius at which viscosity correction has previously been applied [14]. Considering steady laminar flow, the pressure difference over a branch, ΔP, is given as

$$\Delta P = \frac{8\mu L}{\pi r^4} Q = RQ, \tag{1}$$

TABLE I
SUMMARY OF NOTATION

Physical and derived values		
μ	Dynamic viscosity	[kPa·s^{-1}]
γ	Murray's law exponent	[-]
x	Position vector in $\mathbb{R}^3$	[mm]
Q	Volumetric flow rate	[mm^3·s^{-1}]
P	Pressure	[kPa]
L	Branch length	[mm]
r	Branch radius	[mm]
R	Laminar flow resistance of a branch	[kPa·s·mm^{-3}]
R^*	Reduced resistance	[kPa·s·mm]
f	Child branch radius fraction of parent	[-]
Collision resolution		
	Calculated values	
s_r	Radial slenderness, L/r	[-]
χ	Branch length fraction of intersection	[-]
w	Weighting	[-]
$\hat{n}$	Unit normal vector in $\mathbb{R}^3$	[mm]
v_p	Perturbation vector in $\mathbb{R}^3$	[mm]
δ	Overlap between branches	[mm]
	Input parameters and default values	
k_r	Radial capture factor	1.2 [-]
k_a	Resolution aggression factor	1.5 [-]
C	Relative compliance of a network	1.0 [-]
$\lvert v\rvert_{\text{min}}$	Minimum perturbation distance	50 µm
z_p	Terminal node cull penalty	4 [-]
z_t	Terminal node cull threshold	20 [-]
β_c	Critical flow ratio for immediate culling	100 [-]
r^+	Radial padding to ensure separation	125 µm
r_{min}	Minimum feature size (radial)	125 µm
Optimization		
	Calculated values	
$\tilde{g}(x)$	Approximation to the direction of the gradient of the volume of the entire network with respect to the position, x, of a bifurcation	[mm^3·mm^{-1}]
$\nabla_\epsilon V(x)$	Finite-difference estimation of the volume gradient with respect to the position, x, of a bifurcation, with stride ϵ	[mm^3·mm^{-1}]
Δ	The stride length to take, having determined the direction: $x \leftarrow x - \Delta \cdot \tilde{g}(x)/\lvert\tilde{g}(x)\rvert$	[mm]
	Input parameters and default values	
η	Step length fraction	0.1 [-]
τ	Termination length fraction	0.2 [-]
β	Step fraction reduction ratio	0.5 [-]
z_b	Iterations between step fraction reduction	5 [-]
Construction		
S	Skip limit: the number of attempts allowed where no progress is made. The most 'lazy' selection is achieved with $S = 0$, and traditional CCO selection is with $S = \infty$.	
Computational scaling		
N	Total number of terminal nodes in the perfusion space.	
n	Number of terminal nodes currently in tree.	
$d(i)$	Terminal depth distribution: for a positive integer i, the fraction of terminal nodes which have i parent segments to the root node.	
Sets and indexing		
$\mathcal{B}$	All branches in the tree.	
$\mathcal{C}$	Children, the immediate downstream elements.	
$\mathcal{U}$	The chain of upstream elements to the root. Indexed as $\{1...M\}$, with 1 being the first child of the root and M being the immediate upstream element.	
$\mathcal{T}$	Bifurcation triad, the elements surrounding a bifurcation: $\mathcal{T} = \{\mathcal{C}, \mathcal{U}_M\}$.	
0	Subscript for the root branch/node.	
p	Subscript for the parent branch/node in a given context.	

TABLE II
THE ESSENTIAL INPUTS TO THE CCO ALGORITHM, AND THEIR UNITS

	Global	
ΔP_s	Pressure drop from source to terminals	kPa
x_s	Source node position	mm
	Per Terminal	
x	Position	mm
Q	Volumetric flow rate	$\text{mm}^3\cdot\text{s}^{-1}$

where L and r are the branch length and radius respectively, μ is the dynamic viscosity (taken to be 4×10^{-6} $\text{kPa}\cdot\text{s}^{-1}$ for blood), Q is the volumetric flow rate and R the laminar resistance. Physical constraints are provided by flow and pressure consistency between children ($\mathcal{C}$) and their parent (p):

$$Q_p = \sum_{i\in\mathcal{C}} Q_i, \tag{2}$$

$$P_p = P_i + Q_i R_i \quad \forall i \in \mathcal{C}. \tag{3}$$

In calculations, the reduced resistance, R^*:

$$R^* = Rr^4, \tag{4}$$

is used [12], as radii are only determined when needed.

The relationship between parent and child radii is fixed by Murray's law [2] to minimise power dissipation over the network:

$$r_p^\gamma = \sum_{i\in\mathcal{C}} r_i^\gamma, \tag{5}$$

where $\gamma = 3$, the standard result for minimum work, is the default value. Values in the range $2 \leq \gamma \leq 3$ are physiologically relevant, with lower exponents being observed in high-flow vessels. [3].

In this implementation, we use only nodes with at most two children (i.e. bifurcations), but nodes with higher splitting may be approximated by multiple close bifurcations with low separation. At each bifurcation, the child radii are set by their fraction, f, of the parent radius:

$$r_i = f_i r_p, \tag{6}$$

from which follows:

$$f_1^\gamma + f_2^\gamma = 1, \tag{7}$$

and, by dividing (5) by r_1^γ:

$$f_1 = \left[1 + \left(\frac{r_1}{r_2}\right)^{-\gamma}\right]^{-\frac{1}{\gamma}}, \tag{8}$$

with a similar form for f_2. By defining the terminal pressures to be uniform, we can determine the global resistance from the total flow rate, Q_0, and pressure drop from source to terminals, ΔP_s, allowing the root radius, r_0, to be calculated with (1) and (4):

$$r_0 = \left(\frac{R_0^*}{R_0}\right)^{\frac{1}{4}} = \left(\frac{Q_0 R_0^*}{\Delta P_s}\right)^{\frac{1}{4}}, \tag{9}$$

which is propagated down the tree using (6).

TABLE III
NODE TYPES, THEIR RELATIONSHIPS, AND DEPTH MODIFIERS.

Type	Parents	Children	Depth modifier
Source	0	1	$\stackrel{\text{def}}{=} 0$
Bifurcation	1	2	$+\,1$
Transient	1	1	$+\,0$
Terminal	1	0	$+\,1$

Since terminal pressures are uniform, we can use the consistency constraint in (3) to calculate the child radius ratio at a bifurcation,

$$\frac{r_1}{r_2} = \left(\frac{R_1^* Q_1}{R_2^* Q_2}\right)^{\frac{1}{4}}, \tag{10}$$

with which the radius fractions may be calculated using (8). The reduced resistance of the parent branch at a bifurcation is then calculated from its downstream components as

$$R_p^* = \frac{8\mu L_p}{\pi} + \left(\frac{f_1^4}{R_1^*} + \frac{f_2^4}{R_2^*}\right)^{-1}. \tag{11}$$

III. NEW METHOD: ACCELERATED CCO

The established CCO approach to producing multiple non-intersecting networks adds terminals into the network by considering multiple candidate topologies, optimising each for volume and then selecting the minimum volume network with no intersections [11], [15], [22]. The intersection test at each stage leads to poor performance: the most efficient algorithm reported in the literature [15] scales as $\mathcal{O}(N^2 \log N)$, where N is the number of terminal nodes, requiring days of computing time for complex cases.

We propose a novel method, breaking the process into 3 separate stages: Construction, Optimization, and Collision Resolution (see Fig. 1). With the reasoning that the diameter of any practical vasculature is small compared to the space it serves, networks are constructed in parallel with no knowledge of each other, and collisions between them are resolved by making small excursions around the contact points. This is achieved by introducing a new type of node, the *transient* (see Table III), which acts to create piecewise approximations to curved branches. Such nodes have previously been considered in the context of single arterial trees [10] to introduce tortuosity.

We will use the term *depth* of a node similarly to its standard usage with respect to tree data structures: the number of edges between a node and the root. The difference is that we do not regard the transient as increasing logical depth, as we can package segments connected by transients into single branches, which we view as the fundamental unit of the tree. The tissue volume around a single terminal will be referred to as a *unit cell*, and will be considered to be supplied by the network if there is a terminal vessel linking this node to the network (the terminal is *constructed*), or if a much larger vessel is occupying the unit cell. The volume supplied by the entire network is referred to as the *perfusion space*. When multiple networks are created to meet at the same terminal points (e.g. arterial/venous pairs), we refer to them as being *matched*. There is no limit to the multiplicity of matching.

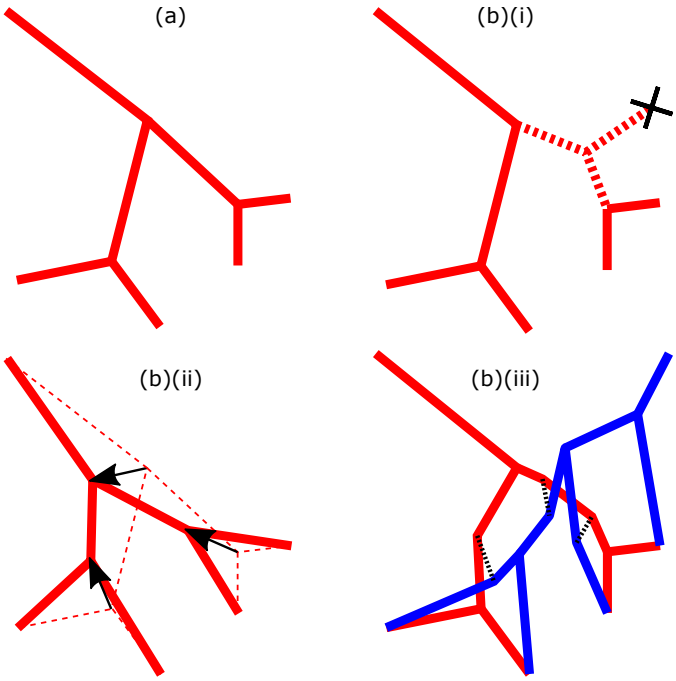

Fig. 1. The Accelerated CCO process: starting from (a), there are three actions available. (b)(i) Construction: adding new terminals into the network. The network topology is altered, and the only geometry which is altered is that of the branches connecting the new bifurcation (the bifurcation triad, $\mathcal{T}$, shown dashed). (b)(ii) Optimization: The network topology is fixed, and the geometry of the entire network is altered to minimize the total network volume. (b)(iii) Collision Resolution: All networks are considered, and an attempt is made to resolve any disallowed intersections geometrically. If this fails, the network topology is altered by removing terminals to make space for larger vessels, reducing the network density in regions of unresolvable intersections.

We demonstrate techniques for these stages which scale as $\mathcal{O}(N \log N)$ in the best case, no more than $\mathcal{O}(N^2)$ at worst, with an expected average best case performance for volumes of $\mathcal{O}(N^{4/3})$.

A. Creating a bifurcation

CCO iteratively adds terminal sites into the network, meaning that the overall complexity is N times the complexity within each iteration. At each iteration, a branch must be selected from the existing network from which to create a bifurcation, and the location of this bifurcation must be determined. The standard method is to evaluate all existing branches and pick the best, which must give at least $\mathcal{O}(N^2)$ scaling. A new *counted* selection process is proposed, which moves down the tree until it encounters a given number of consecutive branches which make no further progress to the target site than their parent (Algorithm 1).

1) Candidate evaluation: The selector is supplied with an evaluation function: this takes a branch and a terminal, and returns a structure containing a reference to the branch, a non-negative cost against which branches are compared, and a flag indicating suitability. A branch being unsuitable does not prevent its children from being tested, but does prevent it from being passed back upstream if a less optimal but suitable alternative is available. If the evaluator wishes to reject the terminal (e.g. it is completely contained within a branch) the cost is set to be negative and the suitability flag is set - this is then guaranteed to be returned from the search process. The terminal is rejected if no suitable candidates were found or the cost is negative. For the investigations in this paper, we consider only the Euclidean distance from the terminal node to the branch as the cost function: with prescribed flow through branches and pressure being fixed over the network, we expect that longer paths will require larger radii. The suitability test is whether the triangle formed by the set of nodes around the bifurcation (the *bifurcation triad*, $\mathcal{T}$) is likely to degenerate into a line. Extensions which we do not investigate here (but are briefly touched upon in supplementary material) include augmenting the cost with a flow rate term and providing a maximum permissible flow asymmetry.

Algorithm 1 Counted selection method

```
Require: terminal, skipLimit : int[0,∞], evalFunc
Function Select(current, count) :
  best ← current
  for all child c of current.Branch do
    candidate ← evalFunc(c, terminal)
    if candidate.Cost < current.Cost then
      downstream ← Select(candidate, 0)
      if downstream.Suitable
      and downstream.Cost < best.Cost then
        best ← downstream
      end if
    else if count < skipLimit then
      downstream ← Select(candidate, count + 1)
      if downstream.Suitable
      and downstream.Cost < best.Cost then
        best ← downstream
      end if
    end if
  end for
  return best
```

2) Performance: The counted selection method will never do worse than testing all existing branches, giving an upper bound on complexity of $\mathcal{O}(n)$, where n is the number of terminals currently constructed, since the number of branches is $\Theta(n)$. When a low number of skips, S, is permitted, the performance should scale as the mean depth of the branches currently in the network. Furthermore, we can guarantee that the depth of the bifurcation created will be less than or equal to the depth produced by searching all existing branches, which will help maintain the good balance required to achieve better query performance. The minimum scaling would be achieved if the tree had uniform depth balance, which would scale as $\mathcal{O}(\log n)$.

We expect that the depth distribution along these extremal paths will be approximately uniform, with a mean that scales with the characteristic number of terminals in any dimension, for the following reasons: (1) the average path to the most extreme terminals of a perfusion space will pass a number of internal terminals that scales with the characteristic number of terminals in any dimension; (2) minimum volume principles suggest that the path should bifurcate into these terminals as

soon as they are passed (observed in [16], [19], [23]–[25]). We therefore expect performance of volumes to scale no better than $\mathcal{O}(n^{1/3})$. For perfusion spaces where the terminals are arranged as a shell around the inlet, better performance is expected, whereas for long, thin volumes we expect the worst case performance to be achieved.

Perfect depth balance can be achieved by using a space-filling fractal [26], but the suitability of fractals for large-scale organs is debatable [27]–[29]. Notably, manufactured models which have been designed to achieve uniform depth balance have only been shown to supply a line or plane of points [30], [31].

B. Volume optimization

The network volume is approximated as being the sum of cylindrical volumes over the the set of existing branches, $\mathcal{B}$,

$$V(\mathcal{B}) = \pi \sum_{b\in\mathcal{B}} r_b^2 L_b. \tag{12}$$

Previous implementations have optimized the position of bifurcations as they are added into the network, which we will refer to as 'incremental' optimization.

1) Accelerated incremental optimization: Karch *et al.* [32] considered that when a bifurcation is moved, only the reduced resistances and radii fractions in the direct path upstream to the source must be re-evaluated. We introduce a new derived property, the *effective length*, L^*, which allows the approximation to the volume to be re-evaluated in a similar manner, requiring only calculation along the path to the root, avoiding the costly evaluation of the sum. By expanding the sum and replacing radii with the root radius multiplied by their chained fractions from (6) we get:

$$V(\mathcal{B}) = \pi \left(L_0 r_0^2 + L_1 f_1^2 r_0^2 + L_2 f_2^2 r_0^2 + L_{1,1} f_{1,1}^2 f_1^2 r_0^2 + ... \right), \tag{13}$$

from which common factors may be recursively extracted:

$$\pi r_0^2 \left(L_0 + f_1^2 \left(L_1 + f_{1,1}^2 \left(L_{1,1} + ... \right) + ... \right) + f_2^2 \left(L_2 + ... \right) \right). \tag{14}$$

We can now define the effective length relationship at terminals and bifurcations:

$$L^*_{\text{term}} = L_{\text{term}}, \tag{15}$$

$$L^*_p = L_p + f_1^2 L_1^* + f_2^2 L_2^*. \tag{16}$$

The total volume downstream of a branch is therefore

$$V(\mathcal{B}) = \pi r_0^2 L_0^*. \tag{17}$$

The update rule can be generalized to any cost function of the form considered by Schreiner *et al.* [23]:

$$C(\mathcal{B}, \lambda, \rho) = \sum_{b\in\mathcal{B}} r_b^\rho L_b^\lambda, \tag{18}$$

where λ, ρ are arbitrary constants. Using the following substitutions: $L_i \to L_i^\lambda$, $f_i^2 \to f_i^\rho$, the general form of the evaluation becomes:

$$C(\mathcal{B}, \lambda, \rho) = r_0^\rho L_0^*. \tag{19}$$

2) Batch optimization: The fundamental limitation of incremental optimization is that the effect of moving a single bifurcation on the entire network becomes negligible as the total number of nodes becomes large. Since we will deal with any intersections at a later point in the algorithm, we may interrupt the build at various stages to optimize every single bifurcation at once.

Karch *et al.* [32] demonstrated that the volume of the whole network when perturbing a single bifurcation has a single minimum somewhere within the bifurcation triad, permitting the use of the simplest method of minimization; gradient descent of the volume with respect to the position of each bifurcation:

$$x' = x - \eta \nabla V(x), \tag{20}$$

where η is some small parameter. Whilst we could use the accelerated volume calculation method to estimate the gradient at each node, we would pay a computational cost per iteration of N times the mean depth of the whole tree, giving an expected $\mathcal{O}(N^{4/3})$ scaling for perfusion spaces.

A constant time approximation to the gradient, giving an overall scaling per iteration of $\Theta(N)$, is justified here. Starting by differentiating (17) with respect to a bifurcation position,

$$\nabla V(x) = \pi r_0^2 \nabla L_0^*(x) + 2\pi L_0^* r_0 \nabla r_0(x), \tag{21}$$

we note from (9) that the root radius is a function of only root reduced resistance, R_0^*, given that the flow rates and total pressure drop are fixed by the logical configuration of the tree. Since changes in R^* propagate upwards from the bifurcation to the root, we see from (11) that there is a scalar gradient chain through the upstream nodes, $\mathcal{U}_m$: $m \in \{1 \dots M\}$, where 1 denotes the first child of the root and M the immediate upstream node of the bifurcation, and the vector terms are introduced at the bifurcation triad via their physical lengths (using $\mathcal{C}$ to denote the set of child branches):

$$\nabla R_0^*(x) = \frac{\mathrm{d}R_0^*}{\mathrm{d}R_{\mathcal{U}_1}^*} \left(\prod_{i=1}^{M-1} \frac{\mathrm{d}R_{\mathcal{U}_i}^*}{\mathrm{d}R_{\mathcal{U}_{i+1}}^*} \right) \frac{\mathrm{d}R_{\mathcal{U}_M}^*}{\mathrm{d}R_p^*} \nabla R_p^*(x), \tag{22}$$

$$\nabla R_p^*(x) = \frac{8\mu}{\pi} \left(\nabla L_p(x) + \sum_{i\in\mathcal{C}} \frac{\mathrm{d}R_p^*}{\mathrm{d}R_i^*} \nabla L_i(x) \right). \tag{23}$$

In order to avoid vanishing gradients due to deep chains, we wish to approximate the gradient direction and work out the step size later. Defining $\beta = r_1/r_2$, the bifurcation relationship yields:

$$\frac{\mathrm{d}R_p^*}{\mathrm{d}R_1^*} = -\left(\frac{f_1^4}{R_1^*} + \frac{f_2^4}{R_2^*} \right)^{-2} \left(\frac{(1+\beta^{-\gamma})^{-\frac{4}{\gamma}}}{R_1^{*2}} \left(\frac{\beta^{-\gamma}}{1+\beta^{-\gamma}} - 1 \right) - \frac{(1+\beta^{\gamma})^{\frac{-4-\gamma}{\gamma}} \beta^\gamma}{R_2^* R_1^*} \right). \tag{24}$$

The expression for the second child can be found by exchanging the indices. Considering that

$$\frac{z}{1+z} - 1 \le 0 \qquad \forall z \ge 0, \tag{25}$$

we know that the gradient chain elements are all positive. $\nabla R_0^*(x)$ therefore points in the same direction as $\nabla R_p^*(x)$.

Disregarding the scaling factor $(8\mu/\pi)$ and evaluating the length gradients in (23), we get $\tilde{g}(x)$, the approximation to the gradient direction:

$$\tilde{g}(x) = \frac{x - x_p}{L_p} + \sum_{i \in \mathcal{C}} \frac{\mathrm{d}R_p^*}{\mathrm{d}R_i^*} \frac{(x - x_i)}{L_i}. \tag{26}$$

For the approximation to be suitable requires the terms in (21) from the effective length derivative to be either pointing in a similar direction, or small. The gradient of the parent effective length at a bifurcation with respect to the position of a different bifurcation downstream is:

$$\nabla L_p^*(x) = \nabla L_p(x) + \sum_{i \in \mathcal{C}} \left(f_i^2 \nabla L_i^*(x) + 2 f_i L_i^* \nabla f_i(x) \right), \tag{27}$$

and we claim the following:

1) $\nabla L_p(x) = 0$ for all terms not in the bifurcation triad. Whilst not in the exact same direction as (26), they have similar components (physical length gradients multiplied by positive coefficients) and arrive at the root expression having been attenuated by every f_i^2 on the way, reducing their effect.
2) One of the $f_i^2 \nabla L_i^*(x)$ terms at each bifurcation is zero, as it has no downstream changes.
3) Each $f_i L_i^* \nabla f_i(x)$ term has a gradient chain downstream to an expression proportional to (26). A pair is introduced at each bifurcation, and are attenuated by fewer f_i^2 terms than the physical length terms at the root. It is possible for the proportionality to (26) of this term to be negative: a small number of bifurcations will suffer this at enough nodes that the approximation will be pointing in the reverse direction to the true gradient.

The approximation of the gradient direction in (26) seems acceptable when considering the bulk movement of nodes, increasingly so at larger depths. In the case of transient nodes, we can view the parent and child branches together as a single element, meaning that only the length term of the parent reduced resistance is involved:

$$\nabla R_p^*(x) \propto \left(\nabla L_{\text{total}}(x) = \frac{x - x_p}{L_p} + \frac{x - x_c}{L_c} \right), \tag{28}$$

which gives a direction that is the mean of the branch unit vectors towards the transient, and the expected result of equilibrium lying on the line between parent and child.

All that remains is to determine the perturbation distance, Δ, which is set to be a fraction, η, of the minimum length in the bifurcation triad, $\mathcal{T}$:

$$\Delta = \eta \cdot \min(L_p, L_1, L_2). \tag{29}$$

However, without a termination condition we may find that bifurcations are pulled into overlap with other nodes, increasing collisions. When the overlap is at a terminal, this can have disastrous consequences for the completeness of perfusion due to terminal culling (see §III-C.6). Movement is prevented when the shortest triad length is less than a termination fraction, τ, of the minimum distance between surrounding nodes:

$$\min(L_p, L_1, L_2) \leq \tau \cdot \min(|x_1 - x_p|, |x_2 - x_p|, |x_2 - x_1|). \tag{30}$$

Should equilibrium lie outside of the termination regions, low amplitude oscillations or limit cycles may be established due to the increased step size in this region. Convergence can be encouraged in practice by ensuring that η starts small (limiting the potential overstep) and by reducing η over time. A simple implementation is to specify a reduction fraction and block length, with care taken to ensure that the block length is appropriate for the initial η - small strides need more iterations before reduction.

3) Initial bifurcation position: If we do not optimize the network incrementally, we need a method of determining where to place a bifurcation upon creation. Rather than place it at the arithmetic mean position of the nodes in $\mathcal{T}$, we propose that a more suitable (yet still trivial to calculate) approximation to the optimal position, $\tilde{x}$, is a weighted arithmetic mean, with the weighting being the flow:

$$\tilde{x} = \frac{\sum_{n \in \mathcal{T}} Q_n x_n}{\sum_{n \in \mathcal{T}} Q_n}. \tag{31}$$

Considering the extreme case, we see that this reduces the deviation taken by high flow branches:

$$Q_1 \gg Q_2 \implies \tilde{x} \approx \frac{1}{2}(x_p + x_1), \tag{32}$$

which helps minimize the length of the main branches.

C. Collision resolution

The collision resolution scheme is an iterative approach with three actions available:

1) Insert a transient in a branch to create an excursion.
2) Move a bifurcation/transient.
3) Remove a terminal.

We aim to minimise the removal of terminals, and care is taken to ensure that this only occurs when a much larger branch has no possible place to be moved - in this case the unit cell is essentially marked as having no useful tissue volume to support, and the loss to total flow does not harm the completeness of perfusion.

1) Detection: A naive approach to collision detection would be to test every branch against each other, which gives a complexity of $\Theta(N^2)$. By the use of an appropriate hierarchical bounding tree we can achieve $\mathcal{O}(N \log N + k^{1+\varepsilon})$ query, where k is the number of intersections, by rejecting entire groups using simple tests. In this case, we use *Axially Aligned Bounding Boxes* (AABBs), which can find a rejection in at most 6 comparisons [33].

Unfortunately, structures optimized for query have a high construction cost [34] – we use the assumption that we have produced a roughly balanced tree, combining sibling bounds at bifurcations into *downstream bounds*, which are then combined with their parent branch *local bounds* to produce the *global bounds* for that branch. This is by no means optimal in query, as at each layer of the tree we have both a leaf element and two child branches. We again expect query performance to scale as N times the mean depth – we sacrifice optimal query for linear cost of construction, as we must recalculate the bounds for each iteration of resolution. The query routines are demonstrated in Algorithms 2 and 3.

TABLE IV
INTERSECTION DATA AND THEIR PURPOSE.

Name	Type	Description
Indeterminate	Boolean	Whether a single mutual normal direction could not be established.
NormalAB	Vector3 (Normalised)	If determinate, the normal from branch A to B. If indeterminate and collinear, a randomly generated mutual normal. Otherwise, the in-plane direction from A to B.
Fraction, Closest (A,B)	Real[0,1], Vector3	The fraction along each branch of the closest point, and the evaluated point itself. Only valid if determinate.
Start, End (A,B)	Real[0,1], Real[0,1]	If indeterminate, the start and end fractions of intersection.
Distance2, Overlap	Real, Real	The square separation distance, used for determining intersection, and the overlap if intersecting.

Algorithm 2 Collision detection

```
Subroutine TestTree(a, b) :
  immune ← GenerateImmuneSet(a)
  TestDownstream(b, a)
  for all child ca of a do
    if TestBounds(ca.Global, b.Local) then
      TestTree(ca, b)
    else
      for all child cb of b do
        if TestBounds(ca.Global, cb.Global) then
          TestTree(ca, cb)
        end if
      end for
    end if
  end for
```

Algorithm 3 Collision detection downstream search

```
Subroutine TestDownstream(root, check) :
  if not immune.Contains(check)
  and TestBounds(root.Local, check.Local) then
    data ← TestBranches(root, check)
    if data.Intersecting then
      intersections.Add(data)
    end if
  end if
  for all child c of root do
    if TestBounds(check.Local, c.Global) then
      TestDownstream(c, check)
    end if
  end for
```

We model branches as capsules, which have simple intersection tests. The intersection data produced are shown in Table IV. Intersections where a single mutual normal direction between the branch directions cannot be established (the branches are either co-directional or collinear) are referred to as *indeterminate*, and are handled separately, as there may not be a single point of closest approach between the centrelines of the branches.

2) Immune sets: In the case of matched networks, the parent branch of a terminal node must intersect with each of the parents of the other matched terminals. An *immune set* is generated whenever the end node of a branch indicates it has matched partners, and intersection tests with branches in the set are skipped. For intersections within a network, each branch must generate an immune set of at least its sibling, children and parent. However, in cases where the branches have low slenderness we allow the set to be expanded: branches are recursively added both upstream and downstream until the logically closest node pair between the test branch and set-generating branch no longer touch.

3) The determinate case: Firstly, a decision is made as to whether the start and end nodes of each branch are involved in the intersection, by comparing the distance between the nodes and the point of closest approach, d, to the branch radius. Using the radial slenderness of the branch, s_r, the fraction along the branch of closest approach, $\chi = d/L$, and a *radial capture* constant, k_r, the start and end conditions are respectively

$$\left(\frac{d}{r} = \frac{d}{L}\frac{L}{r} = \chi s_r\right) < k_r, \tag{33}$$

$$(1-\chi)s_r < k_r. \tag{34}$$

If either node passes its test, a request is filed to move the intersecting nodes by a perturbation vector, v_p. If both nodes fail, a transient is requested at the closest approach point of the centreline plus v_p. The perturbation distance is calculated by multiplying the overlap, δ, defined as the sum of the branch radii minus the separation between the branch centrelines, by an aggression factor k_a, which tries to prevent the potential increase in branch radii after movement from causing the same collision again; this distance is then divided between the branches by their weighting fractions, which must satisfy

$$w_A + w_B = 1. \tag{35}$$

The perturbation vector, v_p, is therefore given by

$$v_p = \delta k_a w\hat{n} \tag{36}$$

where $w\hat{n}$ is the weighted normal in the relevant direction for each branch.

After all intersections have been processed, all node perturbation requests are averaged into a single perturbation for this iteration, which is clamped to ensure a minimum step size, $|v|_{min}$. This reduces the number of iterations required, but can introduce needless tortuosity if $|v|_{min}$ is too large. Transient requests are also averaged to insert a single transient per branch in each iteration. There is no minimum distance for these, as any unresolved intersections will be captured by the newly inserted transient in the next iteration, which will then follow the rules for node perturbations to ensure a solution is found.

4) The indeterminate case: In this case the node tests are of the same form as the determinate case, but with the length fraction replaced by the intersection range start and end fractions, respectively. There is now the possibility to request both node perturbations and transient insertion: all node requests are made the same as before; then, should any node in the intersecting set of A be stationary, a transient is requested at the midpoint of B's intersection range plus the perturbation vector, and vice versa.

5) Perturbation weighting: It often makes little sense to perturb a high-flow and low-flow path equally, as volume minimization suggests that the path with lower flow should yield. The simplest way to weight the perturbations to achieve this is using the total flow fraction of the opposing branch. However, between networks where the overall pressure differences are substantially different, the effect of perturbations on the root radius (and subsequently all radii) is amplified in the low-pressure case, so the user can specify a *relative compliance*, C, for each network to compensate for this. The overall perturbation weights are therefore:

$$\begin{aligned} w_A &= \frac{Q_B C_A}{Q_A C_B + Q_B C_A}, \\ w_B &= \frac{Q_A C_B}{Q_A C_B + Q_B C_A}, \end{aligned} \tag{37}$$

which satisfy the constraint in (35).

6) Terminal culling: As generated volumes become larger, the probability of situations where root branches grow to sizes similar to the terminal spacing increases, creating potentially unresolvable collisions with the stationary terminal nodes. Even if a solution were to be found, the tortuous structures created by diverting the flow around the terminal would increase volume, and optimization steps would most likely pull the branches back into the terminal, perpetuating the issue. To resolve this, a *culling* stage was added.

Whenever a request to move a terminal is made, the resolver increases its *cull count* by a user-specified penalty, z_p. After each iteration of resolution, the resolver checks the count of each node against a threshold value, z_t, and any nodes with counts greater than this are culled from their networks, alongside their matched partners. The resolver then decrements the count of all nodes, clamping values at zero.

Aggressive culling prevents the distortion of branches close to the source at the expense of those downstream, where solutions may be available if more attempts were allowed. To permit less aggressive culling whilst still protecting the root branches a critical ratio, β_c, is specified and if the condition

$$Q_{\text{term}} \beta_c < Q_{\text{branch}} \tag{38}$$

is met, the terminal is immediately removed from the network.

7) Ensuring 3D printability: The radii of vessels are kept above the printer minimum feature size, $r_{\min}$, by adjusting the overall pressure drop if necessary. To ensure vessel spacing is above the minimum feature size of the printer, their radii are padded for collision resolution purposes by another input parameter, r^+.

In terminal segments spaces between the narrow vessels are at risk of infill in printing, should the angle at which they meet be too large. We therefore provide the user with the option to smooth terminal segments as a final action before exporting the model to a CAD format. We export to Autodesk® Inventor® 2017 onwards, as well as directly exporting to .STL files.

IV. Results

A. Collision resolution

A demonstration of collision resolution and terminal smoothing working for a simple case is shown in Fig. 2. Whilst not perfect, angles have been mostly reduced to below 90°. An instance of terminal culling can also be seen, as there is a terminal in Fig.1a(ii) (right inset, circled) that is not present in Fig.1b(ii). Regions such as this, where we have one or more high-flow branches, are exactly where we expected to need to cull terminals.

B. Computational scaling

Performance of construction was tested by generating networks for an *Edge-fed Cubic Volume* (ECV), where the source node was placed above the centre of one face of a cubic terminal layout; a *Centre-fed Spherical Shell* (CSS), where the terminals were placed uniformly over the surface of a sphere, and the source node placed at the centre; and an *Above-fed Planar Surface* (APS), where a square surface of terminals is supplied by a network descending from a central out-of-plane source node. An example of each network is shown in Fig. 3. An important note is that for the CSS case the skip limit, S, must be increased. If $S = 0$, the order of construction becomes important: if the children of the first bifurcation point away from the target site, the bifurcation will be created at the root. Should the next terminal have a similar relationship to the new initial bifurcation, the process will repeat and the output will be a spiky sphere. For each setting 100 tests were performed with a fresh random number generator, seeded by the system tick count.

Fig. 4 shows a polynomial scaling estimate for each case. It can be seen that the mean performance on the ECV scales faster than previous algorithms by roughly a factor of $N^{1/2}$, with a number of outliers seeing far better performance. The CSS case scales best, despite the fact that S had to be increased for this geometry. Whilst we might have expected the APS to have seen better performance again, due to the ability to use lower values of S, the minimum volume structure does not bifurcate out-of-plane very often: the optimal volume solution is to move rapidly to the plane and then branch out in-plane (see Fig. 3), which we would expect to scale as $\mathcal{O}(N^{3/2})$, using the argument that the extremal paths pass a number of terminals scaling as $\mathcal{O}(N^{1/2})$.

1) Terminal depth distributions: Comparing terminal depth distributions (which we will denote as $d(i)$ to represent the fraction of terminals at depth i) across sizes is meaningless without a normalization: in this case, we know that the perfectly balanced tree will have all of its terminals at a depth of $\log N$, and we therefore compare distributions normalized

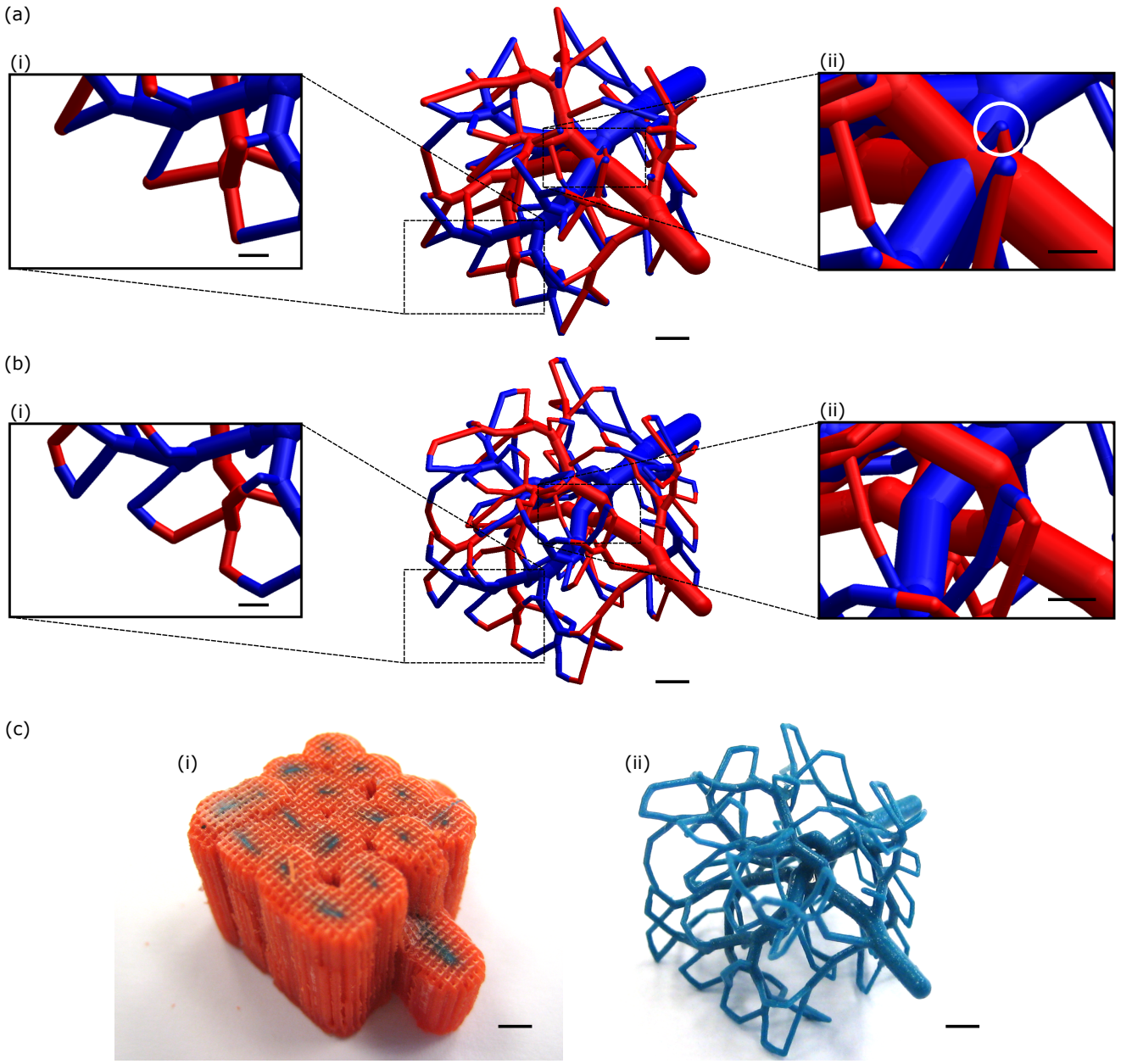

Fig. 2. A matched arterial/venous network pair supplying a cube of 64 terminals spaced at 4 mm, with a minimum radius of 125 µm. In (a), a network is shown after construction and volume optimization, showing (i) sharp angles prone to infill, and (ii) a large intersection. Shown in (b) is the same network after collision resolution and terminal smoothing, with (i) greatly reduced terminal angles, and (ii) clearance between the major vessels. In a(ii), the circled region shows a terminal which is not present in b(ii) due to terminal culling in collision resolution. Scale bars: main image: 2.5 mm, insets: 1 mm. (c) A 3D printed model template of the matched arterial/venous network. (i) The model was printed on a Solidscape S350 using Midas build material (blue) and melt dissolvable support (orange). (ii) Support material was removed using a selective solvent bath and model dried in air. Model displays precise reproduction of the algorithmically-generated network. Scale bar: 2.5 mm.

against this. These comparable distributions are referred to as $d(i')$, where $i' = i/\log N$. For notational convenience, we do not explicitly mention the dependence of d on N and geometric factors.

Fig. 5 shows that, as expected, the depth distribution mean and standard deviation for an ECV is seen to grow almost exactly as $N^{1/3}$, suggesting that we could do no better than $\mathcal{O}(N^{4/3})$ performance overall, if we were able to find a cost function to use in the construction phase that prevents needless downstream searches more effectively than the Euclidean distance. The distribution for the CSS case shows a very slight decay (Fig. 6), suggesting that the performance floor would be closer to $\mathcal{O}(N \log N)$ if we were to supply a shell-like geometry for which we may use $S = 0$.

2) Collision resolution: The number of iterations required for total collision resolution (Fig. 7) is seen to have a weak increase with the number of terminal nodes, but remains fairly constant over the range of problem sizes that we are currently interested in. The position of source nodes determines the shape of the distribution: when inlet and outlet are on opposite sides of the perfusion volume, low-flow branches of each network will be intersecting high-flow branches of the other,

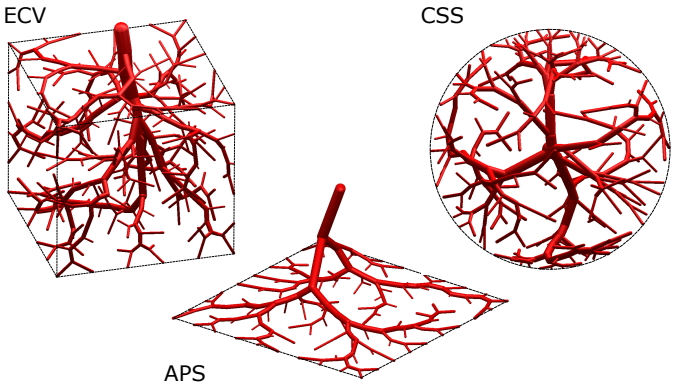

Fig. 3. Single sample networks for the 3 configurations used for testing performance: edge-fed cubic volume (ECV), centre-fed spherical shell (CSS) and above-fed planar surface (APS), with their bounding geometries shown.

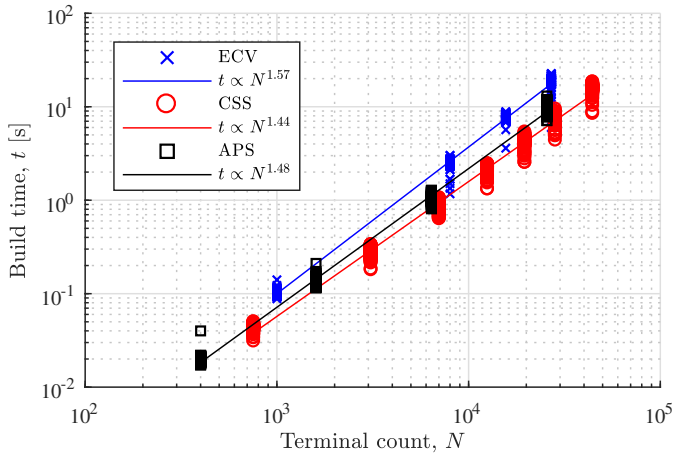

Fig. 4. Estimate of polynomial scaling of build time, $\boldsymbol{t}$, for an edge-fed cubic volume (ECV), centre-fed spherical shell (CSS) and above-fed planar surface (APS). Note that the CSS case required $\boldsymbol{S = 1}$ due to the geometry of the problem, whereas the others were able to use $\boldsymbol{S = 0}$. 100 tests performed at all settings.

which is more likely to require many iterations to fully resolve.

An estimate of the computational scaling of a single iteration of collision resolution is shown in Fig. 8. Whilst there is a lot of variability in the time taken, it is seen to scale better than construction. However, for sizes that we are currently generating, the time taken is similar to, if not greater than, the time taken in construction.

C. Volume of generated networks

The mean volumes of ECV networks with 1000 terminals generated using various selection methods, $\bar{V}$, are shown in

TABLE V
VOLUME MEAN AND STANDARD DEVIATION FOR A 1000 TERMINAL ECV BY SKIP SELECTION PARAMETER RELATIVE TO THE FULL SELECTION POLICY (EQUIVALENT TO $\boldsymbol{S = \infty}$). 100 TESTS PERFORMED AT ALL SETTINGS.

S	0	1	2	∞
$\bar{V}/\bar{V}_\infty$	0.8869	0.9074	0.9193	1.0000
$s_V/\bar{V}_\infty$	0.0243	0.0245	0.0245	0.0383

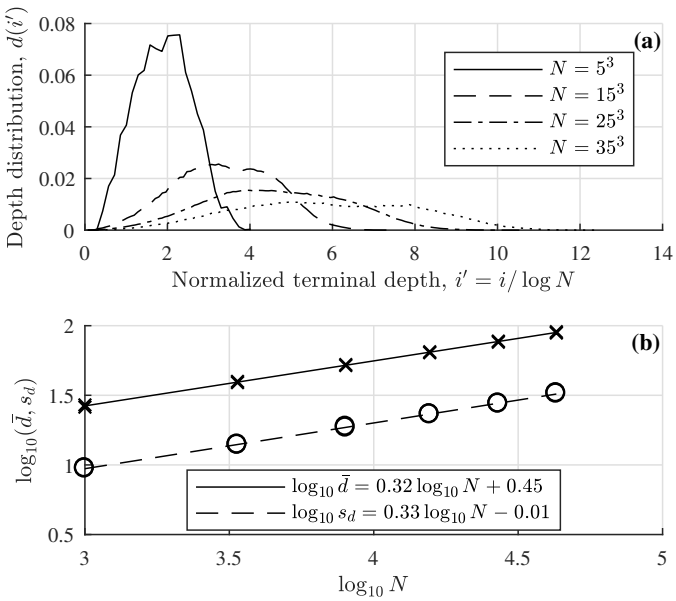

Fig. 5. Terminal depth distributions for the cubic volume (ECV). (a) The observed comparable distributions $\boldsymbol{d(i')}$ for a range of terminal counts $\boldsymbol{N}$, which broaden with increasing $\boldsymbol{N}$. (b) The growth in mean, $\bar{\boldsymbol{d}}$, and standard deviation, $\boldsymbol{s_d}$, of terminal depth distributions follow the expected power law in $\boldsymbol{N}$. 100 tests performed at all settings.

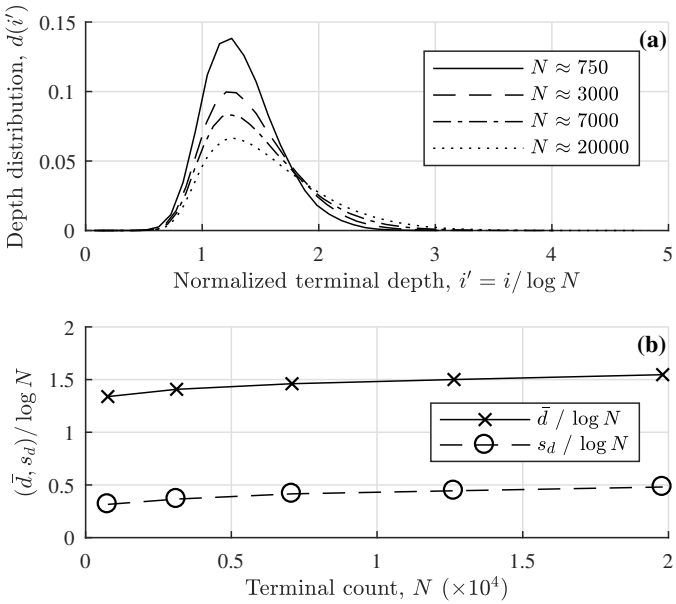

Fig. 6. Terminal depth distributions for the spherical shell (CSS). (a) The observed comparable distributions $\boldsymbol{d(i')}$ for a range of terminal counts $\boldsymbol{N}$, which remain similar despite large changes in $\boldsymbol{N}$. (b) The normalized mean, $\bar{\boldsymbol{d}}$, and standard deviation, $\boldsymbol{s_d}$, of terminal depth distributions are seen to vary much less with $\boldsymbol{N}$ than the ECV case. 100 tests performed at all settings.

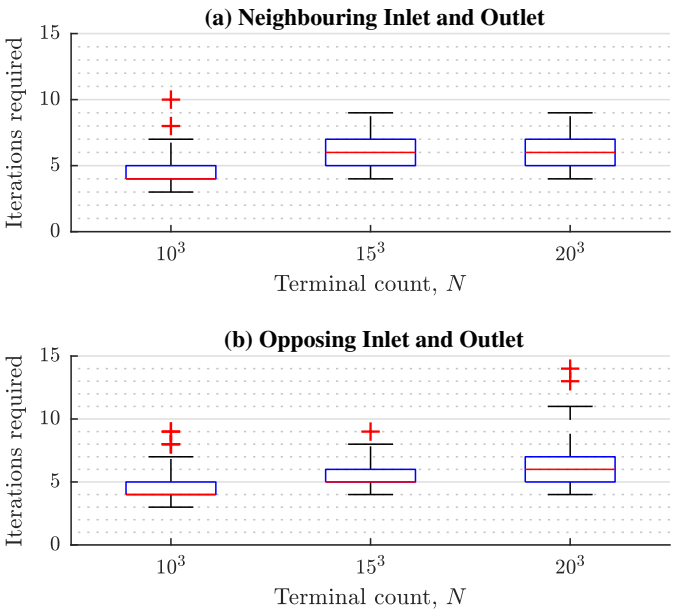

Fig. 7. The number of iterations required for collision resolution for an ECV network with (a) neighbouring and (b) opposing inlet and outlet. Box plot shows minimum, median, maximum, and interquartile range. Outliers (determined as being more than 1.5 interquartile ranges beyond the upper and lower quartiles) are plotted as crosses. 100 tests performed at all settings.

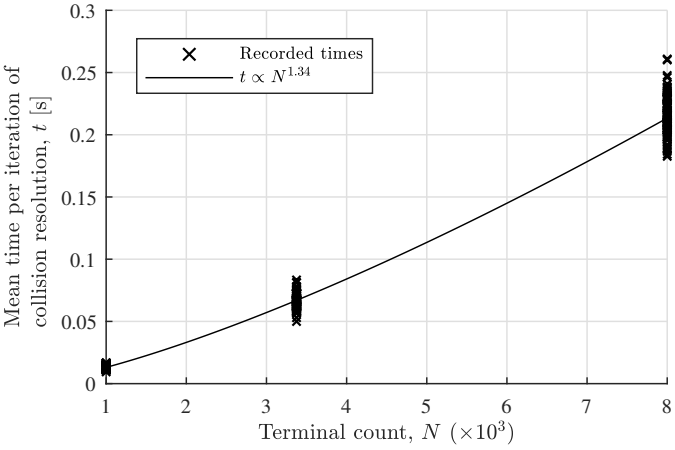

Fig. 8. An estimate of a polynomial scaling law for the time required per iteration of collision resolution for an ECV network. 100 tests performed at all settings.

Table V, normalized against the reference volume of the traditional 'full' search, $\bar{V}_\infty$. The standard deviations, s_V, seen in this are also shown, again normalized by $\bar{V}_\infty$, to give an idea of the susceptibility of each method to the randomness in the build order. The optimization scheme used for these was incremental and flow weighted. It can be seen that the minimum volume is achieved using the most aggressive scheme (skip limit, $S = 0$) and that the standard deviation of volumes is lower when using counted selection compared to the full search case, $S = \infty$. Due to the fact that it is not possible to use $S = 0$ with all geometries, we recommend using $S = 1$ as a default.

1) Verifying the gradient approximation: Fig. 9 shows the distribution of alignment between the fast gradient approximation, $\tilde{g}(x)$, and a finite-difference estimation of the gradient, $\nabla_\epsilon V(x)$, with stride size $\epsilon = 0.01$ mm, measured as the dot product between their directions:

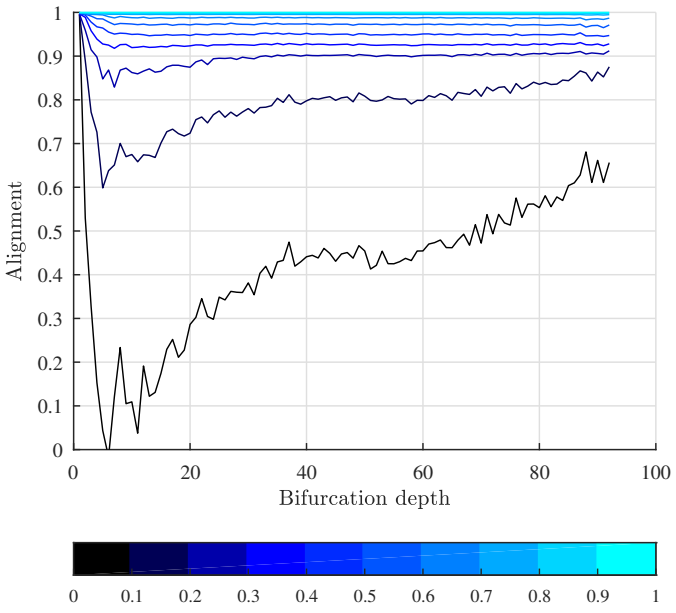

Fig. 9. Alignment of finite difference estimated gradient, $\boldsymbol{\nabla_\epsilon V(x)}$, with $\tilde{\boldsymbol{g}}(\boldsymbol{x})$, the fast approximation to the gradient, for 100 instances of an 8000 terminal ECV network. The contours represent the cumulative distribution of alignments at each bifurcation depth, plotted in increments of 0.1, with the color scale varying from 0 to 1.

$$\text{Alignment} = \frac{\tilde{g}(x)^T \nabla_\epsilon V(x)}{|\tilde{g}(x)||\nabla_\epsilon V(x)|}. \tag{39}$$

This was measured for all bifurcations in 100 instances of an 8000 terminal ECV as a function of bifurcation depth. Whilst we see somewhat poor alignment at lower bifurcation depths (up to 10% of nodes being moved in the wrong direction), at depths at which the majority of the bifurcations reside (25 to 75) we find that 80% of nodes show alignment above 0.8, showing that in almost all cases, the $\mathcal{O}(N^{1/3})$ saving in complexity does not excessively sacrifice optimality.

2) Effect of optimization schemes: The effect of various optimization schemes is shown in Fig. 10. At each size, the mean volumes and confidence intervals were normalised against the flow-weighted, incrementally optimized scheme for that size, denoted $V_{Q,I}(N)$. Our implementation of an incremental optimizer terminated when it failed to reduce the volume by at least a certain fraction (in this case 10^{-4}) of the current volume. Once the network is large enough, it is not possible for the newly created bifurcations to make such a significant impact on the volume, and the optimizer terminates almost immediately. This is most likely the reason for flow weighting having such an effect when using the incremental optimizer; when many bifurcations are deemed too insignificant to spend time optimizing individually, the effects of better initial position for each have a large impact on total volume.

Batch optimization is a clear improvement over incremental, giving a mean volume reduction greater than 10% as well as being more consistent. In this example, we interrupted

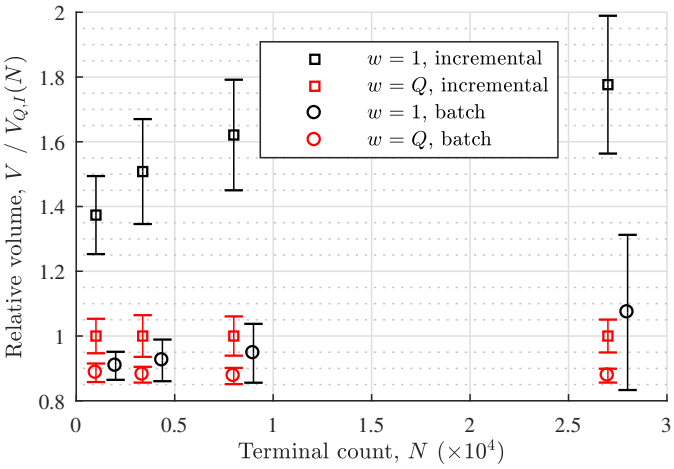

Fig. 10. Relative volumes at various sizes with $\boldsymbol{S} = \mathbf{2}$ under various initial bifurcation weightings (unit, $\boldsymbol{w} = \mathbf{1}$, and flow, $\boldsymbol{w} = \boldsymbol{Q}$) and optimization schemes, with 95% confidence bounds. Note that the points for $\boldsymbol{w} = \mathbf{1}$ with batch optimization have been offset for clarity. 100 tests performed at all settings with ECV configuration.

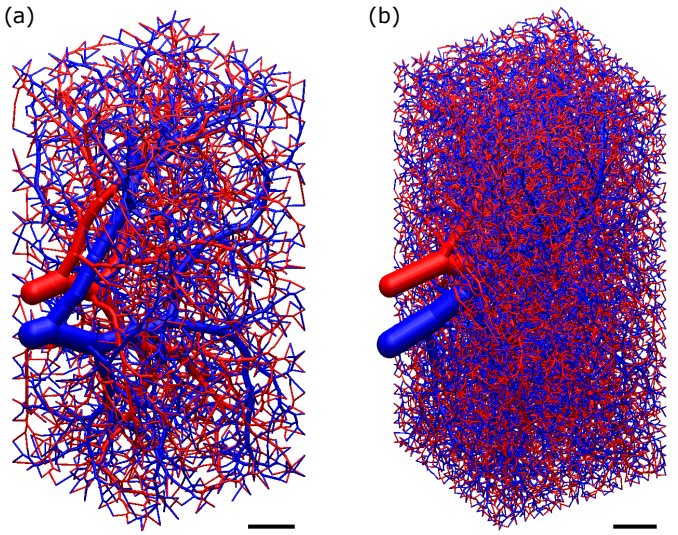

Fig. 11. A matched arterial/venous pair of kidney-like dimensions. In (a), terminals are spaced at 5 mm giving 2,000 total, with all vessels greater than 400 µm in diameter. This model took 1.95 s computational time to produce. In (b), terminals are spaced at 2.5 mm giving 16,000 total, but vessels are approaching the printable limit of 250 µm diameter. This model took 39.13 s computational time to produce. Scale bars: 10 mm.

the build at fractions of (0.1, 0.2, 0.5, 1) to perform 10 iterations of optimization, which gave similar total time to the incrementally optimized case at the intermediate value of $N = 8000$. Even with batch optimization, flow-weighting is still relevant - unit-weighting gives worse performance and higher uncertainty at all terminal counts, and worsens as terminal count increases. This is due to the fact that geometric and topological optimization are not actually separable: flow-weighting provides a better guess of the optimal geometric configuration, which leads to better topology (we select candidates based on distances, a geometric property), which in turn is more responsive to geometric optimization.

Using finite-difference gradient estimation at depths below 10 produced a small (<1%) improvement to mean volume in the case of $N = 8000$, suggesting that the power of batch optimization arises from the bulk movement of nodes that are individually insignificant, where the approximation to the gradient is better.

D. Pseudo-biomimetic 3D printable vascular systems

One of the key aims in manufacturing networks is that the inlet vessel is of a realistic diameter, to allow future anastomoses to native vessels. To achieve this, we adjusted pressures after construction to give correct dimensions at the inlet. Since the focus is on the higher flow branches, Murray's exponent was set to $\gamma = 2$ to give more realistic radius decay in these vessels [3]. We have aimed to produce the most volume-optimal models, which can result in highly asymmetric bifurcations near the root. The most extreme bifurcation ratio, $\beta = r_1/r_2$ where $r_2 > r_1$, we have recorded is $\beta = 0.082$, which is within the reported range in real tissue, $\beta > 0.05$ [35], but real tissue does not show as many large asymmetries as the volume-optimal designs. A short discussion on the distribution of vessel lengths and radii, as well as a method to improve biomimimeticism and its impact on performance, is provided as supplementary material. For these models, $r_{\min} = r^+ = 125$ µm, the current 3D printable limit in removable material.

Whilst we have not yet implemented the accurate geometry of organs, we have used simple approximations with realistic dimensions to demonstrate 3D printable models at large scales. For a method to adapt the collision resolution technique to approximate real organ geometry, the reader is referred to supplementary material. Fig. 11 shows an approximation to a human-sized kidney model as a cuboid with a matched arterial/venous network pair. Provided that there is sufficient space to resolve the intersections without excessively culling terminals (dependent upon the ratio between unit cell spacing and minimum vessel diameter, as well as the number of networks), the Accelerated CCO approach can be successfully deployed over a wide range of unit cell densities. The volume of the kidney was taken to be 5 cm × 5 cm × 10 cm, with arterial diameter 5 mm and venous diameter 6 mm [36]–[38].

A human-sized liver (Fig. 12) was approximated as a triangular prism supplied and drained by a quadruply-matched network. The volume was taken to be 20 cm × 15 cm × 5 cm, with inlet diameters of 12 mm (hepatic vein), 10 mm (portal vein), 6 mm (hepatic artery), and 4.5 mm (bile duct) [39]–[42].

The model displays several biomimetic features; these include the four networks present in the liver, alongside comparable inlet sizes and overall dimensions. However, there are currently some limitations to the level of biomimicry using this algorithm. In the liver, the hepatic artery, portal vein and biliary tree follow the same path through to the lobe, segmental and sub-segmental regions. In our model, the three networks follow independent paths through the structure, and the subunit regions are not defined at any length-scale. However, these characteristic elements of the physiological liver could be incorporated into our models through subdivision of the perfusion space, and by initially creating a single network which is split into three. The inability to 3D print capillary-

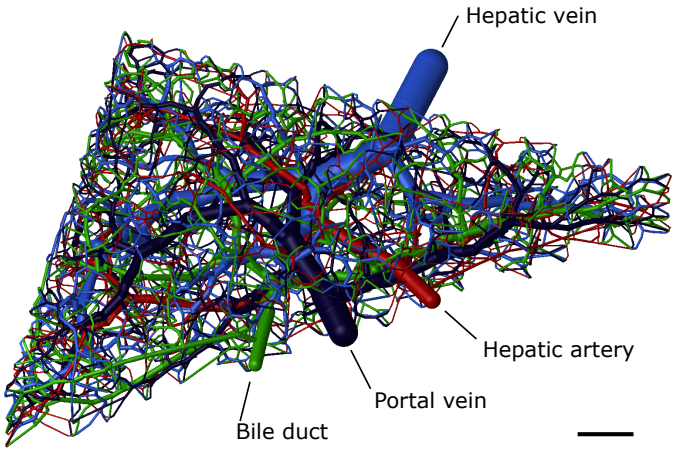

Fig. 12. A quadruply-matched network with liver-like dimensions. Terminals are spaced at 10 mm, giving 750 total. All vessels are above 300 µm diameter. This model took 1.22 s computational time to produce. Scale bar: 20 mm.

sized features limits the inclusion of the lobule micro-structure present in the native liver. Finally, one notable departure from the physiological liver is that the biliary tree connects to the blood vessel trees through each of the terminals. This, while not biomimetic, is necessary to clear the 3D printed template material from the channel network (when the bulk matrix is micro-porous). A discussion and close-up renderings of possible unit cell structures is included as supplementary material.

It is of note that each unit cell in the CCO model is supported with its own terminal, therefore producing a uniform pressure distribution in the perfusion space, which leads to biomimetic vascular networks. This is in contrast to networks fabricated via 3D printing methods where the perfusion space is typically supplied through a series of one-dimensional channels [30], [43], [44] leading to variation in pressure across the perfusion space. Further, the space occupied by connections between inlet and terminal one-dimensional channels will produce unsupported regions within the tissue construct.

V. Conclusion

The success of tissue engineered constructs largely depends on the incorporation of perfusable vascular networks which can support the biological functions of the embedded cells. While 3D printing techniques have evolved to create complex structures with geometric precision, currently there is no method for automatic generation of 3D printable vascular networks for arbitrary scales and spaces, and for interpenetrating networks. Here, we report a method for generating 3D printable vasculature templates, based on Constrained Constructive Optimization, by dividing the process into distinct stages of construction, optimization and collision resolution. The method accounts for physical constraints (flow conservation, pressure consistency), physiological constraints (minimum network volume, Murray's law, no short-circuit intersections), and manufacturing constraints (minimum feature sizes, infill prevention). Construction times are significantly faster than those previously reported on vascular trees. Computational complexity is observed to scale as $\mathcal{O}(N^{1.57})$, a factor of $N^{0.43} \log N$ better than the previous best implementation (N: number of terminal nodes). The conversion of these networks into perfusable channel systems, embedded in biomaterials, requires the use of sacrificial 3D printing techniques; however, discussion of this is beyond the scope of this computational work.

References

[1] J. Rouwkema and A. Khademhosseini, "Vascularization and Angiogenesis in Tissue Engineering: Beyond Creating Static Networks," *Trends in Biotechnology*, vol. 34, pp. 733–745, Sept. 2016.

[2] C. D. Murray, "The physiological principle of minimum work applied to the angle of branching of arteries," *The Journal of General Physiology*, vol. 9, pp. 835–841, July 1926.

[3] T. F. Sherman, "On connecting large vessels to small. The meaning of Murray's law.," *The Journal of General Physiology*, vol. 78, pp. 431–453, Oct. 1981.

[4] I. S. Kinstlinger and J. S. Miller, "3d-printed fluidic networks as vasculature for engineered tissue," *Lab on a Chip*, vol. 16, no. 11, pp. 2025–2043, 2016.

[5] K. H. Benam *et al.*, "Engineered In Vitro Disease Models," *Annual Review of Pathology: Mechanisms of Disease*, vol. 10, no. 1, pp. 195–262, 2015.

[6] E. Perica and Z. Sun, "Patient-specific three-dimensional printing for pre-surgical planning in hepatocellular carcinoma treatment," *Quantitative Imaging in Medicine and Surgery*, vol. 7, pp. 668–677, Dec. 2017.

[7] D. Selle *et al.*, "Analysis of vasculature for liver surgical planning," *IEEE Transactions on Medical Imaging*, vol. 21, pp. 1344–1357, Nov. 2002.

[8] G. Hamarneh and P. Jassi, "VascuSynth: Simulating vascular trees for generating volumetric image data with ground-truth segmentation and tree analysis," *Computerized Medical Imaging and Graphics*, vol. 34, pp. 605–616, Dec. 2010.

[9] M. A. Galarreta-Valverde *et al.*, "Three-dimensional synthetic blood vessel generation using stochastic L-systems," vol. 8669, p. 86691I, International Society for Optics and Photonics, Mar. 2013.

[10] M. Schneider *et al.*, "Physiologically Based Construction of Optimized 3-D Arterial Tree Models," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011*, Lecture Notes in Computer Science, pp. 404–411, Springer, Berlin, Heidelberg, Sept. 2011.

[11] M. Krętowski and J. Bézy-Wendling, "Computer modelling of vascular systems," *TASK Quarterly : scientific bulletin of Academic Computer Centre in Gdansk*, vol. Vol. 8, No 2, pp. 223–229, 2004.

[12] R. Karch *et al.*, "Staged Growth of Optimized Arterial Model Trees," *Annals of Biomedical Engineering*, vol. 28, pp. 495–511, May 2000.

[13] W. Schreiner and P. F. Buxbaum, "Computer-optimization of vascular trees," *IEEE transactions on bio-medical engineering*, vol. 40, pp. 482–491, May 1993.

[14] L. O. Schwen and T. Preusser, "Analysis and Algorithmic Generation of Hepatic Vascular Systems," 2012.

[15] L. O. Schwen *et al.*, "Algorithmically generated rodent hepatic vascular trees in arbitrary detail," *Journal of Theoretical Biology*, vol. 365, pp. 289–300, Jan. 2015.

[16] A. A. Linninger *et al.*, "Cerebral Microcirculation and Oxygen Tension in the Human Secondary Cortex," *Annals of Biomedical Engineering*, vol. 41, pp. 2264–2284, Nov. 2013.

[17] M. Kociński *et al.*, "3d image texture analysis of simulated and real-world vascular trees," *Computer Methods and Programs in Biomedicine*, vol. 107, pp. 140–154, Aug. 2012.

[18] X. Liu *et al.*, "Simulation of Blood Vessels for Surgery Simulators," in *2010 International Conference on Machine Vision and Human-machine Interface*, pp. 377–380, Apr. 2010.

[19] M. Georg *et al.*, "Global Constructive Optimization of Vascular Systems," Jan. 2004.

[20] D. Szczerba and G. Székely, "Macroscopic Modeling of Vascular Systems," in *Medical Image Computing and Computer-Assisted Intervention — MICCAI 2002*, Lecture Notes in Computer Science, pp. 284–292, Springer, Berlin, Heidelberg, Sept. 2002.

[21] J. Li *et al.*, "An approach to integrating shape and biomedical attributes in vascular models," *Computer-Aided Design*, vol. 39, pp. 598–609, July 2007.

[22] M. Krętowski *et al.*, "Physiologically based modeling of 3-D vascular networks and CT scan angiography," *IEEE Transactions on Medical Imaging*, vol. 22, pp. 248–257, Feb. 2003.
[23] W. Schreiner, "The influence of optimization target selection on the structure of arterial tree models generated by constrained constructive optimization," *The Journal of General Physiology*, vol. 106, pp. 583–599, Oct. 1995.
[24] A. Klarbring *et al.*, "Topology optimization of flow networks," *Computer Methods in Applied Mechanics and Engineering*, vol. 192, pp. 3909–3932, Aug. 2003.
[25] A. Runions *et al.*, "Modeling Trees with a Space Colonization Algorithm," Sept. 2007.
[26] G. Vozzi *et al.*, "Microfabricated fractal branching networks," *Journal of Biomedical Materials Research Part A*, vol. 71A, pp. 326–333, Nov. 2004.
[27] M. Zamir, "Fractal Dimensions and Multifractility in Vascular Branching," *Journal of Theoretical Biology*, vol. 212, pp. 183–190, Sept. 2001.
[28] B. R. Masters, "Fractal Analysis of the Vascular Tree in the Human Retina," *Annual Review of Biomedical Engineering*, vol. 6, no. 1, pp. 427–452, 2004.
[29] S. Lorthois and F. Cassot, "Fractal analysis of vascular networks: Insights from morphogenesis," *Journal of Theoretical Biology*, vol. 262, pp. 614–633, Feb. 2010.
[30] A. W. Justin *et al.*, "Multi-casting approach for vascular networks in cellularized hydrogels," *Journal of The Royal Society Interface*, vol. 13, p. 20160768, Dec. 2016.
[31] D. M. Hoganson *et al.*, "Principles of Biomimetic Vascular Network Design Applied to a Tissue-Engineered Liver Scaffold," *Tissue Engineering Part A*, vol. 16, pp. 1469–1477, May 2010.
[32] R. Karch *et al.*, "A three-dimensional model for arterial tree representation, generated by constrained constructive optimization," *Computers in Biology and Medicine*, vol. 29, pp. 19–38, Jan. 1999.
[33] J. Nievergelt and P. Widmayer, "Spatial data structures: Concepts and design choices," in *Algorithmic Foundations of Geographic Information Systems*, Springer-Verlag Berlin Heidelberg, 1997.
[34] N. Beckmann *et al.*, "The R*-tree: An Efficient and Robust Access Method for Points and Rectangles," in *Proceedings of the 1990 ACM SIGMOD International Conference on Management of Data*, SIGMOD '90, (New York, NY, USA), pp. 322–331, ACM, 1990.
[35] W. Schreiner *et al.*, "Limited Bifurcation Asymmetry in Coronary Arterial Tree Models Generated by Constrained Constructive Optimization," The Journal of General Physiology, vol. 109, no. 2, pp. 129–140, Feb. 1997.
[36] S. A. Emamian *et al.*, "Kidney dimensions at sonography: correlation with age, sex, and habitus in 665 adult volunteers.," *American Journal of Roentgenology*, vol. 160, pp. 83–86, Jan. 1993.
[37] L. Zanoli *et al.*, "Renal artery diameter, renal function and resistant hypertension in patients with low-to-moderate renal artery stenosis," *Journal of Hypertension*, vol. 30, pp. 600–607, Mar. 2012.
[38] K. S. Satyapal *et al.*, "Morphometric analysis of the renal veins," *The Anatomical Record*, vol. 241, no. 2, pp. 268–272, 1995.
[39] F. Mookadam *et al.*, "Effect of positional changes on inferior vena cava size," *European Journal of Echocardiography*, vol. 12, pp. 322–325, Apr. 2011.
[40] G. Geleto *et al.*, "Mean Normal Portal Vein Diameter Using Sonography among Clients Coming to Radiology Department of Jimma University Hospital, Southwest Ethiopia," *Ethiopian Journal of Health Sciences*, vol. 26, p. 237, May 2016.
[41] K. Ishigami *et al.*, "Does Variant Hepatic Artery Anatomy in a Liver Transplant Recipient Increase the Risk of Hepatic Artery Complications After Transplantation?," *American Journal of Roentgenology*, vol. 183, pp. 1577–1584, Dec. 2004.
[42] N. Lal, "Ultrasonographic Measurement of Normal Common Bile Duct Diameter and its Correlation with Age, Sex and Anthropometry," *Journal of Clinical and Diagnostic Research*, 2014.
[43] D. B. Kolesky *et al.*, "3d Bioprinting of Vascularized, Heterogeneous Cell-Laden Tissue Constructs," *Advanced Materials*, vol. 26, no. 19, pp. 3124–3130, 2014.
[44] J. S. Miller *et al.*, "Rapid casting of patterned vascular networks for perfusable engineered three-dimensional tissues," *Nature Materials*, vol. 11, pp. 768–774, Sept. 2012.