# Crack Modeling via Minimum-Weight Surfaces in 3d Voronoi Diagrams 

Christian Jung ${ }^{1} \quad$ Claudia Redenbach ${ }^{1}$<br>${ }^{1}$ Technische Universität Kaiserslautern<br>Gottlieb-Daimler-Straße 47, 67663 Kaiserslautern, Germany


#### Abstract

Shortest paths play an important role in mathematical modeling and image processing. Usually, shortest path problems are formulated on planar graphs that consist of vertices and weighted arcs. In this context, one is interested in finding a path of minimum weight from a start vertex to an end vertex. The concept of minimum-weight surfaces extends shortest paths to 3d. The minimum-weight surface problem is formulated on a cellular complex with weighted facets. A cycle on the arcs of the complex serves as input and one is interested in finding a surface of minimum weight bounded by that cycle. In practice, minimum-weight surfaces can be used to segment 3d images. Vice versa, it is possible to use them as a modeling tool for geometric structures such as cracks. In this work, we present an approach for using minimum-weight surfaces in bounded Voronoi diagrams to generate synthetic 3d images of cracks.


Keywords: cracks, shortest paths, macrostructure modeling, minimum-weight surfaces, Voronoi diagram, integer programming, 3d images, microstructure modeling, adaptive dilation

## 1 Introduction

The shortest path problem (SPP) is formulated on a planar (directed) graph consisting of vertices and weighted arcs. Given a start and an end vertex, one is interested in finding a path of minimal weight connecting these vertices. Shortest paths represent a useful tool in 2d image processing in several contexts including image quilting [19] and image segmentation. In particular, shortest paths have been used to segment linear, 1d structures such as cracks in 2d images [2].
Minimum-weight surfaces were introduced in [21] and [15] as an extension of shortest paths to 3d. The minimum-weight surface problem (MSP) is formulated on a cellular complex consisting of vertices, arcs, weighted facets and cells. Here, the goal is to find a set of facets of minimal weight that is bounded by an input cycle. The voxel lattice in
a 3 d image can be interpreted as a cellular complex. In this context, minimum-weight surfaces have been used to segment flat structures in medical 3d images [15.
The segmentation of cracks in images of concrete is a broad field of research. A large variety of segmentation methods have been studied for 2 d and 3 d images. Comprehensive overviews and comparison studies in 2d and 3d can be found in 9 and [4, respectively. In image segmentation, ground truths are necessary for an objective output evaluation. Furthermore, they are prerequisite for training machine learning models such as convolutional neural networks [8] and random forests [7]. In 2d, annotated data is abundantly available, for example SDNET2018 [11. For 3d images, manual annotation is not feasible due to the large amount of data. Therefore, 3 d ground truth images are scarce and one has to rely on artificial data. Previously, the generation of synthetic cracks has been realized via fractional Brownian surfaces [1, 4]. However, the Hurst index - a measure of roughness - is the only parameter that can be used to control the shape of the surface. Therefore, this model is rather limited.
Voronoi diagrams have been used previously for simulating crack propagation via finite element methods [13, 16]. In this work, we propose a novel method to synthesize crack structures in 3d images using Voronoi diagrams generated from random point processes. The method includes two aspects:
First, we use minimum-weight surfaces as a tool to model the macrostructure of cracks. Bounded 3d Voronoi diagrams serve as the underlying cell complexes. This leaves several degrees of freedom such as the choice of the generating point processes, bounding cycles and the facet weights.
Second, the computed surfaces are discretized to 3d binary images. The rough microstructure of cracks is modeled via a second Voronoi diagram on a finer scale. Then, the cracks are embedded into patches of real computed tomography (CT) images of concrete. This paper is structured as follows. In Section 2, we outline the concept of shortest paths. Their extension to minimum-weight surfaces is described in Section 3. Section 4 comprises our crack modeling pipeline using Voronoi diagrams. It includes a description of our approach for macro- and microstructure modeling. Section 5 serves as conclusion and outlook to possible future research.

## 2 Shortest paths

Let $G=(V, A)$ be a directed graph consisting of a set of vertices $V$ and directed arcs $A \subseteq V \times V$. We use the notation $\alpha(a)$ for the start vertex and $\omega(a)$ for the end vertex of an $\operatorname{arc} a \in A$.
Furthermore, let $c: A \rightarrow \mathbb{R}_{>0}$ be a function assigning a non-negative weight to every arc in $A$. A path in $G$ is a finite sequence of vertices and arcs, $P=\left(v_{0}, a_{0}, v_{1}, a_{1}, \ldots, a_{k-1}, v_{k}\right)$, $k \geq 0$, with $v_{i} \in V$ and $a_{i} \in A$ with $\alpha\left(a_{i}\right)=v_{i}$ and $\omega\left(a_{i}\right)=v_{i+1}$ for $i=0, \ldots, k-1$. It is called a cycle (or closed contour) if no arc and no vertex is included more than once except for $v_{0}=v_{k}$. The weight of a path $P$ is given as $c(P)=\sum_{i=0}^{k-1} c\left(a_{i}\right)$.
Given two vertices $s, t \in V$, the SPP is looking for a path of minimum weight from $s$ to $t$. Many algorithms for solving the SPP exist, for example Dijkstra's 10 or Bellman's
and Ford's algorithm [5].
SPPs can also be formulated as binary integer programs [17, 22]. Note that this approach is less efficient than the ones described in [10] or [5]. However, it gives a good intuition for an analogue approach to compute minimum-weight surfaces which we describe in Section 3
Let $s$ be the start- and $t$ the end vertex of the path and let $x$ be a vector of binary variables $x_{i} \in\{0,1\}$ assigned to each arc $a_{i}$. The SPP can then be formulated as

$$
\begin{array}{ll}
\text { minimize } & \sum_{i: a_{i} \in A} c\left(a_{i}\right) x_{i} \\
\text { subject to } & B x=p \\
& x_{i} \in\{0,1\} . \tag{3}
\end{array}
$$

The vertex-arc incidence matrix $B$ and the vector $p$ in constraints (2) are given as

$$
B_{j, i}=\left\{\begin{array}{ll}
1 & \text { if } v_{j}=\alpha\left(a_{i}\right), \\
-1 & \text { if } v_{j}=\omega\left(a_{i}\right), \\
0 & \text { else }
\end{array} \quad \text { and } \quad p_{j}= \begin{cases}1 & \text { if } v_{j}=s \\
-1 & \text { if } v_{j}=t \\
0 & \text { else }\end{cases}\right.
$$

The constraints (2) are flow-conservation constraints. They ensure that every vertex that is part of the path is incident to exactly one incoming and one outgoing arc, except for the start- and end vertex. This results in the fact that any feasible solution must contain a path from $s$ to $t$ and, by the assumption that all costs are strictly positive, this ensures that the optimal solution is in fact a path without repeated vertices. The variables $x_{i}$ are binary by constraint (3) and indicate whether arc $a_{i}$ belongs to the path $\left(x_{i}=1\right)$ or not $\left(x_{i}=0\right)$. Finally, the objective in (1) is to minimize the weight over all paths from $s$ to $t$.
Note that, in case of negative arc costs, the problem of finding a shortest path without repeated vertices becomes NP-hard (which can be seen by a reduction from the Hamiltonian Path Problem [12]). Thus, additional constraints become necessary and, hence, we assume that all arc costs are strictly positive.

## 3 Minimum-weight surfaces

Minimum-weight surfaces have been presented in [21] and [15] as an extension of shortest paths to 3 d .
For our purposes, let $K=(V, A, F, C)$ be a cellular complex consisting of a set of vertices $V$, directed arcs $A \subseteq V \times V$, facets $F \subseteq A \times \ldots \times A$ and cells $C \subseteq F \times \ldots \times F$.
Further, let $w: F \rightarrow \mathbb{R}_{>0}$ be a function assigning a non-negative weight to every facet in $F$.
Note that $(V, A)$ defines a (directed) graph. Given a cycle $H$ on $(V, A)$, the MSP is looking for a connected set of facets in $K$ of mimimum weight that is bounded by $H$. To this end, arc directions may be assigned arbitrarily. Every facet is considered twice, once in clockwise and once in counterclockwise orientation. The orientation of $H$ must
be chosen to be either clockwise or counterclockwise. If the direction of arc $a$ coincides with the direction of its counterpart in an incident facet $f$ (or cycle $H$ ), we call $a$ and $f$ (or $a$ and $H$ ) coherent. If it does not coincide, we call them anti-coherent.
The MSP can be formulated as a binary integer program analogously to the one in Section 2, It is given as

$$
\begin{array}{ll}
\text { minimize } & \sum_{i: f_{i} \in F} w\left(f_{i}\right) y_{i} \\
\text { subject to } & D y=q \\
& y_{i} \in\{0,1\} . \tag{6}
\end{array}
$$

The arc-facet incidence matrix $D$ and the vector $q$ in constraints (5) are given as

$$
D_{j, i}= \begin{cases}1 & \begin{array}{l}
\text { if } a_{j} \text { and } f_{i} \text { are incident } \\
\text { and coherent, } \\
-1 \\
\text { if } a_{j} \text { and } f_{i} \text { are incident } \\
\text { and anti-coherent } \\
\text { else }
\end{array}\end{cases}
$$

and

$$
q_{j}= \begin{cases}1 & \begin{array}{l}
\text { if } a_{j} \text { is part of } H \\
\text { and coherent to } H
\end{array} \\
-1 & \text { if } a_{j} \text { is part of } H \\
\text { and anti-coherent to } H \\
0 & \text { else. }\end{cases}
$$

Similarly to the flow-conservation constraints in (2), constraints (5) ensure that we obtain a connected set of facets that is bounded by $H$. The variables $y_{i}$ are binary by constraint (6) and indicate whether facet $f_{i}$ belongs to the surface $\left(y_{i}=1\right)$ or not $\left(y_{i}=0\right)$. Finally, the objective in (4) is to minimize the weight over all possible surfaces that are bounded by $H$.
We always assume positive costs to ensure that the output is indeed a connected surface.

## 4 Crack modeling

Minimum-weight surfaces have been used for 3d image segmentation with the 3d grid being the underlying cell complex [15]. Reversing this approach, minimum-weight surfaces can also be used to model surface-like, connected structures such as cracks on a macroscopic level.
Here, we focus on minimum-weight surfaces in bounded 3d Voronoi diagrams. Our goal is to develop an approach for modeling 2d crack structures to generate semi-synthetic 3 d images of cracked concrete. Using this approach in combination with various point process models allows us to control the geometry of the resulting crack structure.

The most striking geometric characteristics of cracks in 3d CT images of concrete have been identified and discussed in [18]. In particular, we observe that 1. crack widths are varying and cracks may appear on multiple scales, 2 . crack surfaces are not totally smooth but rather rough due to the granularity of the concrete's cement matrix, 3 . when propagating through concrete, cracks may branch. An example is given in Fig. 1. These observations are used in the discretization procedure for modeling the cracks' microstructure.


Figure 1: 2d slice of a 3d CT image of cracked concrete of size $1000 \times 550 \times 880$ voxels with a voxel edge length of $22.7 \mu \mathrm{~m}$. The image stems from a normal-strength concrete sample that was exposed to a tensile test. Sample: Department of Civil Engineering, University of Kaiserslautern, Imaging: Fraunhofer ITWM, Kaiserslautern

### 4.1 Voronoi diagrams

Voronoi diagrams are defined on a metric space $(S, d)$ with Euclidean distance function $d$. Given a set of points $R=\left\{p_{1}, \ldots, p_{n}\right\} \subset S$ with $2 \leq n<\infty$, the Voronoi diagram generated by $R$ is given as $W=\left\{C_{1}, \ldots, C_{n}\right\}$ with

$$
C_{i}=\left\{x \in S \mid d\left(x, p_{i}\right) \leq d\left(x, p_{j}\right) \forall p_{j} \in R\right\} .
$$

The $C_{i}$ are called the cells of the Voronoi diagram.
Note that, if $S$ is not bounded, the Voronoi diagram contains cells of infinite size. In practice, it is often convenient to only consider bounded cells. Therefore, we restrict our attention to the bounded Voronoi diagram given by $W \cap Q=\left\{C_{1} \cap Q, \ldots, C_{n} \cap Q\right\}$ for some bounded region $Q \subset S$. Note that this operation yields additional vertices, arcs and facets on the boundary of $Q$ that belong to the bounded Voronoi diagram.

### 4.2 Minimum-weight Voronoi surfaces

Bounded Voronoi diagrams in 3d can be considered as a cellular complex. Therefore, given a cycle on the arcs of the cell complex induced by a 3d Voronoi diagram and weighted facets, we are able to compute a minimum-weight surface by solving the optimization problem given in Section 3. As a result, we obtain a connected set of facets that we call a minimum-weight Voronoi surface.

### 4.3 Crack generation

We propose the following method to simulate crack structures via minimum-weight Voronoi surfaces.

1. Define a cuboid $Q=\left[0, d_{1}\right] \times\left[0, d_{2}\right] \times\left[0, d_{3}\right] \subset \mathbb{R}^{3}$.
2. Compute a random point pattern $R \subset Q$ as a realization of some point process model.
3. Compute the Voronoi diagram generated by $R$, bounded by $Q$.
4. Define functions $c, w$ assigning a non-negative weight to each of the arcs and facets, respectively.
5. Choose a vertex on each of the four vertical edges of $Q$. Denote them by $u_{1}, u_{2}, u_{3}, u_{4}$. Compute shortest paths from $u_{1}$ to $u_{2}, u_{2}$ to $u_{3}, u_{3}$ to $u_{4}$ and $u_{4}$ to $u_{1}$, via Dijkstra's algorithm, only using arcs that lie on the boundary of $Q$. Denote the paths by $P_{1}, P_{2}, P_{3}, P_{4}$. Then, $H=\cup_{i=1}^{4} P_{i}$ is a cycle on the boundary of $Q$.
6. Compute a minimum-weight surface bounded by $P$ by solving the integer program from Section 3.

The concept is illustrated in Fig. 2. It shows a minimum-weight surface in a Voronoi diagram bounded by $Q=[0,1] \times[0,1] \times[0,1]$. Its generators are a realization of a Poisson point process in $Q$ of intensity 500 . Arcs and facets are both assigned unit weights. The approach above leaves us several degrees of freedom. Facet shape and variability can be controlled by choice of the underlying point process model, while the intensity of the generator process influences the mean facet size. Additionally, the size of the bounding cuboid, the input cycle and the weighting functions $c$ and $w$ can be varied. Minimum-weight surfaces for the same realization of a Poisson point process but different choices of input cycles are given in Fig. 3. Note that the input cycle is not restricted to lie on the boundary of a cube but may be chosen arbitrarily.
Moreover, Fig. 4 shows minimum-weight surfaces in Voronoi diagrams generated by Poisson point processes, Matérn cluster processes and regular processes obtained by a force-biased sphere packing [3, 6] with a volume fraction of $60 \%$. Arcs and facets were weighted by their lengths and areas, respectively. For the cluster process, we observe a bimodal distribution of the facet areas. The surfaces resulting from the regular model


Figure 2: Minimum-weight surface generation. Left: Poisson-Voronoi diagram bounded by a cube, middle: cycle on the boundary of the Voronoi diagram, right: minimum-weight surface.


Figure 3: Examples of minimum-weight surfaces in the same Voronoi diagram but with different choices of input cycles.
are far more homogeneous in facet size and shape than those obtained from Poisson point processes.
For the Voronoi diagram generation we use the $\mathrm{C}++$ library voro ++20 and for solving the integer program we use the C++ library GLPK [14.
The runtime for computing a minimum-weight surface depends mainly on the size of the arc-facet incidence matrix, that is, on the number of facets in the cell complex. For a Poisson Voronoi diagram in $[0,1] \times[0,1] \times[0,1]$ with intensity 500 the expected runtime is around 12 seconds. It is obtained on a Red Hat Enterprise Linux Workstation 7.9 with an $\operatorname{Intel}(\mathrm{R}) \operatorname{Xeon}(\mathrm{R})$ CPU E5-2680 v2 2.8 GHz ( 10 cores) and 125 gigabytes of RAM.

$\lambda=50$

$\lambda=100$

$\lambda=1000$

$\lambda=5000$

$\lambda=2, \mu=50, r=0.1$


$$
\lambda=5, \mu=100, r=0.1
$$


$\lambda=20, \mu=50, r=0.1$

$\lambda=50, \mu=100, r=0.1$

$\lambda=50$

$\lambda=100$

$\lambda=1000$

$\lambda=5000$

Figure 4: Minimum-weight surfaces in Voronoi diagrams bounded by $Q=[0,1] \times[0,1] \times$ $[0,1]$. The generators are realizations of Poisson point processes (first column) with intensity $\lambda$, Matern cluster processes (second column) with parent and daughter point process intensities $\lambda$ and $\mu$, respectively, and cluster radius $r$, and hardcore processes (third column) with constant radii, intensity $\lambda$ and $60 \%$ volume fraction.

### 4.4 Discretization

The method described in Section 4.3 outputs a set of vertices of convex facets. In this section, we describe a method to transfer this representation to a discrete image.
Let $I$ denote a 3d label image and $J$ a 3 d binary image, both of size $d_{1} \times d_{2} \times d_{3}$. The discretization procedure is given as follows.

1. Compute a minimum-weight Voronoi surface in a cuboid of size $d_{1} \times d_{2} \times d_{3}$.
2. Discretize the Voronoi diagram. For every voxel $(p, q, r)$ do: Set $I(p, q, r)=l$ if $(p, q, r)$ is contained in cell $l$.
3. Discretize the minimum-weight surface. For every two neighboring voxels $(p, q, r)$, $\left(p^{\prime}, q^{\prime}, r^{\prime}\right)$ (with respect to the 26-neighborhood) do: Set $J(p, q, r)=1$ if $I(p, q, r)=$ $j$ and $I\left(p^{\prime}, q^{\prime}, r^{\prime}\right)=k$ for generators $j, k$ whose cells share a facet that is part of the minimum-weight surface. Output $J$.

The procedure is visualized in Fig. 5. It yields a binary image whose foreground is a piecewise planar structure of constant width. However, real crack structures usually are far more complex as has been pointed out in Section 4. In the following, we propose techniques to account for these observations.


Figure 5: Slice view visualizing the discretization procedure. Left: Label image obtained from discretizing the Voronoi diagram. Red voxels indicate facets that are part of the minimum-weight surface. Right: Output binary image.

### 4.4.1 Adaptive dilation

Our goal is to model cracks of varying thickness. We propose a procedure to dilate the foreground of image $J$ : Every $x$-slice of $J$ is dilated separately and iteratively. We choose a quadratic structuring element of size $2 \times 2$. The number of iterations depends on a random walk with Bernoulli-distributed increments and index set $\left\{0,1, \ldots, d_{1}\right\}$. The increments are either 1 with probability $p$ or 0 with probability $1-p$. Thus, the crack thickness can be controlled via parameter $p$. The procedure is visualized in Fig. 6 for
different choices of $p$. Note that the random walk can be substituted by any suitable stochastic process, for example to produce decreasing crack widths.


Figure 6: Adaptive dilation applied to the input image with $p=0.02$ (left) and $p=0.05$ (right).

### 4.4.2 Microstructure modeling

In order to model the rough microstructure on the boundary of cracks, we compute a second (Poisson-) Voronoi diagram with a higher intensity than the one used for the computation of the minimum-weight surface. Then, for every foreground voxel in the dilated crack image $J$, we identify the Voronoi cell it is contained in. The whole cell is then discretized with voxel value 1 according to the approach described in Section 4.4. The procedure is visualized in Fig. 7. Afterwards, we apply a median filter to the resulting image.


Figure 7: Microstructure generation. Left: Discretized Poisson-Voronoi diagram and input crack (red). Right: Set of Voronoi cells (white) that are intersected by the input crack.

### 4.4.3 Crack branching

Crack branches emerge when a crack splits into two or more cracks. Often, the thickness of these branches lies in a range of 1-2 voxels.
Branching cracks can be modeled by combining two minimum-weight surfaces obtained from different cycles on the cuboid. If the underlying set of generators for the Voronoi diagram is identical, the surfaces may share several facets, see Fig. 8 .

### 4.4.4 Crack embedding

An approach for embedding crack structures in real 3d concrete images has been proposed in [4]. We extract image patches of the same size as the ground truth images from the real CT images. These patches are multiplied voxelwise with the inverse ground truth images. This leads to crack voxels having grayvalue 0 while the background does not change. Cracks and air pores both consist of air. Therefore, they should possess the same grayvalue distribution. We assume these grayvalues to be i.i.d. normally distributed. Mean and standard deviation are estimated via sample mean and sample standard deviation of the empirical distribution of air pore grayvalues. Then the crack voxels are simulated according to that distribution. To smooth the transition between background and crack, we apply a Gaussian filter to crack voxels and all voxels in their 26 -neighborhood. The final image together with its ground truth is given in Fig. 8, A corresponding 3d rendering is given in Fig. 9 .


Figure 8: Crack embedding. Left: Ground truth image after applying a median filter, right: synthesized crack image.

## 5 Conclusion

In this work, we have presented a novel method to generate artificial crack images. It includes the generation of a macrostructure via minimum-weight surfaces and a discretization procedure for generating its microstructure.
The shape and size of the output can be controlled by several parameters. Thus it allows for the generation of a wide range of surface structures.


Figure 9: 3d volume rendering of the simulated crack structure.

Our next steps include the generation of a full semi-synthetic data set for training machine learning models and evaluating segmentation methods on multiscale crack images. Furthermore, the model may be extended to include the grayvalue information of real concrete images. We can assume that cracks, when propagating through concrete, take the path of least resistance. Certain parts of the concrete mixture are less prone to cracking than other parts. In particular, this holds true for parts with a higher density. Therefore, facet weights may be derived from the mean voxel grayvalue in their vicinity.

Acknowledgements: This work was supported by the German Federal Ministry of Education and Research (BMBF) [grant number 05M2020 (DAnoBi)].
The authors would like to thank Prof. Dr. Sven O. Krumke for his helpful suggestions.

## References

[1] P. S. Addison, L. T. Dougan, A. S. Ndumu, and W. M. Mackenzie. A fractional Brownian motion model of cracking. In Paradigms Of Complexity: Fractals And Structures In The Sciences. Proc. Conf. Fractal, pages 117-123. World Scientific, 2000.
[2] Rabih Amhaz, Sylvie Chambon, Jérôme Idier, and Vincent Baltazart. Automatic crack detection on 2d pavement images: An algorithm based on minimal path selection. IEEE T. Intell. Transp., 17:2718-2729, 2016.
[3] Monika Bargieł and Jacek Mościński. C-language program for the irregular close packing of hard spheres. Computer Physics Communications, 64(1):183-192, 1991.
[4] Tin Barisin, Christian Jung, Franziska Müsebeck, Claudia Redenbach, and Katja Schladitz. Methods for segmenting cracks in 3d images of concrete: A comparison based on semi-synthetic images. Pattern Recognition, 129:108747, 2022.
[5] Richard Bellman. On a routing problem. Quarterly of Applied Mathematics, 16:8790, 1958.
[6] Alexander Bezrukov, Monika Bargieł, and Dietrich Stoyan. Statistical analysis of simulated random packings of spheres. Particle \&3 Particle Systems Characterization, 19(2):111-118, 2002.
[7] L Breiman. Random forests. Machine Learning, 45:5-32, 102001.
[8] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger. 3D U-Net: Learning dense volumetric segmentation from sparse annotation. In Lect. Notes Comput. Sc., pages 424-432, 2016.
[9] S. Chambon and Jean-Marc Moliard. Automatic Road Pavement Assessment with Image Processing: Review and Comparison. Int. J. Geoph., 2011:1-20, 2011.
[10] Edsger W. Dijkstra. A note on two problems in connexion with graphs. Numerische Mathematik, 1:269-271, 1959.
[11] Sattar Dorafshan, Robert Thomas, and Marc Maguire. Sdnet2018: An annotated image dataset for non-contact concrete crack detection using deep convolutional neural networks. Data in Brief, 21:1664-1668, 2018.
[12] Michael R. Garey and David S. Johnson. Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman \& Co., USA, 1979.
[13] E. Ghazvinian, M.S. Diederichs, and R. Quey. 3d random Voronoi grain-based models for simulation of brittle rock damage and fabric-guided micro-fracturing. J. Rock Mech. Geotech. Eng., 6(6):506-521, 2014.
[14] GNU Project. GNU Linear Programming Kit, Version 5.0, 2020. http://www.gnu. org/software/glpk/glpk.html.
[15] Leo J. Grady. Minimal surfaces extend shortest path segmentation methods to 3d. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32:321-334, 2010.
[16] Yiqun Huang, Shaowei $\mathrm{Hu}, \mathrm{Zi} \mathrm{Gu}$, and Yueyang Sun. Fracture behavior and energy analysis of 3d concrete mesostructure under uniaxial compression. Materials, 12(12), 2019.
[17] M.S. Ibrahim, N. Maculan, and M. Minoux. A strong flow-based formulation for the shortest path problem in digraphs with negative cycles. International Transactions in Operational Research, 16(3):361-369, 2009.
[18] Christian Jung, Franziska Müsebeck, Tin Barisin, Katja Schladitz, Claudia Redenbach, Martin Kiesche, and Matthias Pahn. Towards automatic crack segmentation in 3d concrete images. In 11th Conference on Industrial Computed Tomography (iCT), 2022.
[19] Lara Raad and Bruno Galerne. Efros and Freeman Image Quilting Algorithm for Texture Synthesis. Image Processing On Line, 7:1-22, 2017.
[20] Chris Rycroft. Voro++: A Three-Dimensional Voronoi Cell Library in C++. Chaos (Woodbury, N.Y.), 19:041111, 122009.
[21] John M. Sullivan. A crystalline approximation theorem for hypersurfaces. PhD thesis, Princeton University, 1990.
[22] Leonardo Taccari. Integer programming formulations for the elementary shortest path problem. European Journal of Operational Research, 252(1):122-130, 2016.

