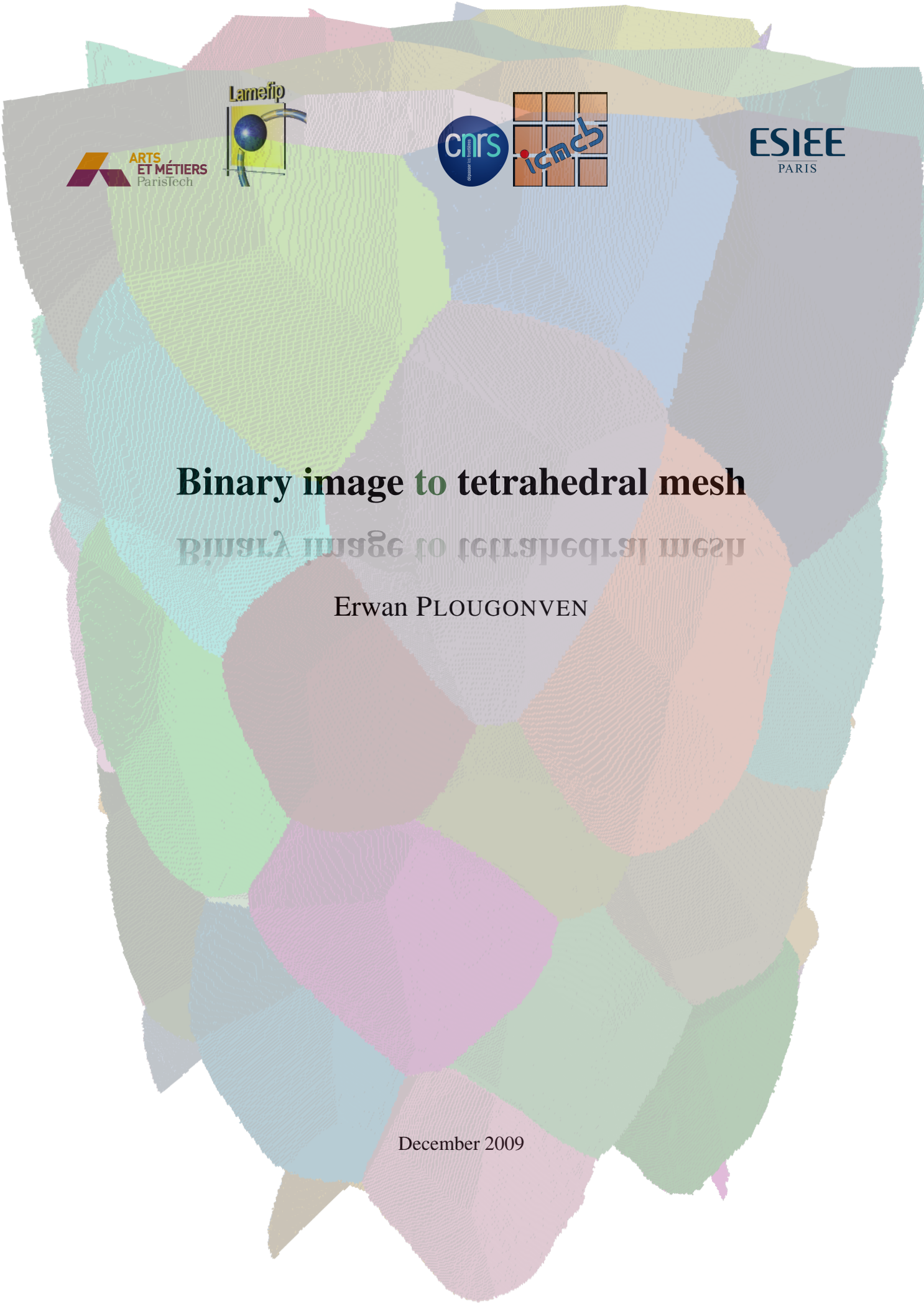


Binary image to tetrahedral mesh

Erwan PLOUGONVEN

December 2009

This project, initiated by Philippe Viot of the Laméfip (LAboratoire Matériaux Endommagement Fiabilité et Ingénierie des Procédés) at ENSAM Bordeaux, and in collaboration with Dominique Bernard at the icmcb (Institut de Chimie de la Matière Condensée de Bordeaux) and John Chaussard, Michel Couprie and Hugues Talbot of the A³SI laboratory (Algorithmes, Architectures, Analyse et Synthèse d'Images) of ESIEE Paris (Ecole Supérieure d'Ingénieurs en Electronique et Electrotechnique) consists in the study of the deformation of polymeric foam under dynamic compression, also called crash loading, through the use of a non-destructive imaging technique, X-ray absorption contrast microtomography. The polypropylene foam that was used is a multi-scale material, microscopic bubbles contained in millimetric beads, and the challenge in this study, excluding the particulars of the experimental procedure, is to consider the deformation at the scale of the beads [Viot et al., 2007]. To accurately model this deformation, each bead must be properly segmented, its boundary triangulated at a coarse resolution yet still precisely mapping its physical location.

The first section presents past work on this material, which focused on tracking the bead centers and determining an central interior volume [Plougonven et al., 2006] for average density computation. The following sections presents the work of John Chaussard on the foam bead boundary triangulation procedure and its transfer to reusable computation modules integrated in Avizσ®.

CONTENTS

1 Background **5**
- 1.1 Objectives . . . 5
- 1.2 Experiment . . . 5
- 1.3 Initial image processing . . . 6
  - 1.3.1 Obtaining the bead centres . . . 6
  - 1.3.2 Obtaining the bead boundaries . . . 7

2 Introduction **11**
- 2.1 What is a cubical complex ? . . . 11
- 2.2 Conversion from a binary image . . . 11
- 2.3 Internal structure of a cubical complex . . . 12

3 The collapse **15**
- 3.1 Free pairs and elementary collapse . . . 15
- 3.2 Control of the collapsing order . . . 15

4 Surface separations **17**
- 4.1 Surface intersections . . . 17
- 4.2 Importance of the collapsing order . . . 17

5 Surface meshing **21**
- 5.1 Decomposition of the intersecting edges . . . 21
- 5.2 Projection onto a plane and surface meshing . . . 21
- 5.3 Backprojection in the 3D space . . . 21

1 BACKGROUND

DYNAMIC COMPRESSION OF POLYPROPYLENE FOAM SAMPLES

1.1 Objectives

The main objective is to be able to predict the deformation of polypropylene foam at the scale of the beads, and validate the model with experimental measurements. By using microtomography to follow the compression of a foam sample (through interrupted crash loadings), the images are used to extract an physically representative mesh for the finite element model, defining specific properties for the boundaries of the foam beads.

The difficulty lies in the fact that this material is multiscale and requires elaborate image processing in order to segment the beads and generate a useable mesh. The microtomograms contains several billion pixels, and as many elements in the finite element model is out of the question. For this reason, we focus on the bead scale, and attempt to fit a model of lesser resolution, yet still accurately defining the beads volumes.

1.2 Experiment

The device used to apply the loading consists in a 0.37 kg cylindrical impactor, guided in a rectified metallic tube 1.6 meters high. The tube is placed above the measurement module, which consists in an aluminium base, a die-punch set and sensors for displacement and force. The sample is placed inside the die and compressed by the punch (c.f. figure 1.1). The impactor is dropped down the tube and hits the punch which compresses the sample.

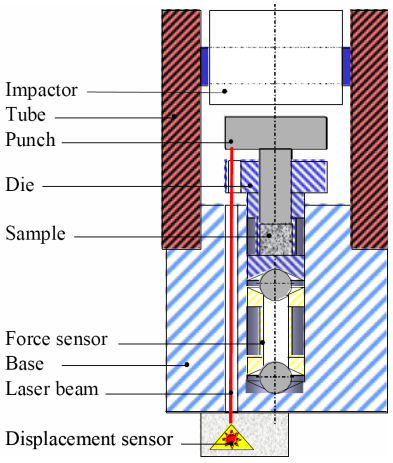
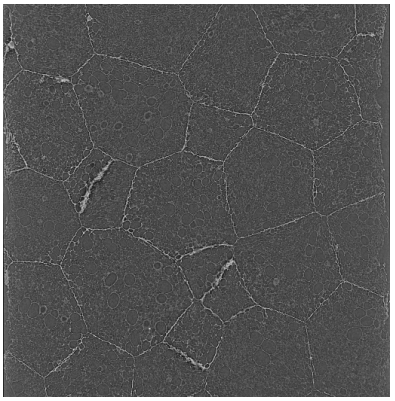
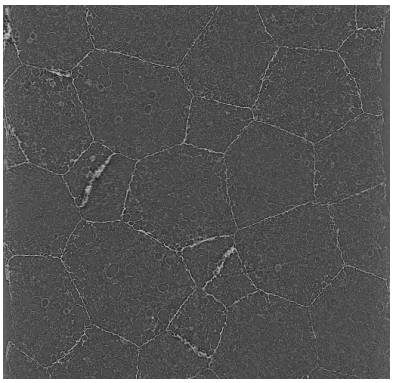


Figure 1.1: Experimental setup for the dynamic loading

After an impact, the punch shoulder stays against the die to keep the sample compressed during the tomographic measurement. The acquisitions have been obtained on the BM05 beam line at the European Synchrotron Radiation Facility (ESRF) in Grenoble (France). The acquired projections are 2028x2048 pixels radiograms, with a pixel corresponding to 4.91 µm. The sample of polypropylene foam is 10 mm in diameter, and the energy of the beam was set to 16 keV. Figure 1.2 shows cross-sections parallel to the loading axis for the four loading stages.



(a) Initial state (10 mm)



(b) First compression (9 mm)

(c) Second compression (7 mm)

(d) Third compression

(e) Fourth compression

Figure 1.2: Vertical cross-sections of the images for the different loading stages.

1.3 Initial image processing

Previous work focused on extracting an interior volume of each bead of the polypropylene foam sample. Some precautions were taken for the required image processing, as bead boundaries are not immediately obvious (c.f. figure 1.3).

1.3.1 Obtaining the bead centres

Figure 1.4 shows the sequence of operations that has allowed to extract one connected component per bead, by thresholding the distance transform from figure 1.4(h) to an arbitrary value of 20.

After a closest neighbours approach to track individual beads from one image to the other, a simple active surface algorithm was implemented to determine an interior volume to each of the beads, shown in figure 1.5.

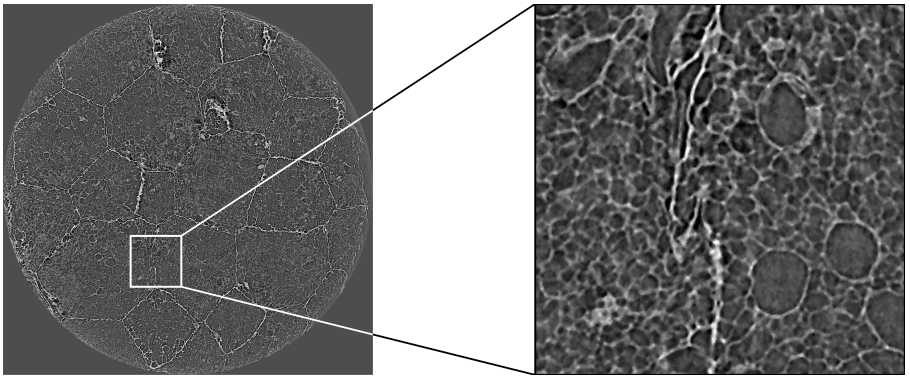
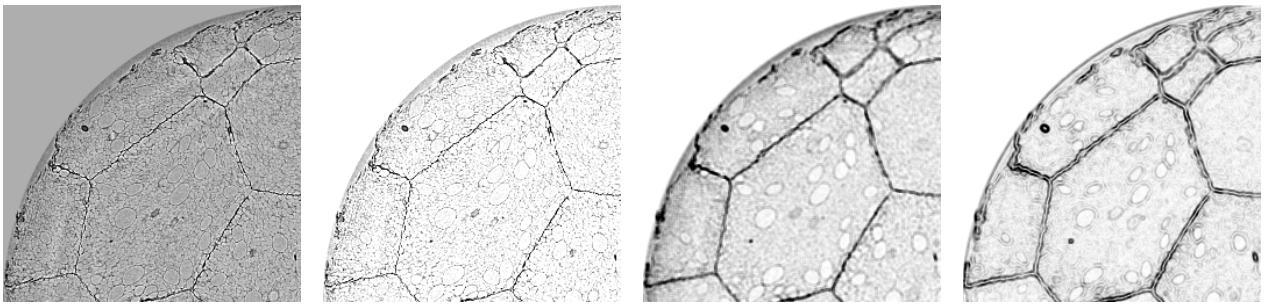
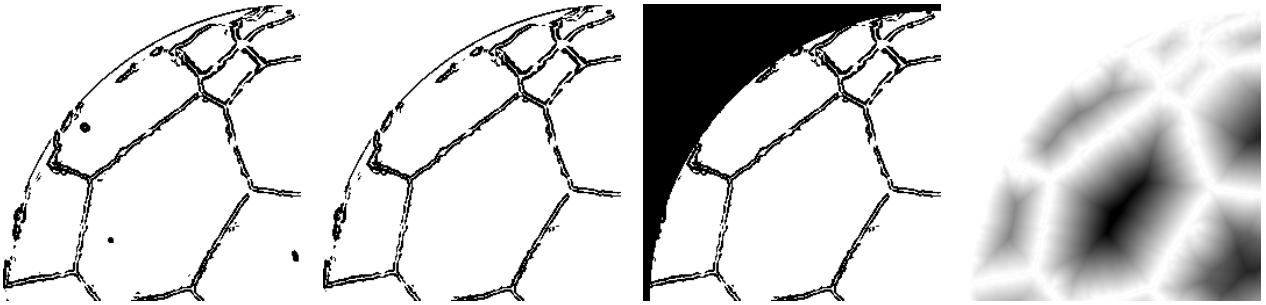


Figure 1.3: Cross-section of the microtomogram of the foam sample at the initial state, perpendicular to the compression axis, with a zoom on a boundary.



(a) 4^3 subsampling (b) Phase contrast reduction (c) Gaussian filter (d) Norm of the Sobel gradient



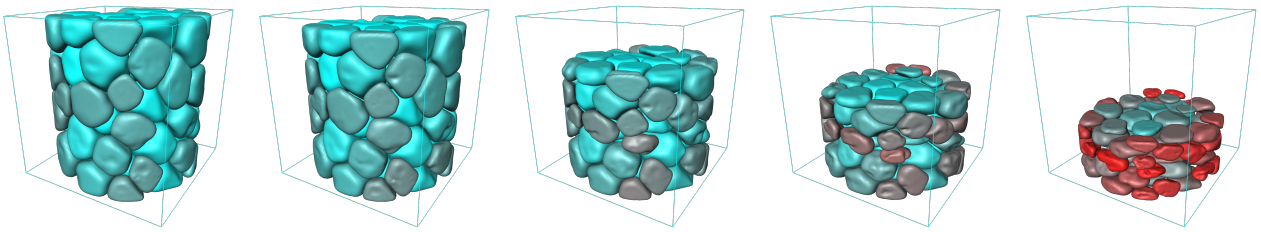
(e) Hysteresis thresholding (f) Largest connected component (g) Circular mask (h) Distance transform

Figure 1.4: Image processing for extraction bead centres: the last image is thresholded and the centres of mass of the largest connected components of that image are defined as bead centres.

1.3.2 Obtaining the bead boundaries

The interior volumes of each bead is defined by a triangulated surface. These surfaces are not in contact with each other, and a significant intersticial volume remains. Since the bead boundaries are thin, this segmentation is insufficient to locate them, which is why a thinning procedure of the intersticial volume is performed.

The first step is to return to a 3D image data structure from the triangulation. This process consists in marking the pixels that intersect the surface triangles, followed by a hole-filling algorithm to mark all



(a) Initial state (b) 1st compression (c) 2nd compression (d) 3rd compression (e) 4th compression

Figure 1.5: Resulting active surfaces for the different loading stages. The colouring is made according to the average density in the volumes delimited by the surfaces.

the interior pixels. The complementary of this set of pixels, shown in figure 1.6, defines the interstitial volume. Note that in the figure, a circular mask was applied, as in figure 1.4(g), to avoid examining outside of the sample.

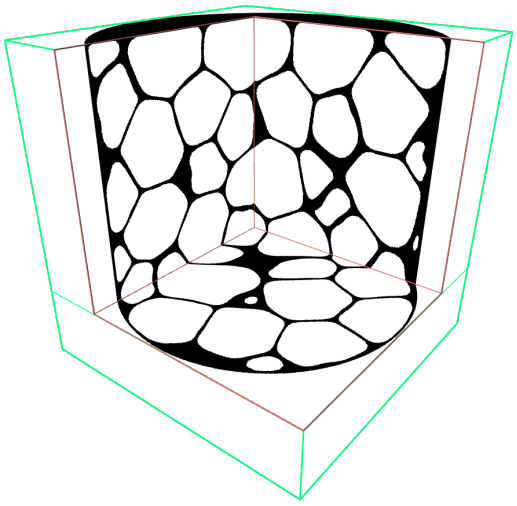
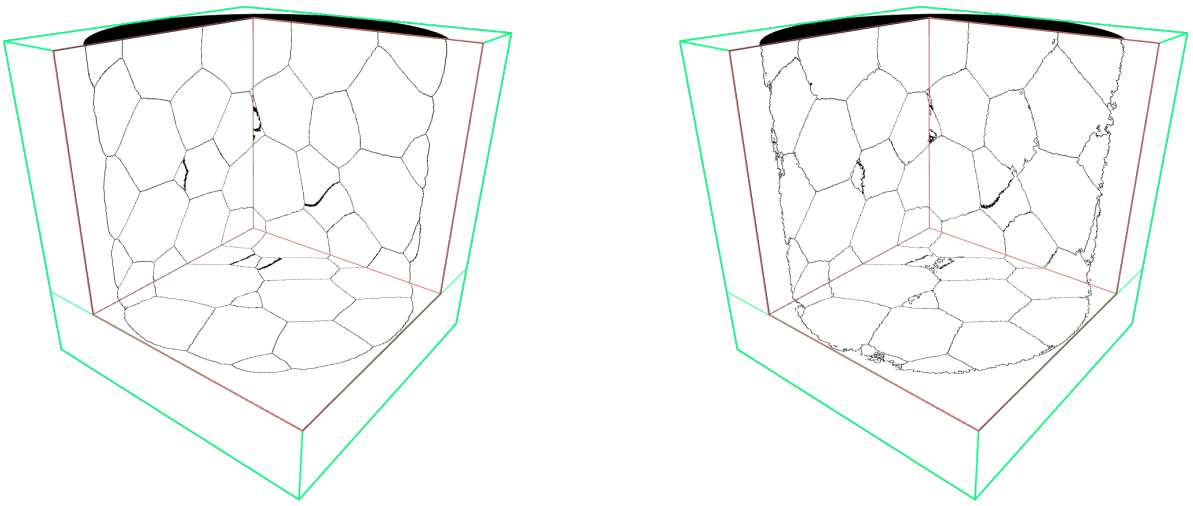


Figure 1.6: Resulting active surfaces for the different loading stages. The colouring is made according to the average density in the volumes delimited by the surfaces.

The thinning procedure is based on distance-ordered homotopic thinning [Pudney, 1998]. This algorithm typically uses as a priority function the distance transform of the object to produce centred skeletons. Applied to the interstitial volume, such a skeleton is illustrated in figure 1.7(a). Although this produces a smoother object, our requirement is that the skeleton match as close as possible the bead boundaries. We use, instead of the distance transform, the blurred gradient of the image as a priority function, insuring that the pixels of highest gradient should be processed last. The resulting skeleton, shown in figure 1.7(b), although of less pleasing appearance better represents the physical boundaries of the beads.

Since the objective is to generate a finite element model, the purpose of this skeleton is to help in generating a triangulated version of the bead boundaries, and for each bead volume, generate the 3D interior mesh from its boundary. As mentioned before, using the pixel resolution produces too many elements, therefore the system must be downsampled. Furthermore, these shell elements that define bead boundaries must mutually coincide, as every part of this boundary separates at least two beads. The idea is then to partition this skeleton, each partition defining a separation between two



(a) (b)

Figure 1.7: Conversion of the active surface triangulations to a 3D binary image, by first detecting the pixels intersecting the triangles, filling the closed surfaces, and applying a cylindrical mask. The intersticial volume is shown in black.

specific beads. These partitions are joined by intersecting edges, that surround each partition. Using this structure, each partition will be triangulated independently from the intersecting edges, then reassembled together for the final mesh.

This methodology assumes that the portions of the skeleton are identifiable, but contrary to intuition, in a 2D discrete skeleton, thick regions can remain at the surface intersection, making it impossible to properly delimit each partition. A truly thin representation is needed in order to decompose the bead boundaries, and for this reason the framework of cubical complexes is used.

We can therefore summarise the techniques presented here as a conversion process from the binary image of the bead boundary skeleton to a downsampled triangulated mesh for the finite element modelling. Presented in this report is the transfer of John Chaussard's implementation of this conversion into user-friendly modules integrated in **Avizo**®.

2 INTRODUCTION

FROM BINARY IMAGE TO CUBICAL COMPLEX

2.1 What is a cubical complex ?

To decompose the skeleton of the foam bead boundaries, we use the framework of cubical complexes (a type of abstract simplicial complex), also called cellular complex [Kovalevsky, 1989]. The following notations and definitions are directly taken from [Bertrand and Couprie, 2006].

Consider the families of sets $\mathbb{F}_0^1 = \{\{a\} : a \in \mathbb{Z}\}$ and $\mathbb{F}_1^1 = \{\{a, a+1\} : a \in \mathbb{Z}\}$. We call an m-face f of $\mathbb{Z}^n$ (with $0 < m \leqslant n$) the Cartesian product of m elements of $\mathbb{F}_1^1$ and $(n-m)$ elements of $\mathbb{F}_0^1$, and m is called the *dimension of* f, noted $dim(f)$. Let $\mathbb{F}^3$ be the set of all m-faces of $\mathbb{Z}^3$. In $\mathbb{F}^3$, m-faces are either *vertices*, unit *edges*, unit *squares* or unit *cubes* (for $m = 0, 1, 2, 3$ respectively).

Let f be a face in $\mathbb{F}^3$. We set $\widehat{f} = \{g \in \mathbb{F}^3 | g \subseteq f\}$ and $\widehat{f}^* = \widehat{f} \setminus \{f\}$. Any $g \in \widehat{f}$ is a *face of* f, and any $g \in \widehat{f}^*$ is a *proper face of* f.

Let X be a finite set of faces in $\mathbb{F}^3$. We call the *closure* X^- of X the set $X^- = \cup\{\widehat{f} | f \in X\}$. A set $X \subset \mathbb{F}^3$ such that $X = X^-$ is called a *complex*, or more specifically a *cubical complex* when in $\mathbb{F}^3$. A subset Y of a complex X that is also a complex is called a *subcomplex of* X, and is written $Y \preceq X$.

Let $X \preceq \mathbb{F}^3$. A face $f \in X$ is *principal for* X if there is no $g \in X$ such that $f \in \widehat{g}^*$. We denote by X^+ the set of all principal faces of X.

Let $X \preceq \mathbb{F}^3$, $dim(X) = \max\{dim(f) | f \in X^+\}$ is the *dimension of* X, and we say that X is an m-complex if $dim(X) = m$. Furthermore, we say X is *pure* if $\forall f \in X^+, dim(f) = dim(X)$.

2.2 Conversion from a binary image

A binary image is an application I from a subset Ω of $\mathbb{Z}^3$ onto the set $\{0, 1\}$. The subset generally takes the form $[0; w[\times[0; h[\times[0; d[$, where w, h, and d are respectively called the *width*, *height*, and *depth* of the image. We define the set of all pixels p, or elements of Ω, such that $I(p) = 1$ as the object X, or foreground. Its complementary is noted $\overline{X}$.

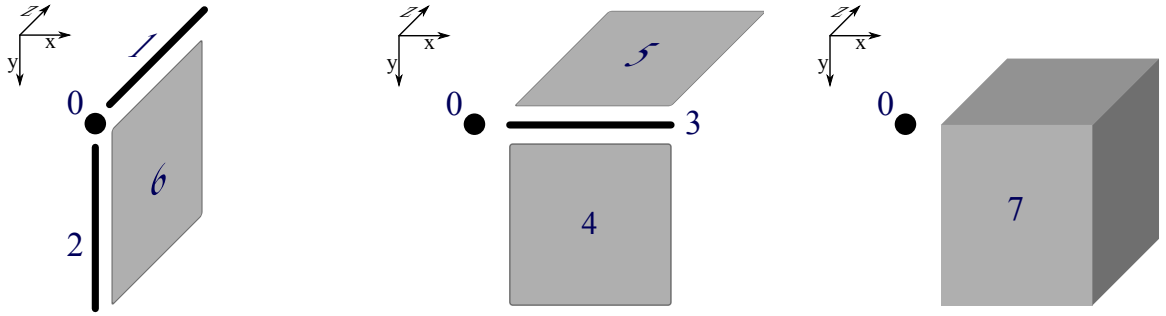
In the source code, such an image is represented as a memory array I of size $N = w * h * d$, and the pixels are stored in column-major order [Knuth, 1997]: a pixel $p \in \Omega$ is defined as a 3-tuple $\{i, j, k\}$ such that $i \in [0; w[$, $j \in [0; h[$, and $k \in [0; d[$, and its value $I(p)$, or intensity, is stored in I at the memory-offset $i + w.(j + h.k)$. Since we are defining binary images, the value is binary, but typically, the array I is never an array of bits, but an array of bytes (8 bits).

The conversion process generating a cubical complex, which we will call X, is implemented in the **Avizo**®module `Cubical Complex - Convert Image`. The result is a pure complex of dimension 3, such that each pixel in X has a corresponding principal 3-face in X.

2.3 Internal structure of a cubical complex

In a similar convention, the cubical complex is stored in memory as an array of bytes. The 3-face f that represents the pixel p in I will have the same coordinates $\{i, j, k\}$ in the array, that is these coordinates will point to the *byte* in which the *bit* that represents f is stored. In each byte of the array, the n^{th} bit (with $n \in [0; 7]$) is always of the same dimension, and the rule is as follows.

Each byte contains 8 faces, which are always on the back bottom left side (i.e. lower i, j, and k coordinates). Bit 0 is the back bottom left 0-face, or corner vertex, bits 1, 2, and 3 are the 1-faces or edges containing this corner, bits 4, 5, and 6 are the 2-faces containing respectively these edges, and bit 7 is the 3-face or volume. Figure 2.1 shows these faces.



(a) (b) (c)

Figure 2.1: Bit structure of each byte in the array storing the cubical complex, from bit 0 to bit 7, which is the 3-face. Bit 0 is displayed as a reference in the three subfigures. Note that not all the proper faces of the 3-face are stored in one byte (7 out of 26).

The following table also shows the structure of one byte of the array:

Bit	0	1	2	3	4	5	6	7
Name	CC_PT	CC_AZ	CC_AY	CC_AX	CC_FXY	CC_FXZ	CC_FYZ	CC_VOL
Face								

All the proper faces of a 3-face f are obviously not contained in the byte that stores f, but also in the bytes to the right, top and front neighbouring bytes (i.e. $i+1$, $j+1$, $k+1$, or any combination, making a total of 7 neighbouring bytes). Therefore on the image borders on those sides, in order to keep the same bit sequence for identifying the faces, there is a need for additional bytes, one layer on these three sides. Therefore the array is of size $(w+1)*(h+1)*(d+1)$. In these added byte layers, many of the bits are out of the bounds of the image, and will not be used. Figure 2.2 illustrates the simplest example of conversion, on an image of size $1*1*1$, in which the only pixel belongs to X. The resulting cubical complex array created has a size of $2*2*2$, and if we want to access, for instance, the 1-face of highest y- and z-coordinate, which is stored in the byte at coordinates $(0, 1, 1)$, then the memory offset will be $0 + (w+1)*(1 + (h+1)*1)$.

Figure 2.3 illustrates this conversion on a cropped portion of the foam bead boundaries. Of course it is difficult to display every face of the cubical complex, but it should resemble the set of cubes of figure 2.3(b). Note that since the array size of the cubical complex is slightly bigger ($(w+1)*(h+1)*(d+1)$), a scaling of the *displayed* cubes must be done in order to have fit in the same bounding box as the

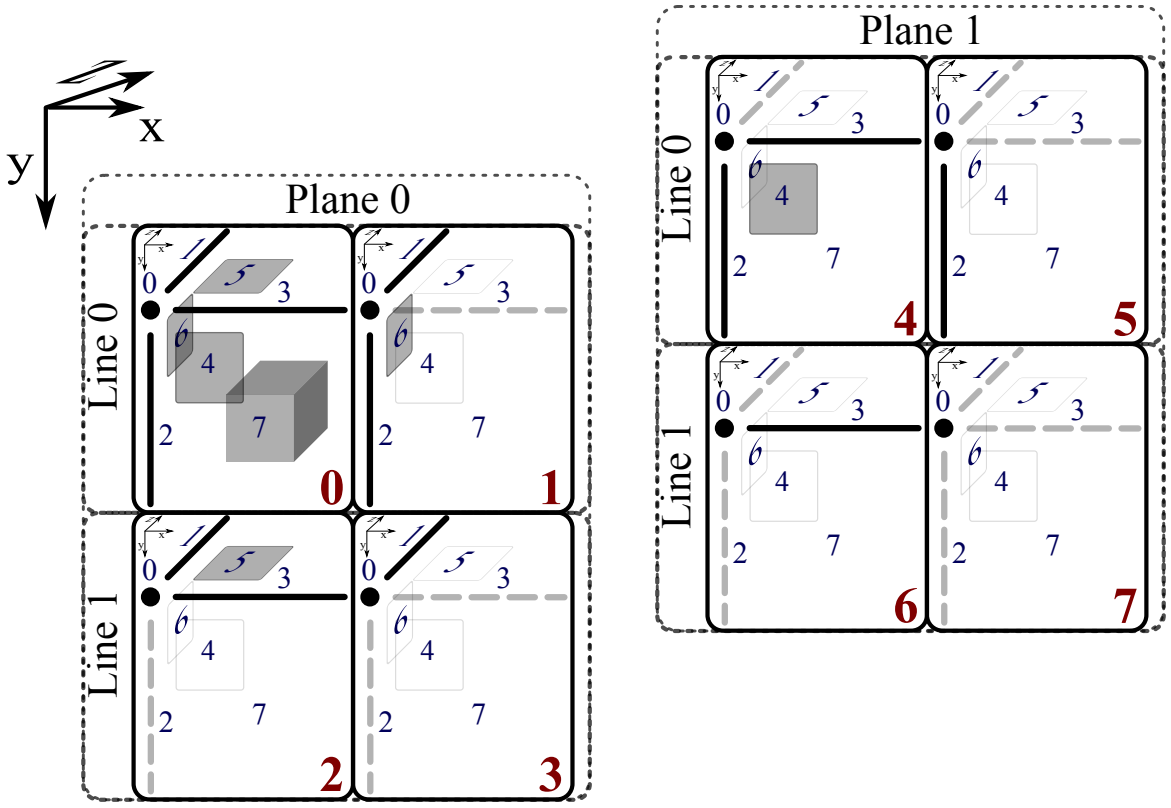
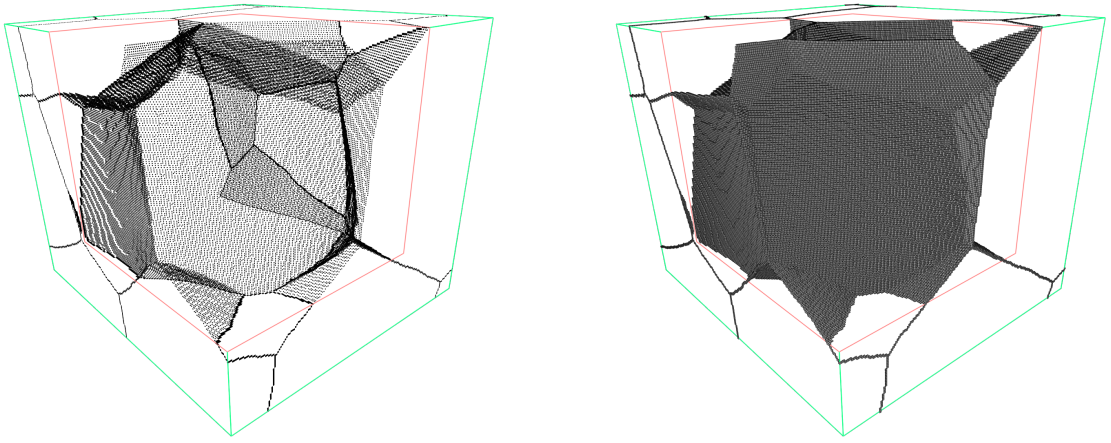


Figure 2.2: Diagram representing the structure of the byte array that stores the cubical complex, showing the association of each face in an image composed of one pixel, i.e. a 3D image of size 1x1x1. The solid-lined boxes represent one byte, the number in blue the bits of the corresponding byte, the numbers in red the index of the byte in the array. This simple example is the worst case scenario for memory usage, but when examining images of relatively large sizes, adding a layer of byte on the three far sides of the bounding box becomes negligible.

original image (i.e. $(\frac{w}{w+1}, \frac{h}{h+1}, \frac{d}{d+1})$). Any cubical complex, or even a set of faces, can be displayed with the module `Draw Cubical Complex` (it actually draws only the principal faces of the set).

The generated cubical complex still contains volumic portions, or 3-faces, therefore the next stage is concerned with obtaining a truly thin object, i.e. a pure complex of dimension 2. The following section describes this stage, and is called the collapse.



(a) Binary image (b) Cubical complex

Figure 2.3: Illustration of the conversion of a binary image to a cubical complex. The binary image is a cropped portion of the smooth skeleton shown in figure 1.7(a). (a) shows the pixels of the object, as a set of points on the cubic grid, while (b) shows the cubical complex, as a set of cubes (made up of 3-, 2-, 1-, and 0-faces).

3 THE COLLAPSE

SKELETONISATION EQUIVALENT WITH CUBICAL COMPLEXES

This section describes how to homotopically obtain, from the pixel-thin representation of the bead boundaries, an even thinner separation, i.e. a truly 2-dimensional object. Using the cubical complex framework and the collapse procedure, we do just that, the result being a pure complex of dimension 2.

3.1 Free pairs and elementary collapse

The following additional definitions are again directly taken from [Bertrand and Couprie, 2006]:

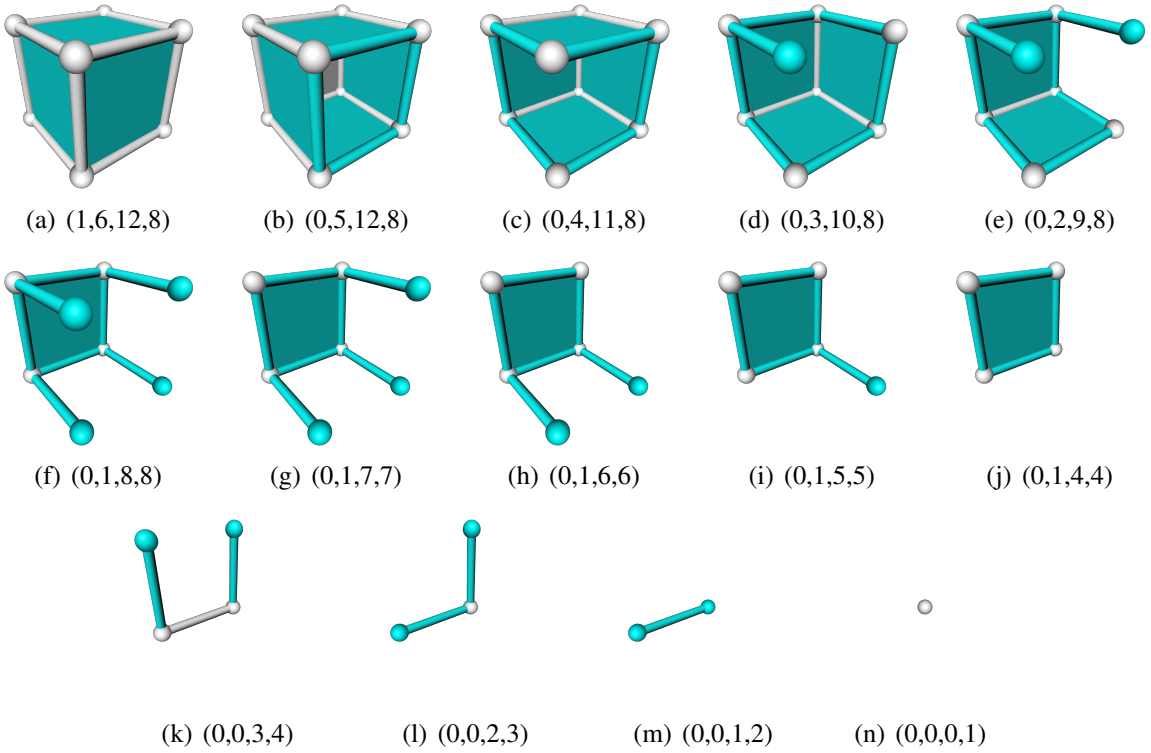
Let $X \preceq \mathbb{F}^3$, and $f \in X^+$. We say that f is a *border face for* X if there exists a proper face g of f such that f is the only face which contains g. Such a face g is said to be *free for* X and the pair (f, g) is said to be *a free pair for* X. The complex $X \setminus (f, g)$ is called an *elementary collapse of* X. Figure 3.1 illustrates this procedure on the simplest cubical complex converted from a binary image, i.e. one having only one principal 3-face. The figure shows one possible sequence of elementary collapses, until no more free pairs exist, i.e. only one 0-face remains.

3.2 Control of the collapsing order

Note that in most cases, there is more than one sequence of elementary collapses. In order to preserve centredness to this thinning procedure, and to avoid *square wave* artifacts along a surface, a directional strategy is put in place: firstly the border 3-faces are scanned, the free 2-faces of a specific orientation are selected first and those pairs removed. Upon removal of these, free pairs of lesser dimension are created (as in the removal of the 3-face in figure 3.1(b)), and are removed next (steps (c) to (f) in figure 3.1). Once all these new free pairs of (2-faces, 1-faces) are removed, more free pairs of lesser dimension can be created, and are thus removed next (steps (g) to (j) in figure 3.1). This accounts for the first direction of collapse. The second direction reconsiders border 3-faces, this time with a 2-face of another orientation, and removes successively free pairs of decreasing dimension. The collapse through all 6 directions (left, right, up, down, backward, forward) is called one iteration, and unless otherwise instructed, the collapse will reiterate on the first orientation again, and so forth until no more free pairs exist.

This process is implemented in the Avizo® module `Cubical Complex - Collapse`, with the collapse type called `Directional`. In this module, if the number of iterations is left at 0, then the collapse will reiterate until no more free pairs are available. Furthermore, the `Anchor border` forbids the removal of the faces on the image boundary.

Figure 3.2 shows the sequence of elementary collapses to *thin out* the pixel-thick foam bead boundaries.

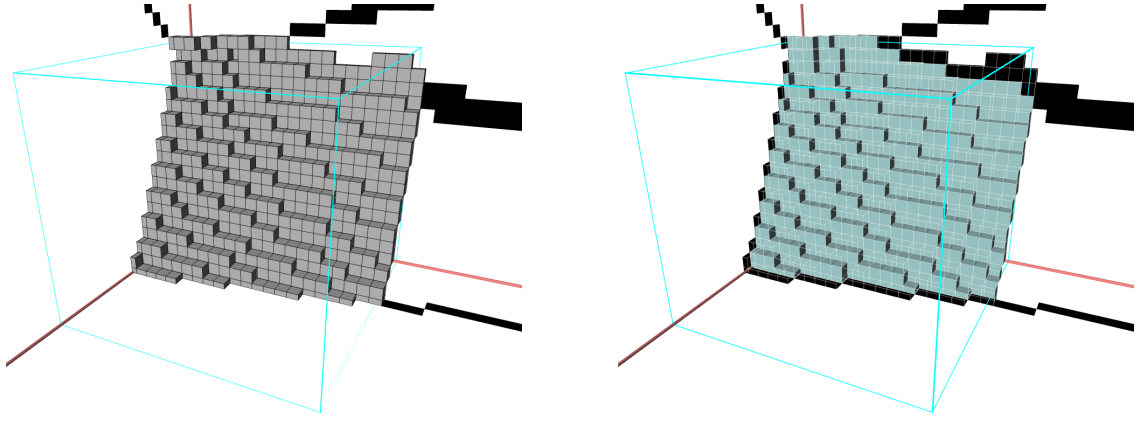


(a) (1,6,12,8) (b) (0,5,12,8) (c) (0,4,11,8) (d) (0,3,10,8) (e) (0,2,9,8)

(f) (0,1,8,8) (g) (0,1,7,7) (h) (0,1,6,6) (i) (0,1,5,5) (j) (0,1,4,4)

(k) (0,0,3,4) (l) (0,0,2,3) (m) (0,0,1,2) (n) (0,0,0,1)

Figure 3.1: Illustration of elementary collapses on a cubical complex with one principal face, of dimension 3. Each figure shows the free pairs of the cubical complex (marked in blue), and the series show successive removal of one of these pairs. In (a), the border face is the 3-face, not shown. Note that in (b), the 2-face in the background is not a border face (therefore not part of a free pair). The captioned numbers give the numbers of 3-, 2-, 1-, and 0-faces respectively that remain in the complex.



(a) Portion of the bead boundary cubical complex (b) Remaining 2-faces after elementary collapses

Figure 3.2: Elementary collapses on part of the bead boundary: (a) shows the cubical complex, pure and of dimension 3. Note that all 3-faces are border faces, and the visible 2-faces (excluding those on the subvolume border) are free faces. After a sequence of removal of free pairs, beginning with (3-face,2-face) pairs, then (2-face,1-face) and finally (1-face,0-face) pairs, we obtain the subcomplex shown in (b), again pure but of dimension 2.

4 SURFACE SEPARATIONS

DETECTION OF SURFACE INTERSECTIONS

The boundaries of the foam beads can be divided into distinct parts, one part separating two specific beads. Each part will be refered to in this section as a surface, and the objective is to decompose the boundary into a set of such surfaces, to be processed independently.

4.1 Surface intersections

Thanks to a thin representation of the foam bead boundaries with the collapse of the cubical complex, these surfaces can be detected by eximining the 1-faces: inside a surface, all 1-faces are proper faces of exactly two 2-faces, while on surface intersections, the 1-faces are proper faces of at least three 2-faces. Thanks to this simple property, the cubical complex can be decomposed into distinct surfaces.

Figure 4.1 shows the 1-faces that are detected with this property, effectively decomposing the cubical complex.

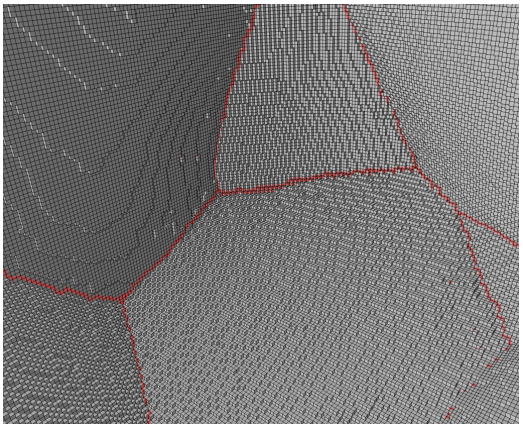


Figure 4.1: Illustration of the decomposition of the surfaces with the intersecting 1-faces, marked in red. As is illustrated on a portion of the foam bead boundary, the intersecting 1-faces produce loops, completely surrounding each surface individually, making the decomposition quite natural.

4.2 Importance of the collapsing order

If the order in which the collapsing process is not properly controlled, then a situation known as *surface pinching* occurs. A simple illustration of this is presented in figure 4.2.

Inside surfaces, surface pinching is actually irrelevant, because it concerns only isolated 1-faces, and the subsequent surface decomposition is oblivious to these. The problem lies near the surface intersections, where these 1-faces are not isolated, and can result in small loops which imply erroneous surfaces of only a few 2-faces.

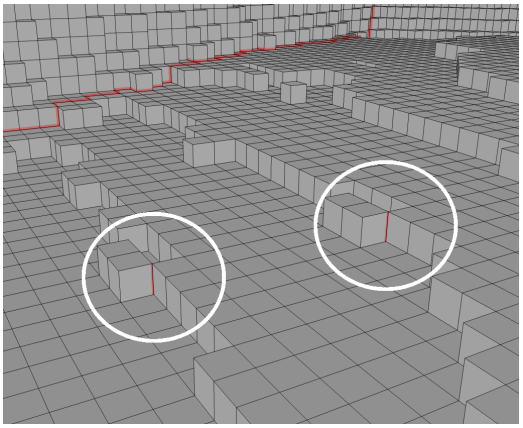


Figure 4.2: Illustration of surface pinching: in the foreground, two isolated 1-faces in red are contained in four 2-face of a surface, because the four 3-faces that originally contained it were collapsed in alternating directions.

The developed procedure to minimise surface pinching reexamines the collapse order near these surface intersections. This naturally requires that a first collapse has been performed (using the method described in section 3). Using the resulting cubical complex, it detects the intersecting 1-faces and reinserts their containing 3-faces into the cubical complex. A second collapse procedure is then applied: it examines each 3-face separately, and considers each direction of collapse. For every direction, a potential collapse is performed, and the resulting number of intersecting 1-faces is determined. Once this is done for all directions, the one that produces the least amount of intersecting 1-faces is chosen, and the collapse in that direction is validated.

This method is found in `Cubical Complex - Collapse`, using type `Min Intersect` and attaching the previous collapse result on the port `Surfaxis`. With this improved collapse, the chance of producing small cycles of intersecting 1-faces near surface intersections is reduced. The process of extracting one surface, i.e. a connected set of 2-faces, from this cubical complex consists in a propagation technique, from 2-face to connecting 2-faces, provided they are not connected by an intersecting 1-face. The module that performs this decomposition is called `Cubical Complex - Decompose`, and has the option of extracting the intersecting edges. As for the slider, it allows to select which surface to extract. An example is presented in figure 4.3.

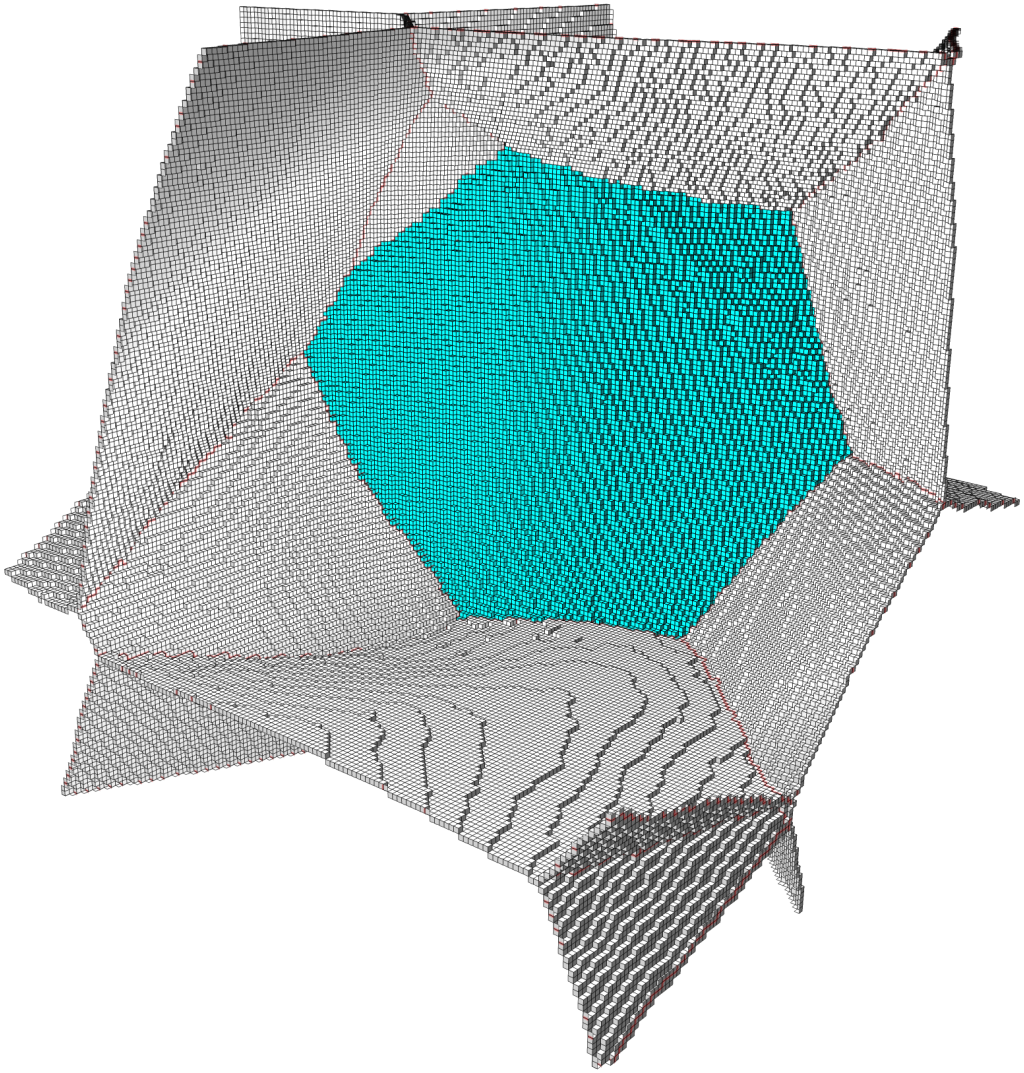


Figure 4.3: Example of surface decomposition, where one surface has been extracted, in blue.

5 SURFACE MESHING

2D TRIANGULATION OF A SURFACE DEFINED IN A 3D SPACE

This portion, although implemented, has not yet been transfered as a fully functional **Avizσ**®module.

5.1 Decomposition of the intersecting edges

The set of 1-faces that compose all surface intersections can be further decomposed into a union of simple lines. In the same manner as for the 2-faces that compose the surfaces, the proper 0-faces of the intersecting 1-faces can be classified into two types: those contained in exactly two 1-faces, and those contained in more. The 0-faces contained in more than two 1-faces are *intersections* of intersecting edges. Using these 0-faces, the set of 1-faces can be decomposed into a set of lines. In other words, one such line is defined by a non-empty connected set of intersecting 1-faces with, at its two extremities, 0-faces contained in more than two 1-faces. This decomposition is performed in order to downsample the number of points needed to define each line, keeping the extremities.

5.2 Projection onto a plane and surface meshing

Meanwhile, each surface is examined separately. A surface has the following properties: it is a connected set of 2-faces surrounded by a set of lines. For each such surface, a plane fitting algorithm is used [Zour et al., 2009]: the 0-faces that compose a surface is used to define a plane that best fits the vertices. When the walls of the bead begin to buckle during the compression, one plane will not be enough to properly represent a surface, therefore surfaces are decomposed accordingly.

For each surface or surface portion, the surrounding downsampled lines are projected onto the computed plane, and an external 2D mesher will triangulate the surface interior at the desired resolution. Initial tests have been conducted with the Computational Geometry Algorithms Library (CGAL) package, but problems for boundary conditions have forced us to consider other meshers, such as FreeFem++.

5.3 Backprojection in the 3D space

Once the projection of the surfaces or surface portions have been meshed in their respective planes, the resulting triangle sets are backprojected in the 3D space, onto the set of 2-faces. It then becomes a matter of data rearrangement to reassemble the surfaces together into a set of shell elements recognized by a 3D mesher.

CONCLUSION AND PERSPECTIVES

This work is the continuation of a study on the deformation process of polymer foam during dynamic crash loading. It is original in the sense that it aims at validating numerical simulation with the actual physical material and its deformation. A new method to perform a crash loading through interrupted shocks and non-destructive 3D imaging to monitor the strain field was proposed in previous publications. With preliminary 3D image analysis it was possible to identify foam bead interiors and quantify their mean density at each loading stage.

Because of the type of material the bead boundaries are difficult to identify, therefore further image processing was necessary. The first stage of the presented work was to obtain a truly two-dimensional representation of the foam bead boundaries, defined as an intersection of surfaces, each surface defining a separation between two beads.

Keeping the objective in mind, these boundaries are to be converted to a triangular mesh that define shell elements of the beads. With additional tetrahedral meshing of the interior volumes, a complete 3D mesh is obtained, so as to perform deformation simulation in LS-DYNA. This implied seperating each surface, and meshing them independently.

These processing stages were developed by John Chaussard at ESIEE, and are being ported into reusable versions in the **Avizo**®framework. Although the code for the surface meshing process is available it has not yet been ported.

The remaining work will thus focus on porting the plane fitting algorithm, that will define a 2D plane on which to project the surface boundaries. This boundary projection will be used for a 2D meshing, performed by a specialised software such as FreeFem++. The difficulty of this stage lies in the high deformations, where bead wall buckling impairs the correct identification of a single plane to represent a given surface. In these cases, the surface is further decomposed in portions that are independently *sufficiently* planar. Once the 2D meshing is complete, the mesh will be backprojected onto the bead boundary in the 3D space. The meshed surface splicing is then straightforward, provided the boundary points were not modified. 3D volume meshing is also performed by a specialised program. Furthermore, with the points of the complete mesh established, they should then be given specific properties depending on the material density or the bead wall thickness at the given location. The data structure should also be handled with caution, so as to make it compatible with LS-DYNA's structure. Finally, the remaining difficulty to handle in future work is the sample boundary, as these beads on the sample border have been physically cut, therefore the defined boundary should be given different properties than the intact ones inside the sample.

BIBLIOGRAPHY

Gilles Bertrand and Michel Couprie. A new 3d parallel thinning scheme based on critical kernels. In *Discrete Geometry for Computer Imagery*, pages 580–591, 2006.

Donald E. Knuth. *The Art of Computer Programming,*, volume Volume 1: Fundamental Algorithms, chapter section 2.2.6, pages –. Addison-Wesley, New York, 1997.

V.A. Kovalevsky. Finite topology as applied to image analysis. *Computer Vision, Graphics and Image Processing*, 46(2):141–161, 1989.

E. Plougonven, D. Bernard, and P. Viot. Quantitative analysis of the deformation of polypropylene foam under dynamic loading. In SPIE, editor, *Developments in X-Ray Tomography V*, volume 6318, page 631813, 2006.

C. Pudney. Distance-ordered homotopic thinning: A skeletonization algorithm for 3d digital images. *Computer Vision and Image Understanding*, 72(3):404–413, 1998.

P. Viot, D. Bernard, and E. Plougonven. Polymeric foam deformation under dynamic loading by the use of the microtomographic technique. *Journal of Materials Science*, 42(17):7202–7213, 2007.

Rita Zour, Yukiko Kenmochi, Hugues Talbot, Lilian Buzer, and Yskandar Hamam. Optimal consensus set for digital line and plane fitting. Technical report, Université Paris-Est, Laboratoir d'informatique Gaspard-Monge, 2009.