Appendix A

Notes on the Shape Library

During the course of the work of this PhD. we needed in many occasions to work with data of particles digitalised as volumetric meshes and to perform a variety of computations such as the calculations of shape descriptors and mass properties of the particles from their digitalisation. In some cases the particles digitalised as $3D$ meshes needed to be rescalled in size. In some cases it was required to smooth the meshes or to convert the file formats to make them compatible with other software such as for instance, visualisation tools. When this thesis started there was a database comprising different kinds of particles digitalised as $3D$ meshes but not an easy-to-use tool to perform the operations that we needed to do on those objects. This motivated the development of a number of programs, scripts, classes and in general, code to quickly manipulate the meshes of the digitalised particles to fulfil our requirements. Towards the end of the Phd, the programs developed were wrapped with a graphical users interface to facilitate their use. The result is a completely functional software package, that we call $SLIB-1$, which was incorporated as part of the of a long-term project of the VGW, the *Shape Library* project. The purpose of this chapter is to describe the functioning and design of the program in its current state of development.

Contents

A.1 Components of the shape library

The shape library has three components. The first one is a storage of the digitalised representations of different kinds of particles as a database. This component aims to gather a collection of shapes of different sources that can later be used for shape analysis of the particles that we can find in particular systems. The database also aims to be a first step to simulate systems where the particles have the characteristics particular to specific types of systems (eolian sands or concrete aggregates for instance). The simulation of those systems is of course out of the scope of the Shape Library but the VGW project includes external simulation tools for that purpose that could benefit from the data stored in the Shape Library.

Besides the database, we also need the numerical tools to work with the particles, calculate their properties, visualise them, etc. Additional tools to convert the digitalised versions of the particles to a suitable data structure that the external simulation tools can accept is also a requirement. The numerical tools for this purpose are the second component of the

Shape Library. This component works as a computer program that operates on the information stored in the particle shape database. The third component is a graphical front-end that enables the iterative use of the Shape Library. For the user, there is not a boundary between the underlying code of the shape library and the graphical interface but the latter is a computer program in itself, implemented completely separated from the rest of the components of the library. Such separation is desirable because in the future we could rapidly change the graphical libraries used, or the visualisation tools without requiring to change the core libraries used for calculations, the database, the data format or in general terms, the low-level part of the code.

A.1.1 The particles stored in the shape library

The shapes stored in the shape library can vary to adapt the Shape Library to different applications. Simple shapes such as as cubes, sticks, cylinders or even spheres can be stored. These simple shapes are useful when developing and debugging simulation codes since they are simple and likely to introduce a relatively small cost in the simulations for either DEM of FEMDEM. For the purposes of research, working with simple shapes such as the ellipsoidal particles modelled in chapter Chap.4 of this thesis can also provide new insights in the collective behaviour of particulate systems. It is also possible, of course, to include in the shape database digitalised representations of real particles with different characteristics such as crushed rock aggregates of different rock type, e.g., basalt, or limestone. Sand grains, or particles from other sources can be also added to the database. These are more realistic shapes and therefore attractive to include in the simulations to model real systems. Shape analysis of the characteristic particle shapes that we can find in particular systems can also be done. For instance, it is possible in principle to digitalise a number of particles of a given concrete aggregate and provide statistics of the particle characteristics that can

serve as predictors of the properties of the concrete.

At present, most of the shapes of real particles available are crushed rock aggregates from a granite quarry that supplies railway track ballast. The current available database comprises around 40 particles within the mass range of $100 - 200gr$. Figure A.1 shows some of the particles already digitalised. For the purpose of this chapter, the particles were subdivided using approximate visual judgement into several classes; Flat particles are in the group labelled F. The letter E stands for *equant* and the group comprises particles of nearly cubic symmetry. The most compact or spherical are in the group L and the most extreme in form are within the group F. The group I comprises the rest of the particles.

Figure A.1: **Example particles digitalised in the Shape Library.** The particles were initially classified in groups on the basis of visual criterion for the purposes of labelling them. Picture taken from [27].

A.1.1.1 Acquisition

Having the target particles to store, the next step to construct the shape library is to produce a digitalised representation of those particles. Simple shapes such as ellipsoids can be obtained by using computer aided design programs (CAD). Digitalisation of real particles

such as those in Fig. A.1 is done by scanning the particles. In our facilities the available technology for this purpose is a $3D$ laser scanner (LADAR), mainly suited for surface capture of objects in the size range of $\approx 10 - 100mm$ with a resolution of the order of $10 - 100\mu m$. The methodology for the acquisition in detailed in previous work of Lanaro and Tolppanen [272] and further details are given by Latham et al. [27]. Basically, the particle is placed in a rotating base while a laser beam scans its surface. The mechanical device is coupled to a computer and a software package interprets the laser scanning and produces a digitalised version of the particle directly viewable in the computer screen. Several tools to reduce the noise in the digitalisation and further processing of the shape information are also provided by the software.

The output obtained from the scanner is a file with the coordinates of a number of sampled points in the surface of the particle given in arbitrary units of length respect to an arbitrary system of coordinates. Optionally, an approximation of the surface as a set of triangles can be exported in a number of formats including *vrml* or *stl*. The *stl* files, for instance, contain a list of the triangles defined by the coordinates of their vertices. Unfortunately, there is not any explicit information about the connectivity of the surface triangles. In other words, the file does not describe a mesh. Neither the *stl* format, nor other file formats obtained after scanning are directly usable for the purposes of shape analysis with our methods. The *stl* or *vrml* files are neither directly imported by the typical visualisation software, the DEM code or the FEMDEM code. For the FEMDEM code, the particles are represented by a mesh of tetrahedra and the connectivity must be given explicitly. To software that we commonly use for visualisations [273] also requires the connectivity. For these reasons, storing the information as *stl* files is of very little interest for our purposes. We need to create a mesh.

A.1.1.2 Meshing

Several specialised meshing tools including those of common use in the group [274] can import *stl* files and generate the connectivity from the list of triangles. Most of the available software is commercial and it is likely that we wont rely on it in the near future but to date, commercial software seems to be our best alternative. These programs are wrapped in graphical user interfaces and provide extra tools for refining, coarsening and repairing defects in the original *stl* files, which commonly have many defects. As an example, Fig. A.2 shows typical post-processing on an originally hyperfine mesh created by using a low-cost mesher [274]. The program also offers the options to create triangular or volumetric meshes of $4-$node and $10-$node tetrahedra or a combination of them. As pointed before, tetrahedra meshes are used by the FEMDEM codes. Surface meshes are more cost-effective to produce visualisations and are as useful as the volumetric meshes for the purposes of shape analysis.

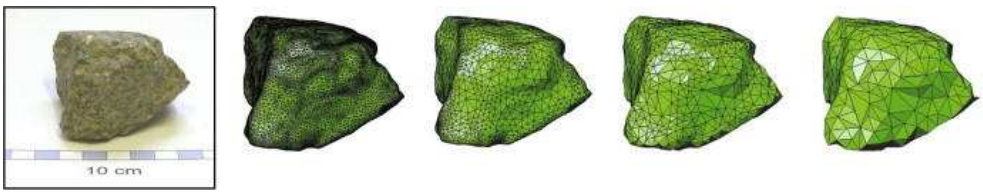

Figure A.2: **Example of a particle stored in the shape library and a sequence of coarsening steps**. The original mesh was further processed to get a coarser mesh where the most relevant features of the particle are preserved but with a more manageable number of elements and nodes. [27].

A.1.1.3 File formats

Having created the mesh, the question is what file format to use to store the information. Among the many available formats, some are copy-righted. Others are not human-readable but use a special codification. A very desirable property of the format chosen is to be human-readable because this facilitates to examine the files and spot possible errors or even

modify the files. There are many possible file formats that satisfy these simple requirements but we need to bear in mind that we will need to write code to parse the files describing the mesh in order to load them in the Shape Library and extract from them the information needed. This is not a particularly difficult step but it is unrealistic to think that we could support the processing of a large number of file formats. We wrote a few file converters and filters but at the end we decided to stick to only one, that we will refer here as the *msh* file format.

A.1.1.3.1 .msh file format

The details of the *msh* file format are found in [274]. It is a human-readable, relatively simple in its structure and can contain all the information of the connectivity's and vertices that we need. This is the standard file format used by the meshing tool that we most frequently use in the group [274]. An example of the mesh file of an scanned particle is shown in Fig. A.3.

The first line in the file specifies the dimensionality and the element type. The element type can be *Triangle* or *Tetrahedra* and in the latter case the elements can be either linear, 4 nodes per element, or non-linear, 10 nodes per element. The keyword *Coordinates* precedes the list of the indexes and vertices of the mesh. After the coordinates there is one or more sections for the elements (connectivity). In Fig. A.3 there are 2 sections for the elements. The first one lists the tetrahedra in terms of the indexes of the vertices of each element. In this particular example the file also contains a list of the triangular elements that comprise the surface of the object. Namely, the triangular surface of the object is given explicitly. This part is not present in the typical case but clearly depends on how the meshing was done and the information exported.

For the purpose of the Shape Library to be as generic as possible, the reading engine of the program is capable of parsing *msh* files with any combination of tetrahedra and triangles

```
MESH    dimension 3 ElemType Tetrahedra  Nnode 4
Coordinates
1    0.0571352 -0.0309456  0.768534
2    0.0482777  0.224479   0.833099
        (....)
290  1.52337 -0.237696 0.0336184
291  1.56588 -0.0752636 0.0874164
end coordinates

Elements
0  1 0 47
1  2 1 46
(...)
578  284 291 288
579  291 283 290
end elements

MESH    dimension 3 ElemType Triangle  Nnode 3
Coordinates end coordinates

Elements
12642       7       1       3
12643       1       7       5
(...)
14197    56    255    154
end elements
```

Figure A.3: **File format for .msh files**

and with any combination of volumetric and surface meshes given in any order in the input files as long as the keywords; *coordinates, elements,etc.* is respected.

A.1.1.4 Particle shape database

A database is formally a complex program comprising a search engine a special way of storing information in forms or tables and a querying program, the later typically written in languages such SQL or similar. A database is in itself a complete platform. This is an efficient way of storing an querying information. However, setting and maintaining a

database requires expertise. In an environment where the main focus is research in earth science, it did not seem feasible that the users were willing to spend time on maintaining and using databases so we decided that for us, in regards to the Shape Library, the shape database is simply be a collection of input files placed together in a folder. The files in a database can be in any of the three available file formats. These are the $.msh$ format described before and another two formats that we will introduce later.

A.1.2 Particle shape descriptors

Following from Sec. 2.3, the description of the particle shape can be done in at least three length scales. At the finnest length scale, the shape is described by the texture. This comprises the asphericities in the particle surface at the scale of the interparticle contact areas. The description of the texture is a very complex task and in practice, this length scale is typically not captured in the digitalised shapes. Even if a hyperfine mesh could be produced to capture this level of detail, the texture could not be explicitly included in the simulations. Instead, in the simulations the texture is modelled via friction coefficients as we did in our previous chapter Chap. 4.

At a coarser scale, the angularity is considered an important quantity for shape analysis although it is vaguely defined and rarely quantified in $3D$. For the simulations in this thesis this parameter is really irrelevant because the particles modelled are always rounded. Other tools of the VGW, such as the FEMDEM code could in principle help to correlate emergent properties of the systems to the angularity of the particles. For this purpose the Shape Library intends to provide descriptors for the angularity in the future.

At the current state of development, the shape of the particles is described in the coarse scale by form indexes such as the elongation ratio α or the sphericity ψ. These indexes also serve for the purpose of shape analysis and classification and can act as predictors of the

emergent properties of particulate systems. In the previous chapter Chap. 5, for instance, α and ψ were correlated to the porosity of simulated packs of particles. Other quantities such as the the principal inertia moments are also needed for the solution of the Euler equations in the simulations. These also serve for shape comparison as we did in chapter Chap. 3. The shape library computes these quantities and some others from the digitalised representations of the particles as a mesh. The computed descriptors are the following:

- Inertia moments

  As pointed in chapter Chap. 3, the Mirtich formulae [222] are used to obtain the inertia tensor of the particles from their mesh representation. Then we solve the corresponding eigenvalue equation to compute the principal axes of rotation and the inertia moments. As an example, Fig. A.4 shows the three principal axes of rotation computed for a digitalised particle.

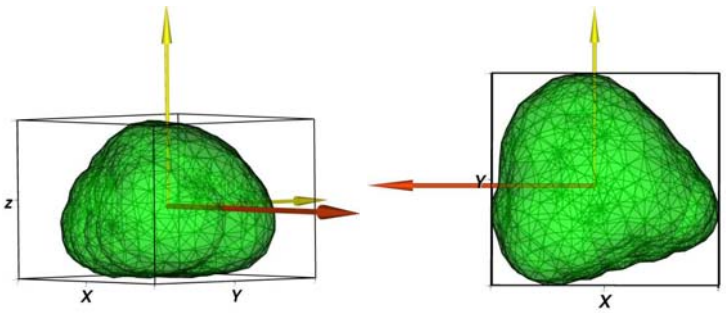

Figure A.4: **Inertial axis of an example particle as computed using the Shape Library.** The principal axes of inertia of each particle are calculated. These axes define the three orthogonal directions along which the lengths of the particle are computed. In the figure, the darker vector signals the direction corresponding to the smallest inertia moment.

- Aspect ratio α and particle Lengths L, I, S

  The problem of computing the aspect ratio of a particle can be stated as follows: For a shape such as the one in Fig. A.4 or Fig. A.2, give a meassure of the relation of the particle lengths $S < I < L$ in three orthogonal directions. Obviously, the

lengths of the particle depend on the orientation respect to the particle of the three axes chosen and there is not a standard definition of those axes. For the Shape Library we took the proposal of Wang et al.[99] of choosing these axes coinciding with the principal rotation axis of the particle. The advantage of this choice is that these axis are unambiguously defined.

Having defined the orthogonal axis, the lengths of the particle are defined as the lengths of the equivalent ellipsoid of the particle along those axes. The equivalent ellipsoid is defined as in previous chapter Chap. 4, as the ellipsoid that coincides with the particle in the orientation of the principal axes of rotation and the inertia moments against those axes. Being $I_1 > I_2 > I_3$ the inertia moments of the particle the three semi-axes a, b, c of the ellipsoid are given by:

$$a = \sqrt{\frac{5}{m}(I_2 + I_3 - I_1)} \tag{A.1}$$

$$b = \sqrt{\frac{5}{m}(I_1 + I_3 - I_2)} \tag{A.2}$$

$$c = \sqrt{\frac{5}{m}(I_1 + I_2 - I_3)}. \tag{A.3}$$

At this point we should note that the mass m of the particle is not known from its mesh. Hence, for our calculations we assume a unit volumetric mass density so m is equals in magnitude to the particle volume V. It is possible, of course, to assume a different volumetric mass density and get a better approximation of the mass. In the shape library the three equivalent ellipsoid lengths are computed for each particle of the database and the aspect ratio is computed as $\alpha = L/S$, where S is the smallest semiaxis among a, b and c and L is the greatest one. We used this definition of the aspect ratio in chapter Chap. 4.

- Gyration Ratio R_g

The gyration ratio R_g is computed as the greatest of the distances from the centre of mass of the particle to the surface vertices of the mesh. This quantity is useful for the Discrete Element code used here to quickly discard the potential contact among particles because R_g defines the radius of smallest sphere that is centred in the particle and covers all the particle surface. Two particles cannot be in contact if their enclosing spheres are not touching.

- Volume V and surface S

  The volume or the surface of a particles are not useful shape descriptors by themselves because two particles can have the same shape and different volumes. Resizing a particle changes its surface and volume but not its shape. However, non-dimensional combinations of S and V define shape descriptors such as the sphericity or compactness. The volume V is obtained directly from the Mirtich formulae regardless of the type of the mesh. Namely, we do not need a volumetric mesh to compute V. However, we need a surface mesh to compute the particle surface. If such surface mesh is given then the area is computed as:

$$S = \frac{1}{2}\sum_{i}^{N} |(v_3^i - v_1^i) \times (v_2^i - v_1^i)|^2, \tag{A.4}$$

  where the sum is over the N triangles on the mesh, v_k^i is kth the vertex of the triangle i. It is likely, however, that the surface mesh is not always available. This will depend on the kind of post-processing done after scanning the particle. Since we need the surface of the triangular mesh to obtain the sphericity and other descriptors we implemented an algorithm to extract the surface from the volumetric mesh of tetrahedra (either linear or non-linear tetrahedra). The program then calculates the surface mesh automatically if it is not given in the input file.

- Sphericity ψ and equivalent volume sphere radius R_v

  The sphericity $\psi = (36\pi V^2)^{1/3}/S$ is by far the most used Form descriptor in the literature. The equivalent volume sphere radius R_v, is simply defined as the radius of the sphere having the same volume of the particle. These two quantities can be computed from the previous calculations.

It must be noted that other form descriptors such as the oblate-prolate index or compactness are not directly computed by the shape library. The case is that there in literally many shape descriptors but besides the sphericity, all of the other form indexes are computed as relations among L, I and S, so in the event that a particular index other than those computed here were needed, it could be obtained from our previous calculations.

Table A.1: **Shape descriptors for real shapes computed by the Shape Library as compared with experimental measurements.** The volume of an elongated (L) and an equant (E) particles is measured experimentally and calculated using the Shape Library. Other shape descriptors are also shown.

Particle	Mass		Moments			Volume	Area	Size
	Exp. (g)	Calc. (g)	I_1 (gcm^2)	I_2 (gcm^2)	I_3 (gcm^2)	V (cm^3)	S (cm^2)	$2R_v$ (cm)
Elongate (L)	142.39	144.79	941.8	891.9	162.3	53.6	93.50	4.68
Equant (E)	172.35	174.15	587	447.4	421.3	64.5	90.80	4.97

An an example, table A.1 shows the calculations done for the two granitic particles in Fig. 3.1 scanned at high resolution. The particles were weighted to determine their real mass. A numerical approximation of the mass was also obtained as the product of the computed volume and an assumed volumetric mass density of $\rho = 2.65gr/cm^3$. The comparison checks for the geometric accuracy of the scanning process and also the calculation of the volume from the Mirtich formulae. The results are more than satisfactory and within the margin of error for the granite density. Several other tests for the calculations of form

descriptors were done by comparing the theoretical and computed values of the volumes and inertia moments of meshes of spheres, ellipsoids and cylinders of known sizes.

A.1.3 Use of the shape libray

The Shape library is an iterative application wrapped in a Graphical User Interface. When the application is executed, a Start-up screen such as Fig. A.5 shows indicating that the program has been loaded successfully and the user can open a database. At the program start-up, the user can either display an informative window showing the version details of the application by clicking the *About* menu or loading a new database by selecting *Open Folder* from the *File* menu in the *Tool Bar* (see Fig. A.5).

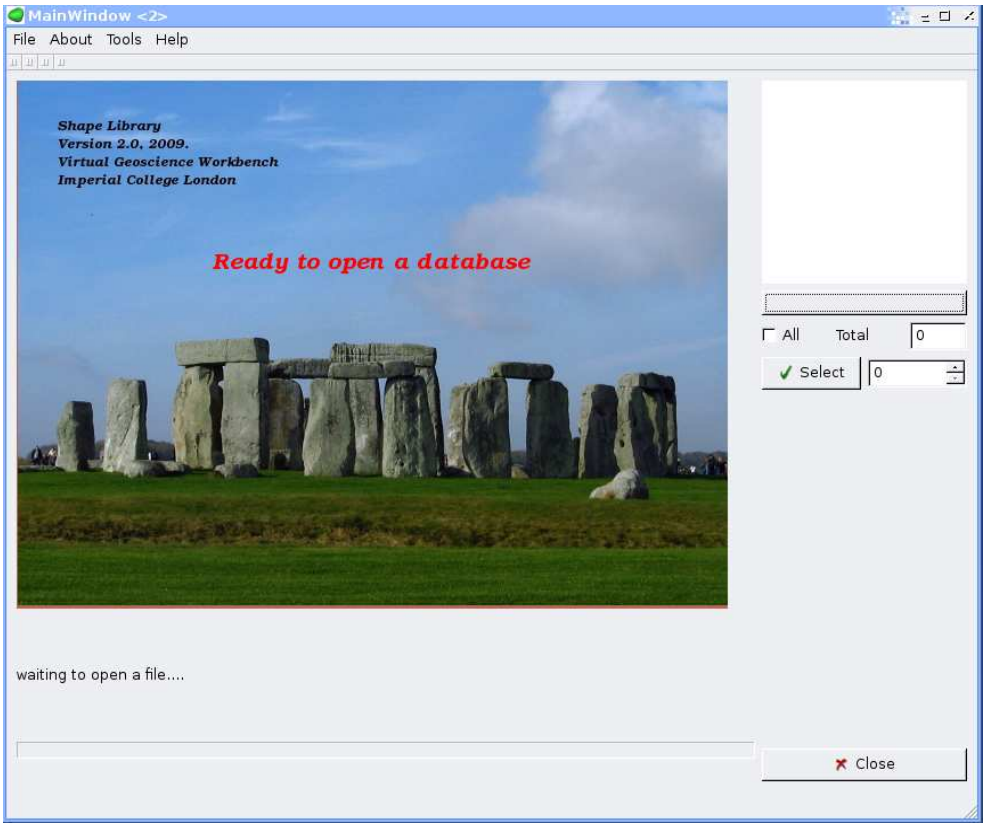

Figure A.5: **Start-up Window of the Shape Library.** When the window is displayed the program was loaded successfully and the used can open a database.

A.1.3.1 Loading and displaying particle shape databases

When the choice is made to open a new database (or folder) a standard file selector window shows up and the user is entitled to select the folder where the meshes are stored. After the folder selection the program scans the folder and makes a list of all the files. In a second step the first 500 lines of each file are read in turn. If the file is not recognised as a digitalised particle then it is not processed and at the end of the loading process a notification is displayed to the user notifying that some files were ignored. Depending on the computer speed and the number of files to process, the loading might take a few minutes. In linux, about 1000 heavy meshes of a few thousands of nodes are read in a few seconds. In Windows this might take about 20 times more. The progress bar shows the percentage of the files that have been processed. The status message (See Fig. A.6) shows the name of the file being currently processed. This is useful to identify whether a problematic file takes longer than usual to be processed, probably because unexpected problems with the data.

After all the information is loaded, the Main Window shows a table with the description of the particles. If a new folder is open, the information previously loaded is lost. The option *Add Folder* in the *File* menu enables to load a new database and add it to a pre-exiting one without destroying previous contents in the table.

A.1.3.2 Visualisation of shapes

At the top right of the main Window is shown a snapshot of the shape selected in the table. By default, the figure is permanently rotating (animated). Double clicking on the animation widget stops or restarts the rotations. An enlargement of the visualisation is possible by clicking the button below the picture (see Fig. A.6). Note that the little animation can be completely eliminated by enlarging the image (button below the picture) and then closing

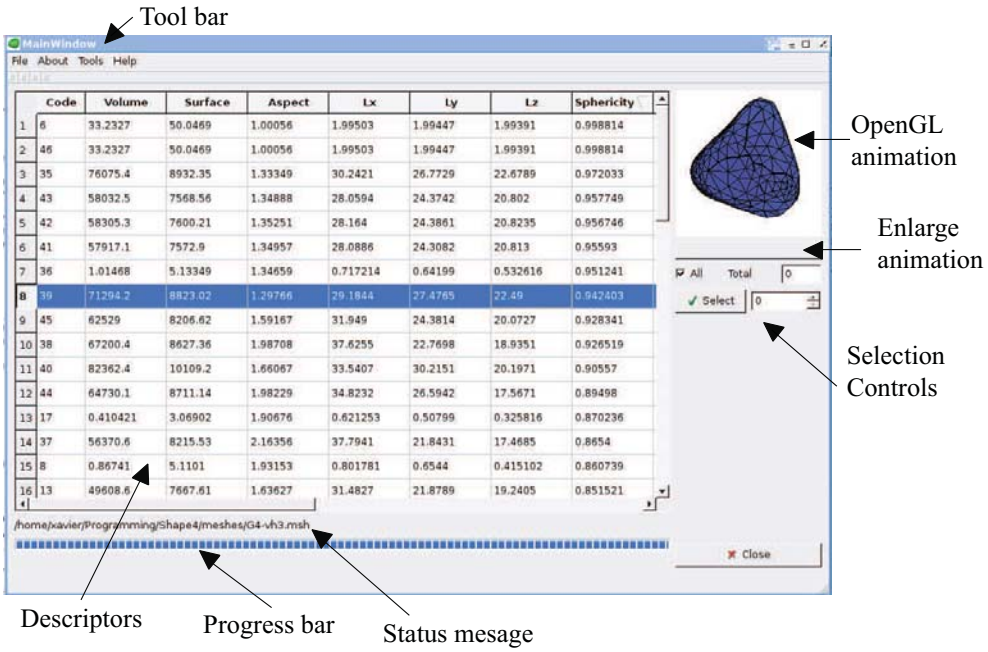

Figure A.6: **Example of Shape Library Main Window displaying a database.** The shape descriptors such as the aspect ratio, sphericity, mas, volume, lengths among others are calculated on the fly for each shape. A small animation of the particles is also shown. Options for resizing the particles and converting formats are available from the *Tools* menu.

the window. If the button *Close* is pressed instead, the small animation is restored at the top right of the Main Window. In either case, the small visualisation is provided only for the purpose of having a quick look of the particle while the user moves through the records in the table.

A.1.3.3 Selecting particles

The library is designed as an interactive tool where the user is allowed to mark populations of particles with different characteristics. The marking is done while the user pre-visualises the particles and their characteristics in the graphical user interface to aid selection. The particle marking has a double purpose. One, is that the program allows several operations on the data of each particle such as resizing, file format conversion or saving and all these operation are performed *only* on the particles selected. For instance, the user can select

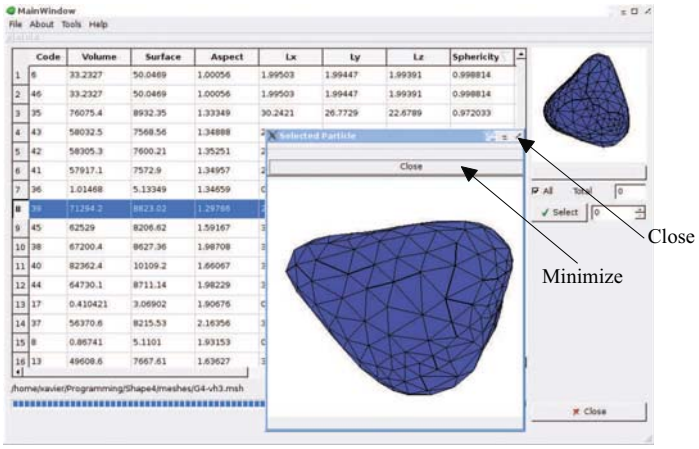

Figure A.7: **Visualisation of a selected particle**. The interface shows an enlarged OpenGL visualisation of the particle selected in the table.

only a few particles, say the rounded ones, and then simply save a new database with only the selected particles. As an example, a typical situation is that the length units of each new particle added to the database are millimetres since this is the default unit length of the software used during particle scanning. In a database where a new particle is added, probably the length units used for rest of the particles (for lengths, volumes and surfaces) are measured in centimetres or meters because this is more common for the numerical simulations. The selection enables to only mark the new particles, rescale them and then save the database again with al the lengths in the same units. A second objective of marking the particles is to create a population of particles with a given set of characteristics as a preamble to produce an output to the solids simulation codes.

The particle selection is done in the middle right part of the Window shown in Fig. A.6 marked before as *Selection controls*. Figure A.8 shows a closeup of the controls. The particles are marked individually or all at once by simply unmarking/marking the box *All* and then clicking the selection button. Each shape can be selected more than once, which is useful to produce shape and size distributions as described below.

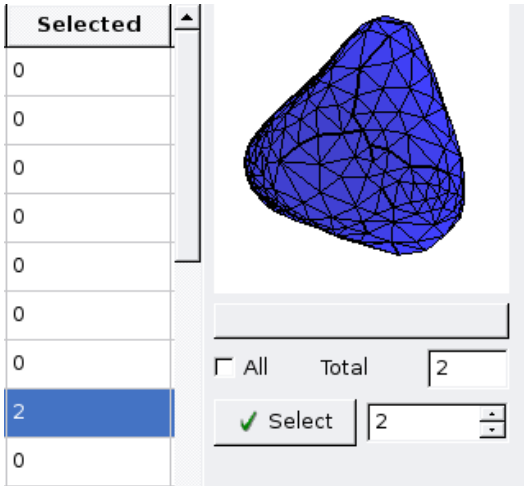

Figure A.8: **Controls provided for the selection of particles.** The particles can be selected individually or all al once.

A.1.3.4 Resizing

The shape library provides a few tools to re-size the particles to a given volume or gyration radius. Populations of particles with a given volume distribution can also be created. The uniform resizing to a given volume/radius is useful for the purpose of shape comparison free of size effects. For instance, comparing the surface of two particles of different volume is of little help but if both particles have the same volume then the surface comparison is an indicative of differences in the irregularity of the particles. The possibility of creating a size distribution aims to produce output files for the solids simulation codes. The size distribution itself is read from a file provided by the user in the window shown in Fig. A.9. The file is assumed to have two columns and codified in ASCII. The first column is the particle volume and the second one is the probability of occurrence of that volume. The code takes the particles selected in the table and assigns to them relative occurrence weights on the basis on the number of particles selected per shape. For instance, if we had only 2 shapes s_1, s_2 and the first one is selected 3 times while the other one is selected only once, then a population with be created with a given volume distribution in which

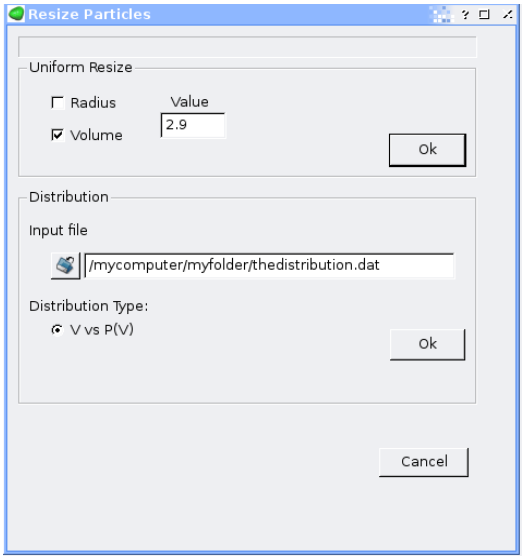

Figure A.9: **Controls to create population of particles with a given size distribution.** A group of particles with a given size distribution can be generated. The distribution is provided externally in a text file. The particles can also be all uniformly resized to a given volume or gyration radius.

$\approx 75\%$ of the particles will have the shape s_1. This has the purpose of creating a population with a given volume distribution but also with a preferential occurrence of some selected shapes. Once a population is created, the code generates a new table corresponding to the newly created group of particles. We used this feature in chapter Chap. 5 to create the size distribution of the grains in the models of sand with some samples having preferentially elongated particles.

A.1.3.5 Exporting the particles

The information in the shape library can be readily converted to different output formats suited for different purposes. Whether these purposes are the storing of the information in different formats or producing an output file for other programs, the Shape library offers several possibilities. One of them is simply saving a modified database in the *msh* format. This has the purpose to store a work session and easily recovering it later if needed. Other options to export the data are also provided:

A.1.3.5.1 Graphics format .vtk

Obviously the visualisation capabilities embedded in the shape library are limited. These are only intended for the sake of providing a quick look of the particles while selecting them and a first tool to compare them visually. A more professional visualisation tool might be required sometimes. For this purpose, all the selected particles in a database can be exported to individual files in vtk format. This is a graphic file format readable by the common visualisation tools used in the department [273]. Fluidity (the CFD solver used in Chap. 6), for instance, exports the data in *vtk* (or compressed *vtu*) formats. The DEM code produces animations and snapshots in *vtk* files (among other other options). The FEMDEM also exports vtk files. The file format is fully described in [275, 276]. Several different data-structures can be described using the vtk syntaxis. For the shape library, we stick to the *Polygonal Mesh* structure that can be fully loaded into the visualisation tools (see documentation for the file structure in [275, 276]).

A.1.3.5.2 Text file format

The program offers the possibility to export the whole table of shape descriptors in a text file loadable by most of the spreadsheet programs available in the marked (Excel for windows, OpenOffice for Linux). Note that since the lengths, surface and volume of each particle are exported, other Form descriptors can be externally computed.

A.1.3.5.3 Modified .msh

So far, we have described the tools of the shape Library making no reference to the particles represented by clustered spheres that we used throughout the thesis. The Shape Library, however, can also load files describing the clustered sphere particles and all the aforementioned operations described in this chapter (resizing, creating populations, calculating descriptors,etc.) can also be performed on the clustered-sphere particles. Two

conditions are required. The first one is that the surface of the sphere cluster needs to be meshed. This requirement is to facilitate and speed up the calculation of the shape descriptors of the clusters and for the purposes of visualisation. The second requirement is that the position and radii of the spheres in each cluster needs to be given in the input files. Figure A.10 shows a modified *msh* file where we added a header between the tags <ShapeRecord> ... < /ShapeRecord> indicating several properties of the particle. In the case shown the part describing the cluster is enclosed in four lines between the tags <Balls> ... < /Balls> . The first line gives the number of spheres, 4 in this case. Each of the other four lines is read as the x, y, z coordinates of each sphere and its radius R. The presence of the header instructs the program to process this file in a slightly different manner to account for the fact that it describes a mesh and also a cluster of spheres. After the header, the file is identical to the standard *msh* file. By default, the clusters are displayed in the Shape Library with light colours contrary to the meshes that are shown in dark colours.

The presence of the header is not exclusive of clustered-spheres particles. This header is added by the Shape Library when a file is saved and its presence does not prevent the meshing tool used here [274] from reading the file. This header is added for convenience because when a mesh is read, its shape descriptors are computed on the fly if these are not available. When saving this information in the file, the Shape Library does not recalculate the shape descriptors again if the file is open in another work session. This saves some time when processing hundreds of particles.

A.1.3.5.4 Exporting data to simulation codes

Among the initial goals of the shape library, we considered to include the capabilities to export input files to the simulation codes. However, a simulation code such as DEM or FEMDEM needs as input several parameters. Some of these are related to the integration of the equations such as the time step or the boundary conditions. Other parameters are

```
<ShapeRecord>
<meshFile>  G4xx.msh </meshFile>
<Code> 1 </Code>
<Count> 1 </Count>
  <MassProperties>
   <Volume> 71294.2  </Volume>
   <Surface> 8823.01 </Surface>
   <Center> 0 0 0    </Center>
   <Axis> 1 0 0  0 1 0  0 0 1 </Axis>
   <Inertia> 1.7977e+02 1.93568e+02 2.29095e+02 </Inertia>
  </MassProperties>

  <GeometricProperties>
    <sRadi> 25.723   <gRadi>
    <gRadi> 34.1915  <sRadi>
    <Lengths> 29.1844  27.4765    22.49 </Lengths>
    <Sphericity> 0.942403 </Sphericity>
    <Compactness> 5.13154 </Compactness>
    <Aspect> 1.29766 </Aspect>
   </GeometricProperties>
  <Balls> 3
    0.1 0.1 0.1   1.12
    0.1 0 0.7     1.32
    0.34 045 0.2  1.22
  </Balls>
</ShapeRecord>

(...here ends the header and starts the msh file )
MESH    dimension 3 ElemType Tetrahedra  Nnode 4
Coordinates
1  30.5831 -8.78917 -11.7216
2  28.7591 -5.01107 -17.8006
3  23.9732 -5.34357 -12.2046
4  25.3246 -11.2291 -16.5076
   (...)
```

Figure A.10: **File format for modified** *.msh* **files.** The format is identical to the *msh* format but stores a header with the properties of the particle. If this header is present then the shape descriptors of the particles is not computed when loading a database.

related to the elastic properties of the particles and several more are instructions to the code set the frequency for producing dump files, etc. Furthermore, given the versatility of such

techniques, several different simulations can be done and the input files can vary depending on the problem studied. None of the aforementioned parameters relates to the shape of the particles. To set up these input files in a graphic program, we need a separate graphical user interface (GUI) dedicated to each of the simulators. The Shape Library is a package in itself, it has its own GUI and this author considers that it is not the attribution of the Shape Library to become the GUI of other tools. Notwithstanding, part of the utility of the Shape Library is to produce a basic input for the other simulators containing all the information in relation to the meshes of the particles, their geometric characteristics, size, number, etc. Such information must be organised in an easy way so that an interface can be written to communicate the library with the other tools part of the VGW. These export capabilities have been implemented in the actual version of the Shape Library. For the DEM code the output is a collection of files, each of them of the type of the modified *msh* format mentioned before. The code then handles the files and organise the data according with its own structure. For FEMDEM, the output is a standard *msh* format in which a single mesh is described. Such a mesh is the addition of all the particles selected. Node renumbering and several other needed details are handled by the Shape Library. The program also distributes the particles equi-spaced in a volume to avoid overlaps. Such output is then processed by the FEMDEM code.

A.2 Applications

The applications of the Shape Library have been already mentioned in different parts of this chapter while giving references to the sections of this thesis where the tools of the Shape Library were used. However, the program can be used also for the purposes of shape analysis of populations of particles. For instance, the surface area to volume relation as size increases is considered an important indicator of shape for $3D$ objects [277, 141].

In laboratory tests, however, surface areas of the particles are difficult to obtain. For our digitalised shapes it is straightforward and enables the calculation of other parameters not easily obtainable experimentally. The sphericity is one of those parameters. The aspect ratio, apparently simple to measure in the laboratory, requires manual measurements using callipers. Following Maertz [147], the process is slow and laborious, and often degenerates into crude approximations. The tools presented here for digitalised particles remove the subjectivity in the choice for the orthogonal axes. The process is fast and the measurements are fully reproducible.

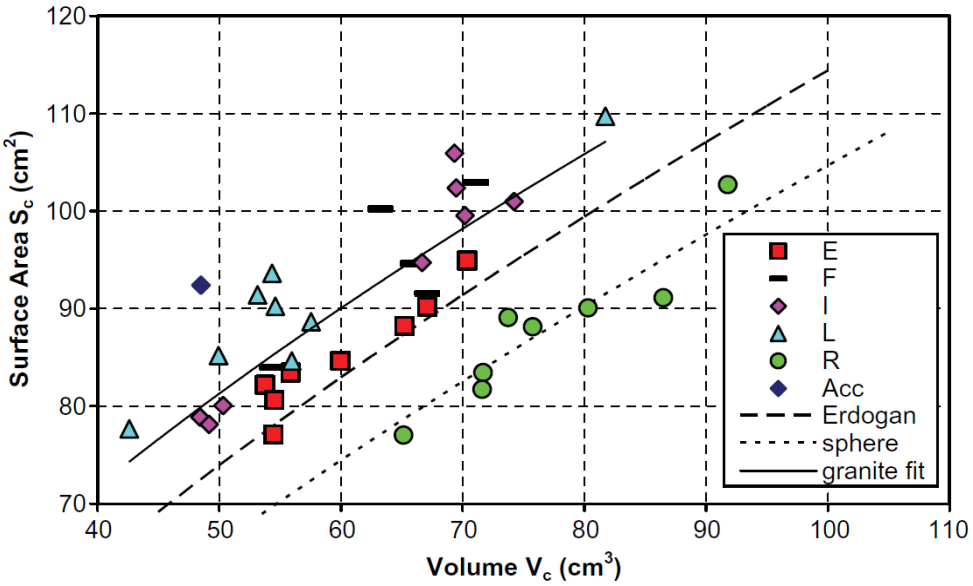

Figure A.11: **Scatter plot S, V and power law fit $S = AV^K$ computed for the groups of particles in Fig. A.1.** The figure shows the analytical trend for spheres $K = 0.667$. For the most angular particles the best fit is obtained with $K = 0.57$. The data for the particles of cubic symmetry (E) coincides with the obtained by other authors for different aggregates ($K \approx 0.6$)[277]. From the figure, it is obvious that although the particles studied come from the same source, there is not an obvious correlation between shape-size.

As an example of the possible uses, Fig. A.11 shows an scattered surface S to volume V plot for the 40 particles shown in Fig. A.1 plus a few particles comprising beach pebbles classified as round (R). The data for each group of particles was fit by a power law $S = AV^K$. The analytical solution for spheres $K = 2/3$, is reasonable close to the quartzite

pebble (R) data. For the group (E), visually classified as those of cubic symmetry, our results compare well with those of Erdogan et al. [277] for a granitic aggregate. The best fit for the most angular particles in our database is obtained with $K = 0.57$. An interesting observation is that despite all our 40 particles in the groups F, E, L, I are from the same origin and distributed in a relative narrow size range, there is not a unique relation shape-size. Indeed, we can differentiate subgroups within the total population, for which the $S(V)$ trends are different. This has been traditionally a source of debate in the literature.

A.3 Acknowledgements

All the particles in the shape library were scanned and made available by Dr. John Paul Latham. The work of the other coauthors in the journal paper [27] is also recognised here.