Parameterised mesh generation for finite element analysis of tunnelling

Manuscript submitted to Computers and Geotechnics

13/11/2008

D.K. Koungelis
Donaldson Associates Ltd
The Pentagon Centre
36 Washington Street
Glasgow, G3 8AZ, UK

C.E. Augarde*
School of Engineering
Durham University
South Road
Durham DH1 3LE, UK
Tel: (+44 or 0) 191 334 2504
Fax (+44 or 0) 191 334 2408
charles.augarde@dur.ac.uk

*Corresponding author

Abstract

The use of three-dimensional (3D) finite element (FE) models for geotechnics is increasing, particularly for the simulation of the progressive excavation and lining of a tunnel (or tunnels), including features such as underground services, or surface structures and loading. A major overhead in the use of 3D FE models is meshing and this is particularly so for tunnelling schemes due to the geometries involved. This paper introduces a parameterised scheme for automated 3D mesh generation of highly complex tunnelling layouts using a freeware mesh generator. Comparisons are made with meshers found in commercial FE software.

Keywords: mesh generation, tunnelling, GMSH, finite elements

Introduction

Tunnelling is a complex construction operation to undertake and considerable effort is required to predict the effects of tunnel construction, particularly in urban areas where interaction with other structures must be checked for damage prediction and to ensure stability throughout [1-3]. Prior to a tunnelling scheme, geotechnical surveys are carried out to learn more about the ground properties and conditions in which the tunnel is to be constructed in order to decide which excavation method will be used. Of prime concern in modern tunnelling in soft ground are the surface settlements and tunnel lining deformations that arise. Despite their small magnitudes their differential nature makes them potentially destructive to brittle surface structures, such as masonry buildings.

Settlement prediction is usually based on a semi-empirical approach as described in many references (e.g. [1]) but all are however incapable of accounting for the presence of surface structures and any three-dimensional (3D) effects. For this reason, engineers have turned to 3D finite element (FE) analysis to make predictions for increasingly complex schemes involving multiple tunnels and where the soil behaviour is simulated with elasto-plastic constitutive models. Until recently 3D modelling was the preserve of academic research, but it is

increasingly being used by industry (e.g. [4]), while much research continues to improve modelling for this problem: some recent examples can be found in references [5-7]

To arrive at predictions of the effects of tunnelling using a FE model can be seen as a three-stage procedure: pre-processing, analysis and post-processing. Many papers report the difficulties associated with 3D FE analysis for tunnelling however the majority concentrate on aspects relating to the analysis stage, such as the choice of material properties and constitutive model. Little has been published to help those attempting to make the pre-processing stage more efficient.

Pre-processing involves the generation of a suitable FE mesh, and imposition of boundary conditions. Preparing, and checking, a mesh for a complex tunnelling simulation can be extremely time-consuming. This paper introduces a parameterised scheme for automated, more efficient and robust 3D mesh generation of highly complex tunnelling layouts using a freeware mesh generator (Gmsh). Comparisons are made with the facilities available in mesh generators found in commercial FE software.

Volume division into elements

Mesh generation is the art of dividing a volume into elements, a subject with a surprisingly long history. Plato and Aristotle were the first who dealt with this issue, almost 2,300 years ago [8] although obviously not for the purposes of FE analysis! A wide variety of techniques have since been developed to fill a 3D volume with finite elements and the research literature on this topic is now vast. References [9,10] summarise much of the recent development.

Mesh generation can be broadly divided into structured (uniform and non-uniform) and unstructured approaches. The former are based on generation from two or more groups of parallel lines which intersect. If both groups have parallel lines which intersect then a uniform mesh is produced. If only one group has parallel lines which intersect with the other non-parallel group then a non-uniform structured mesh results. Structured meshes are straightforward to generate but can be highly inefficient for complex geometries. Unstructured techniques are more widely used and are usually based on one of the following methods: the advancing front method, the paving method or the 3D version of the Delaunay triangulation method. Many geotechnical analyses are carried out with the assumption of near-incompressibility (to model *undrained* conditions). In this case it has been shown that tetrahedra outperform hexahedra particularly for linear elements [11]. Tetrahedra are also much better for modelling curved boundaries (e.g. a tunnel outline).

Mesh generators in commercial FE software

The three stages of pre-processing, analysis and post-processing are usually incorporated into a single package in commercial software. Interestingly, the geotechnical software Plaxis installs as a single package but each stage appears as a separate program (*Input, Calculation* and *Output*). In contrast many geotechnical researchers choose to use separate software for each stage. This allows them flexibility to adapt the part in which they are most interested (usually analysis). This approach tends to be error prone due to data transfer between the stages. Some commercial packages are restricted to the creation of structured meshes and for tunnelling problems parts of the meshing have to be undertaken by hand.

As an example of the shortcomings of using a structured mesh generator a simple single horizontal axis tunnel problem will be meshed. The dimensions of the domain are presented in Fig. 1. This problem is symmetrical around the vertical tunnel axis thus, only half needs to be modelled. To start, a slice of the mesh including the tunnel is created (Fig. 2a). The elements used for this analysis are tetrahedra (for the reasons described above), from decomposed hexahedra. This slice is then extruded in the direction of the *y*-axis (Fig. 2b). Producing even this simple mesh can be time-consuming. For a series of analyses in a parametric study where, say, the tunnel depth (z_0) or diameter (*D*) is varied, this must be repeated almost from scratch,

each time. Some time can be saved if the first mesh is carefully part-generated as shown in Fig. 3a and re-meshing for different analyses then involves the sub-section around the tunnel, although care is needed to ensure coincidence of nodes. This problem is relatively straightforward due to the simplicity of the geometry but things become more complicated when the axis is inclined (Fig. 3b) or even curved (i.e. no symmetry). The procedure is more or less similar to the one described above. This time however, the first block of elements (the elements which form the tunnel in particular) cannot be extruded towards the *y*-axis. This is because the longitudinal axis is now inclined. The whole volume of the inclined tunnel has therefore to be re-designed element-by-element. This procedure is even more time consuming although the coarser elements close to the boundaries can be extruded as before (Fig. 3a).

The two procedures described above refer to a single tunnel analysis. For more complicated geometries, e.g. twin tunnel construction, meshing time tends to be prohibitive for any user if an extended parametric study is required.

Parameterized mesh generation

Motivated by the need to generate a large number of meshes for a parametric study of both single and twin-tunnelling [12], a scheme based on parameterizing analyses has been developed. The mesh generation software used is the versatile freeware package Gmsh, developed by Geuzaine and Remacle [13]. Analysis and visualization in this project used a commercial FE package (Strand7) although this paper focuses on the pre-processing. It has to be stressed here that most of the 3D generated meshes refer to the case of already driven tunnels with their permanent lining installed. Thus finite elements are not generated within the volume of the tunnels. The purpose is to examine the issues associated with pre-processing, not a full analysis.

The parameters that are used to describe a range of tunnelling geometries are shown in Figures 4 and 5, where D is the tunnel diameter, $x1$ and $z1$ are the horizontal and vertical shifts of the tunnel axis respectively, and $k1$ is the tunnel inclination along the *z*-axis. The dimensions of the problem are identical to those described for the commercial structured mesh generator in the previous section (Fig. 1).

Gmsh is a fully automated 3D unstructured tetrahedral mesh generator in which the Delaunay triangulation algorithm is implemented in 3D. In 2D the algorithm produces a mesh of triangles from a distribution of nodes such that no other nodes lie within the circumcircle of each triangular element. In 3D the equivalent algorithm can be expressed as follows: no points exist within the circumsphere of a simplex (i.e. a tetrahedron) other than those which form it.

Gmsh works either via a GUI or from an ASCII file containing commands. An attractive feature is the possibility of variable substitution in this input file, and it is this which makes parameterized meshing possible here. For instance, the outer dimensions of the mesh can be defined as X, Y, Z and their values defined at a single point prior to their first use in the file. This allows easy generation of different meshes by variation of the definitions, a major improvement from the approach using structured meshes described above, and a positive advantage for parametric studies. Mesh refinement is achieved via the specification of a local adaptive parameter “characteristic length” (ChL) which can be applied to selected points in the model. In this way, the user can refine some areas within the mesh to achieve greater accuracy in the areas of interest. Such areas can be between materials of different properties (i.e. soil-structure). However, engineering judgement and experience is required in order to decide on the level of detail within the domain.

Mesh quality

In order to measure and validate the quality of a generated mesh various shape measures are used. The most common found in the literature are presented in [14-17]. For convenience and compatibility purposes the range of these shape measures is (0, 1) with unity indicating an ideal equilateral element in 2D. Gmsh uses the aspect ratio γ which is defined as:

$$\gamma = \frac{12}{\sqrt{6}} \frac{[3V/(\sum FA)]}{L} \quad (1)$$

where V is the volume of the element, L is the maximum edge length and $\sum FA$ is the sum of the face areas.

Code for parameterized mesh generation

In this section mesh generation for the simplest tunnelling scheme (a single tunnel with a horizontal axis) will be described. In this way the difference in philosophy to the structured approach described above will become evident. The user writes a simple set of commands in a text file (see Appendix A1) as which is then read by Gmsh. The main input operations executed by Gmsh in order to generate a FE mesh are presented with a flowchart in Fig. 6. Each operation is described in detail below.

The geometry of the problem should be described from the "bottom-up". At the beginning of the file, several parameters which specify the geometry of the domain (see Figs 4 and 5) and the size of the finite elements are defined with the use of numeric values or by employing other previously initialized parameters as follows:

```
D = 4;
X = 8D;
Y = 5D;
Z = 5D;
ChL1 = D;
```

Where X, Y and Z are the dimensions of the domain in space while $ChL1$ is the general characteristic length used to generate elements in the mesh. In this case the domain consists of elements of roughly the same size. In the case where the domain consists of elements of different sizes then a range of values $ChLn$ should be initialised (where n refers to the number of the different ChL values). After the initialization of the above parameters, points, lines surfaces and volumes have to be defined. These are called "*elementary entities*" in Gmsh. An identification number is assigned to each elementary entity within parentheses. Points at the boundaries of the domain are defined as follows

```
Point (1) = {x, y, z, ChL1};
```

where the first three parameters within the braces refer to the coordinates of the particular point in the 3D space (either numeric values or with respect to the global coordinates x, y and z) while the fourth refers to the characteristic length ($ChL1$) of that particular point. Every point can have a different characteristic length but if none is supplied here then the default is the overall value ChL. By this procedure one can change the values at the beginning of the code to alter the geometry of the problem or the density of the generated mesh rather that having to re-design part of the domain manually (as in the structured commercial mesh generators). This is where most of the time is saved in generation as compared to other commercial packages.

The next step is to connect two consecutive points in order to form a line, an arc, a spline, an ellipse or any other type of curve. For the current problem (Fig. 4) lines and arcs only were required and these are defined as follows:

```
Line (1) = {1, 2};

Circle (1) = {2, 1, 3};
```

The values within the braces of the former command ("Line") refer to the identification numbers of the points which will be connected. The middle value within the braces of the latter

elementary entity (“Circle”) refers to the origin point of a circle while those on either side refer to the start and end point of the circular arc.

A “Line Loop” command then creates a closed loop of lines, circles or ellipses which are later used to form a surface. In order to create surfaces, a distinction has to be made between a “Plane Surface” (formed by straight lines) and a “Ruled Surface” (formed by curved lines), e.g.

```
Line Loop (1) = {1, 2, 3, 4};
Plane Surface (1) = {3};
Ruled Surface (1) = {5};
```

The numbers within the braces of the “Line Loop” command refer to the identification numbers of the elementary lines or circles. A line loop should form a closed loop in Gmsh. The values within the braces of “Plane Surface” and “Ruled Surface” refer to the identification number of the “Line Loop” command. “Surface Loop” creates a closed loop of elementary surfaces which in turn, later on will form a volume.

```
Surface Loop (1) = {1, 2, 3, 4};

Volume (1) = {1};
```

The values within the braces of the “Surface Loop” command refer to the identification number of the elementary surfaces which will form this loop. The latter has to represent a closed volume. The final command indicates the creation of a volume. The number in the braces of the “Volume” command denotes the identification number of a “Surface Loop”. Codes for all examples described in this paper are included in the Appendix and also from the second author’s webpages.

In order to adjust the level of the tunnel axis to the desired position the user has to define two new parameters (*x1* and *z1*, Fig. 4) at the beginning of the code and introduce them within the braces of the “Point” command. With this procedure the tunnel axis can be relocated horizontally and vertically. Another parameter (*k1*) may be introduced in the code in order to create an inclined tunnel geometry. The opening of the tunnel is fixed at the same level as for the horizontal geometry however the far end is now lower. Therefore *k1* will be introduced within the braces of the “Point” command when defining the coordinates of the end of the tunnel section. When $k1 = 0$ then the axis is horizontally aligned. For non zero values though, the tunnel becomes inclined and the angle of inclination is φ (Fig. 5).

Likewise more parameters can be introduced within the coordinates of the points which will alter the shape and the size of the geometry. They will only be initialised if they have non-zero values.

Uniform unstructured meshing

In order to validate the quality of meshes using this system, a series of generation scenarios are now described, using the code described above and fully presented in the Appendices. In the first case the size (i.e. the characteristic length parameter) of the finite elements generated is the same throughout the domain, termed here uniform unstructured meshing. In a second case the elements within the tunnel and those immediately above have different characteristic length values (*ChL1* and *ChL2*). In both cases the dimensions of the problem and tunnel diameter are constant. The two parameters which vary are the level of the tunnel axis (i.e. *x1, z1*) and the size of the finite elements (*ChL1* and *ChL2*).

In the first case a tunnel with a diameter of D = 4m is driven at a depth of $z_0 = 2.5D$ having its longitudinal axis horizontal. The dimensions of the domain are presented in Fig. 4. *ChL* throughout the domain is *D/5* and *D* in the first and second analyses respectively. The two different *ChL* values are chosen in order to observe how mesh quality is affected by the size of the elements while the position of the tunnel varies. Three different positions are chosen for this purpose (Fig. 4). For the first (p_0) the tunnel is driven in the middle of the domain (i.e. the most favourable position in the mesh). In the second (p_1) the distance between the right springline and the right vertical boundary is just *D/40* (i.e. an unfavourable position and, in practice, not a

mesh an analyst would actually use but instructive in this study nevertheless). Finally in the third (p_2) the distance of both the crown and the right springline from the top and right vertical boundaries respectively is $D/40$ (i.e. the worst position in the whole domain). A total of six meshes are therefore generated.

Fig. 7 shows the meshes produced for each position alongside distribution plots of the number of finite elements generated against their quality γ for a *very fine* mesh ($ChL = D/5$). Table 1 contains statistics for the meshes. From the figure it is obvious that the location of the tunnel does not affect the quality of the finite elements notably, even for the two most unfavourable tunnel positions (i.e. p_1 and p_2). The mean values of γ shown in Table 1 are almost unvarying for each of the three tunnel positions and the number of generated tetrahedra is roughly constant regardless of the position of the tunnel. Hence, the required time for the generation of these elements is roughly the same. The three γ curves are smooth and almost identical having modal values of γ at approximately $\gamma = 0.77$. This is an indication that large areas of the domain consist of elements of good quality (i.e. close to 1). For the two unfavourable positions though (i.e. p_1 and p_2) there are a few tetrahedra (less than 1% of the total number) with $0.25 < \gamma < 0.5$. This can be attributed to the position of the tunnel (i.e. close to the boundaries) since there is insufficient space between the liner and the boundaries to form equilateral tetrahedra.

For a *coarse* mesh ($ChL = D$) Fig. 8 indicates the quality to be strongly affected by the position of the tunnel, unlike in the *very fine* mesh case described above. This time the three γ curves are not identical, having different mean values for every different position (p_0, p_1 and p_2). The mean γ value decreases considerably from p_0 to p_2 (0.67 to 0.49, see Table 2). The maximum γ values correspond to the most favourable position of the tunnel in the domain (p_0) while the minimum value corresponds to the most unfavourable position (p_2). This is an indication of mediocre to bad mesh quality. However, the three modal values are only slightly affected. A comparison of Fig. 8a with Fig. 7a at the same axis position (at p_0) but with distinct ChL values produces acceptable results. At p_1 and p_2 though (Figs 8b and 8c), the projected areas are totally different compared to the cases shown in Figs 7b and 7c. This time the area where most of the elements are projected is shifted to the left, towards the zero γ value which implies irregular tetrahedra and hence poor mesh quality. Various badly shaped tetrahedra are generated (the shape measure of these elements is between $0 < \gamma < 0.5$, see Fig. 8a and 8b) which attempt to fill in the limited space between the liner and the boundaries at the two most unfavourable positions (p_1 and p_2). Hence, the number of nodes and the number of tetrahedral generated increase as the position of the tunnel shifts from p_0 to p_2. This also increases the required time for generation.

In summary, for the case where ChL is the same throughout the domain, it is evident that as the sizes of the elements increase, the quality of the mesh deteriorates, especially when the tunnel is placed closer to the boundaries (positions p_1 and p_2). The analysis performed by Gmsh for the same value of ChL, is robust when the size of the domain consists of *very fine* elements (compared with the size of the domain). This is not the case for a coarser mesh though. For this latter case roughly 57 times fewer nodes are produced and approximately 65 times less time is required to generate the 3D mesh compared to the first case where $ChL = D/5$.

Refined meshes

In order to optimise mesh quality (and hence solution accuracy) at the same time as reducing the computational time needed to perform a 3D analysis a combination of coarse and fine finite elements is required; the former at the mesh boundaries, the latter around the tunnel. Another reason for choosing this strategy is to try to fit the mesh to the chosen geometry. Hence, refinement around the tunnel layout takes place in order to overcome the difficulties which arise in the coarse mesh described above (when $ChL = D$, see Fig. 8). Thus two different ChL values are now introduced ($ChL1$ for the boundaries of the domain, $ChL2$ for the tunnel liner and the area above it. This is achieved by introducing surface nodes which have the same $ChL2$ value as the tunnel liner). All the other parameters vary in the same way as described in the first case. In the analyses performed in this section the following two characteristic length values are used

ChL1 = D and *ChL2 = D/5*. These analyses are carried out in order to assess how local mesh refinement influences the overall quality of the mesh.

Fig. 9 shows meshes and distribution plots for the three positions of the tunnel with the dual *ChL* values and the beneficial impact of refinement is clear by comparing this figure to Fig. 8. (Mesh statistics are shown in Table 3). The mesh quality, as measured by γ, has improved significantly (particularly at the two less favourable positions p_1 and p_2) since the mean and modal values of these plots have shifted further towards unity. Table 3 shows a decrease in the number of nodes (and as a consequence the number of elements) produced and the time required for the generation of the 3D mesh as the tunnel axis moves from p_0 to p_2. This decrease is attributed to the reduction of the refinement zone as the tunnel shifts from the centre of the domain towards the boundaries. Thus fewer refined elements are produced around the tunnel (since they are restricted by the boundaries) and less time is required for the generation of the 3D mesh. By moving the tunnel axis from p_0 to p_1 and p_2 only one half and one third of the nodes of the p_0 case are generated respectively. Consequently the required time for the generation of the 3D mesh reduces compared to the p_0 case. However, the mean γ values differ only slightly.

Mesh refinement significantly improves the quality producing finite elements of equivalent quality to a fine generated mesh (see Figs 7 and 9). Furthermore, larger areas of equilateral tetrahedra are formed in the domain. This is the case particularly when the tunnel lies at its most favourable position within the domain (at p_0). However even when it is placed closer to the boundaries (positions p_1 and p_2) the results are still acceptable.

Further examples

Reference [12] contains a study of the effect of surface patch loading on existing tunnels for which a large number of 3D meshes were required and a balance was sought between acceptable accuracy and computational resources. Examples of the use of the parameterised procedure for these meshes are given here and code for generation of an example is given in Appendix A2.

Single tunnel with surface loading

The dimensions of a typical domain are shown in Fig. 10. Also shown are the points where various characteristic lengths were defined. Three areas of different mesh density can be identified (Figs 10 and 11). At the eight corners of the domain a characteristic length value of *ChL1 = 18D/4* is used. This figure creates relatively large elements at the boundaries. At the points around the tunnel, as well as those on the surface which are used to define the loaded areas, a smaller value is employed (*ChL2 = 3D/4*) to attain greater accuracy around these areas. Finally a third region (*ChL3 = 9D/4*) is used to account for a smoother transition between adjacent elements which have the two previously mentioned extreme values (Fig. 10).

Fig. 11 shows the generated mesh for the case of a single horizontal tunnel and the associated quality distribution. The required time for the generation of the 3D mesh is approximately 1s while the mean γ value is approximately 0.7 (Table 4). From the γ plot it is evident that the vast majority of the finite elements have quality measures above 0.5. This indicates that the quality of the generated tetrahedra in total is above average.

Twin tunnels

The requirement to include a second tunnel in a mesh dramatically affects mesh quality as compared to the single tunnel case presented above. The geometric characteristics of the domain are the same as for the single tunnel case. Three new parameters are introduced which are used to position the second tunnel within the domain (Fig. 12). These are *x2*, *z2* and *k2*. The first two shift the tunnel axis horizontally and vertically respectively while the third indicates the inclination of the tunnel along the *z*-axis. The finite elements adjacent to the second tunnel could have a different characteristic length to those adjacent to the first. In this case though, it was decided to use elements of the same characteristic length for both tunnels. Thus, three different density areas (identical to those described above) are identified in the domain.

Three different geometries of multiple tunnelling are examined in this section.

- The case of twin tunnelling when both are horizontally aligned (*TH* case, Fig. 12a).
- The case of twin tunnelling where the tunnels are vertically aligned (*TV* case, Fig. 12b).
- The case of multiple tunnels where the first tunnel is horizontally aligned while the second is inclined (*MHI* case, the inclination angle is 4^o, Fig. 13).

Fig. 12 shows the generated meshes and the corresponding quality distribution plots for the *TH* and *TV* cases. Code for generation of an example is given in Appendix A3. Statistics for these and the *MHI* case are given in Table 5. For the *TH* case the pillar width $P = 1D$ while for the *TV* case the pillar depth is $P_D = 1D$. The number of nodes (and consequently the number of finite elements) produced are slightly higher in the *TH* case. This can be attributed to the wider zone of fine generated tetrahedra ($ChL2 = 3D/4$). Consequently, it takes slightly longer to generate the mesh. In both cases the mean γ value is in excess of 0.7 while the vast majority of the finite elements have quality measures above 0.5. Once more this is an indication of good quality tetrahedral elements. Here it can be said that the position of a second tunnel only marginally affects the quality of the mesh.

In the *MHI* case where the axis of the second tunnel is inclined by approximately 20%, more finite elements are generated (creating a large dense area around the tunnels) compared to the other two cases (*TH* and *TV*), for the same reason described above. Thus more time is required for the meshing (roughly 30%). Given that more fine elements are generated for this particular geometry, the overall quality of the mesh slightly improves as shown by a rise in the mean γ value which in this case increases to 0.72.

By comparing the mean and modal values as well as the shapes of the γ distribution plots for the single and twin tunnel cases (see Figs 11 to 13 and Tables 4 and 5) the beneficial impact of the second tunnel to the quality of the mesh is identified. The explanation is that the existence of the second tunnel introduces another dense area similar to the single tunnel case. This refinement improves the overall condition of the generated finite elements. However, more nodes (and hence, more tetrahedra) are generated (almost 80%) which in turn increases the amount of time required for the generation of the 3D mesh by almost 50% for the *TH* and *TV* cases and by 100% for the *MHI* case.

Modelling excavation

One of the most commonly analysed tunnelling problems must be the simulation of tunnel excavation. This may involve soil-tunnel interaction (various tunnel geometries) or soil-tunnel-structure interaction (where the “structure” can be a second tunnel, a building or a piled foundation). The following example indicates the additional difficulties that are associated with generating appropriate meshes for single tunnel excavation and lining instalment in consecutive steps. The problems described in the previous sections of the paper differ from that presented here, since they were for meshes with pre-installed tunnels (i.e. a void within the mesh).

Fig. 14 shows a domain (which has the same dimensions as that described in Fig. 10) containing a single tunnel split into smaller volumes. Simulation of continuous tunnel excavation then consists of removing these volumes in turn (and dealing with the changing boundary conditions as described elsewhere [5]). In order to define the size of an excavation increment a new variable *Ur* is introduced. *Ur* represents the unsupported region which is created when soil elements are removed from the face of the excavation while the lining is installed at a specific distance behind the face. The importance of the length of this region to the accuracy of any analysis is highlighted in [17]. Three different groups of elements are used in this mesh: *i*) ten-noded tetrahedra representing the soil in the domain outside the tunnel, *ii*) ten-noded tetrahedra representing the soil to be excavated inside the tunnel and finally *iii*) six-noded triangles representing the tunnel lining.

An important feature of Gmsh is the consecutive numbering of the series of volumes into which the tunnel is split. Thus, the user is not required to know the number of every element to be excavated in the particular volume, even though Gmsh provides this kind of information.

Instead, the knowledge of the identification number of that volume is enough to enable inoput data to be written to cover element removal in groups. Code for generation of an example mesh is given in Appendix A4.

Mesh quality as measured by γ appears to deteriorate on comparison of Figs 11 and 14. This is attributed to the discontinuities which are formed between the consecutive smaller volumes of the tunnel. However, the majority of the generated finite elements are projected above average quality ($\gamma > 0.5$) with a mean value of $\gamma = 0.687$. Hence, the domain consists of generated elements of good quality. Statistics for this case are presented in Table 6. The computational time needed to discretise the whole domain is almost 50% higher than that shown in Fig. 11 since twice the number of finite elements are generated.

In order to emphasize the capabilities of Gmsh to generate even more complicated tunnelling geometries, a final example based on case studies from the Channel Tunnel Project is presented for illustrative purposes only in Fig. 15 showing parallel, multiple tunnels which intersect with others of smaller diameter.

Summary

Gmsh is a fully automated 3D unstructured tetrahedral freeware mesh generator which works either via a GUI or from an ASCII file containing commands. Two of its main capabilities are: *i*) the possibility of variable substitution in the input file and *ii*) the specification of "characteristic lengths" which can be applied to selected points in the model. In this way, the user can produce areas of different density within the domain. Various tests were undertaken here to examine the capabilities of Gmsh for tunnelling analyses in particular. From these it was found that this software can rapidly and easily produce 3D meshes of good quality even for the most unfavourable positions of a tunnel in the domain, and for the most complicated tunnelling schemes. Use of parameterised files of commands to generate meshes has been shown to be an effective way of pre-processing for parametric studies.

Acknowledgements

This work was funded by a studentship from the UK Engineering and Physical Sciences Research Council (EPSRC), the School of Engineering at Durham University and Halcrow Ltd.

References

[1] Boone, S.J. (1996) Ground-movement-related building damage, *ASCE Journal of Geotechnical Engineering*, Vol. 122, No. 11, 886-896

[2] Boscardin, M.D. and Cording, E.J. (1989) Building response to excavation-induced settlement. *ASCE Journal of Geotechnical Engineering,* Vol. 115, No. 1, 1-21

[3] Burd, H.J., Houlsby, G.T., Augarde, C.E. and Liu, G. (2000) Modelling tunnelling-induced settlement of masonry buildings, *Proc. Instn. Civ. Engrs. Geotech. Engng.,* 143, pp 17-29

[4] Yeow, H.C., Wong, C., Pillai, A. and Simpson, B. (2005) The use of a substructure method and three-dimensional finite element modelling in assessment of damage due to underground tunnelling. Prediction, analysis and design in geomechanical applications. *Proc. 11th IACMAG Conference, Turin, June 19-24,* Patron Ed. Bologna, Vol. 2, p 711-718. ISBN 88-555-2812-2

[5] Augarde, C.E. and Burd, H.J. (2001) 3D finite element analysis of lined tunnels, *Int. J. Num. Anal. Meth. Geomech.* 25, pp 243-262.

[6] Franzius, J.N., Potts, D.M. and Burland, J.B. (2005) The influence of soil anisotropy and K_0 on ground surface movements resulting from tunnel excavation, *Geotechnique* 55, No. 3, 189-199

[7] Jenck, O. and Dias, D. (2005) Tunnelling on urban areas: 3D numerical analysis of soil/structure interaction. *5th International Symposium Geotechnical Aspects of Underground Construction in Soft Ground.* Amsterdam 15-17 June, Preprint

[8] Senechal, M., (1981) Which tetrahedra fill space?, *Mathematics Magazine,* 54(5), 227-243

[9] Frey, P.J., and George, P.L. (2000) *Mesh generation application to finite elements*, Hermes Science Publishing, Oxford, UK

[10] Thompson, J.F., Soni, B.K. and Weatherill, N.P. (1999) Editors, *Handbook of grid generation*, CRS Press, Boca Raton, USA

[11] Bell, R.W., Houlsby, G.T. and Burd, H.J. (1991) Suitability of Two and Three Dimensional Finite Elements for Modelling Material Incompressibility Using Exact Integration. *Communications in Applied Numerical Methods in Engineering*, 36(14), 2453-2472.

[12] Koungelis DK. (2007). Tools for numerical modelling of tunnelling interactions. PhD Thesis, School of Engineering. Durham University.

[13] Http://www.guez.org/gmsh/

[14] Field, D.A., (2000) Qualitative measures for initial meshes, *International Journal for Numerical Methods in Engineering*, 47, 887-906

[15] Naylor, D.J. (1999) Filling space with tetrahedra, *International Journal for Numerical Methods in Engineering,* 44(10), 1383-1395

[16] Topping B.H.V, Muylle, J., Ivanyi, P., Putanowicz, R. and Cheng, B. (2004) Finite element mesh generation, Saxe-Coburg Publications, Stirling, Scotland

[17] Vermeer, P.A., Bonnier, P.G. and Moller, S.C. (2002) On a smart use of 3D-FEM in tunnelling. In: *Proc. of the 8th Int. Symp. Num. Models Geomech – NUMOG VIII, Rome, 10-12 April 2002*. Rotterdam: Balkema. pp. 361-366.

Figure captions

Tables

PC characteristics		Processor		CPU		RAM	
		Intel (R)		2.66GHz		192MB	
D (m)	4	Position	No. of nodes	No. of 10-noded Tetrahedra	Time for 2D (s)	Time for 3D (s)	*Gamma (γ)* (mean)
X (m)	8*D*						
Y (m)	5*D*						
Z (m)	5*D*	p_0	229,726	163,140	< 1.0	66.3	0.769
ChL (m)	*D*/5	p_1	228,985	162,541	< 1.0	64.3	0.768
z_0 (m)	Z/2	p_2	230,185	163,471	< 1.0	68.8	0.767

Table 1: Mesh statistics for uniform unstructured meshing (Very fine case, *ChL*= *D*/5).

D (m)	4	Position	No. of nodes	No. of 10-noded Tetrahedra	Time for 2D (s)	Time for 3D (s)	*Gamma (γ)* (mean)
X (m)	8*D*						
Y (m)	5*D*						
Z (m)	5*D*	p_0	3,123	1,546	0.04	0.65	0.670
ChL (m)	*D*	p_1	4,128	2,078	0.03	0.8	0.529
z_0 (m)	Z/2	p_2	4,894	2,493	0.04	1.0	0.488

Table 2: Mesh statistics for uniform unstructured meshing (Coarse case, *ChL* = *D*).

D (m)	4	Position	No. of nodes	No. of 10-noded Tetrahedra	Time for 2-D (s)	Time for 3D (s)	*Gamma (γ)* (mean)
X (m)	8*D*						
Y (m)	5*D*						
Z (m)	5*D*						
ChL1 (m)	*D*	p_0	62,888	44,153	0.26	15.5	0.670
ChL2 (m)	*D*/5	p_1	36,304	24,159	0.23	8.4	0.664
z_0 (m)	Z/2	p_2	24,120	15,169	0.22	5.6	0.653

Table 3: Mesh statistics for refined meshes (*ChL*1 = *D* and *ChL*2 = *D*/5).

X (m)	*Y* (m)	*Z* (m)	*D* (m)	z_0 (m)	*ChL1* (m)	*ChL2* (m)	*ChL3* (m)	*ChL4* (m)

17.5*D*	17.5*D*	12.5*D*	4	15	*18D*/4	*3D*/4	*9D/4*	*3D*/4
No. of nodes		No. of 10-noded Tetrahedra			Time for 2D (s)	Time for 3D (s)	*Gamma (γ)* (mean)	
4,574		2,929			< 0.1	0.9	0.701	

Table 4: Mesh statistics for the single horizontal tunnel case with surface loading.

	X (m)	*Y* (m)	*Z* (m)	*D* (m)	z_0 (m)	*ChL1* (m)	*ChL2* (m)	*ChL3* (m)	*ChL4* (m)
	17.5*D*	17.5*D*	12.5*D*	4	15	*18D*/4	*3D*/4	*9D/4*	*3D*/4
	No. of nodes		No. of 10-noded Tetrahedra			Time for 2D (s)	Time for 3D (s)	*Gamma (γ)* (mean)	
TH case	7,714		5,038			< 0.1	1.484	0.707	
TV case	7,502		4,878			< 0.1	1.359	0.713	
MHI case	9,277		6,207			< 0.1	2.234	0.723	

Table 5: Mesh statistics for twin tunnels with surface loading.

.

X (m)	*Y* (m)	*Z* (m)	*D* (m)	z_0 (m)	*ChL1* (m)	*ChL2* (m)	*ChL3* (m)	*ChL4* (m)
17.5*D*	17.5*D*	12.5*D*	4	15	*18D*/4	*3D*/4	*9D/4*	*3D*/4
No. of nodes		No. of 10-noded Tetrahedra			Time for 2-D (s)	Time for 3D (s)	*Gamma (γ)* (mean)	
8,793		5,824			0.25	1.64	0.687	

Table 6. Mesh for the single horizontal tunnel excavation case.

Appendices

Code for mesh generation using GMSH
A1 EQUAL SIZE ELEMENTS

```
D = 4;                  //Tunnel diameter
X = 8*D;                //Dimension of the domain along the x axis
Y = 5*D;                //Dimension of the domain along the y axis
Z = 5*D;                //Dimension of the domain along the z axis
ChL1 = D/5;         //Size of the elements at the boundaries
ChL2 = D/5;         //Size of the elements around the tunnel and at the foundations
x1=X/2;//+X/2-21*D/40;              //Parameter which shifts the tunnel axis horizontally
z1=Z/2;//+Z/2-21*D/40;          //Parameter which shifts the tunnel axis vertically
k1=0;               //Parameter which shifts the end of the tunnel only along the z axis

// Points at the boundaries of the domain
Point (1) = {0, 0, 0, ChL1};          Point (2) = {X, 0, 0, ChL1};
Point (3) = {X, Y, 0, ChL1};         Point (4) = {0, Y, 0, ChL1};
Point (5) = {0, 0, Z, ChL1};          Point (6) = {X/2-6, 0, Z, ChL2};
Point (7) = {X/2+6, 0, Z, ChL2};   Point (8) = {X, 0, Z, ChL1};
Point (9) = {X, Y, Z, ChL1};         Point (10) = {X/2+6, Y, Z, ChL2};
Point (11) = {X/2-6, Y, Z, ChL2}; Point (12) = {0, Y, Z, ChL1};

//Points at the tunnel entrance
Point (13) = {x1, 0, z1, ChL2};          Point (14) = {x1, 0, z1 + D/2, ChL2};
Point (15) = {x1 - D/2, 0, z1, ChL2};  Point (16) = {x1, 0, z1 - D/2, ChL2};
Point (17) = {x1 + D/2, 0, z1, ChL2};

//Points at the tunnel exit
Point (18) = {x1, Y, z1 + k1, ChL2};          Point (19) = {x1, Y, z1 + D/2 + k1, ChL2};
Point (20) = {x1 - D/2, Y, z1 + k1, ChL2};  Point (21) = {x1, Y, z1 - D/2 + k1, ChL2};
Point (22) = {x1 + D/2, Y, z1 + k1, ChL2};

//Creating the boundaries of the domain
Line (1) = {1, 2};      Line (2) = {2, 3};
Line (3) = {3, 4};      Line (4) = {4, 1};
Line (5) = {5, 6};      Line (6) = {6, 7};
Line (7) = {7, 8};      Line (8) = {8, 9};
Line (9) = {9, 10};    Line (10) = {10, 11};
Line (11) = {11, 12}; Line (12) = {12, 5};
Line (13) = {1, 5};     Line (14) = {2, 8};
Line (15) = {3, 9};     Line (16) = {4, 12};

//Creating the entrance opening
Circle (17) = {14, 13, 15};  Circle (18) = {15, 13, 16};
Circle (19) = {16, 13, 17};  Circle (20) = {17, 13, 14};

//Creating the exit opening
Circle (21) = {19, 18, 20};  Circle (22) = {20, 18, 21};
Circle (23) = {21, 18, 22};  Circle (24) = {22, 18, 19};

//Connecting the entrance and exit of the tunnel
Line (25) = {14, 19};   Line (26) = {15, 20};
Line (27) = {16, 21};   Line (28) = {17, 22};

//Create the "Surfaces" of the domain
Line Loop (49) = {17, 18, 19, 20}; //Plane Surface (50) = {49};
Line Loop (51) = {21, 22, 23, 24}; //Plane Surface (52) = {51};

Line Loop (57) = {-13, 1, 14, -7, -6, -5}; Plane Surface (58) = {57, 49};
Line Loop (59) = {-14, 2, 15, -8}; Plane Surface (60) = {59};
```

```
Line Loop (61) = {-15, 3, 16, -11, -10, -9}; Plane Surface (62) = {61, 51};
Line Loop (63) = {-16, 4, 13, -12}; Plane Surface (64) = {63};
Line Loop (65) = {1, 2, 3, 4}; Plane Surface (66) = {65};
Line Loop (67) = {-11, -10, -9, -8, -7, -6, -5, -12}; Plane Surface (68) = {67};

//Create the "Surfaces" of the tunnel
Line Loop (69) = {17, 26, -21, -25}; Ruled Surface (70) = {69};
Line Loop (71) = {18, 27, -22, -26}; Ruled Surface (72) = {71};
Line Loop (73) = {19, 28, -23, -27}; Ruled Surface (74) = {73};
Line Loop (75) = {20, 25, -24, -28}; Ruled Surface (76) = {75};

//Create the boundaries of the domain
Physical Surface (1) = {58};
Physical Surface (2) = {60};
Physical Surface (3) = {62};
Physical Surface (4) = {64};
Physical Surface (5) = {66};

//Create the boundaries of the tunnel
Physical Surface (6) = {70};
Physical Surface (7) = {72};
Physical Surface (8) = {74};
Physical Surface (9) = {76};

//Create the volume of the domain
Surface Loop (1) = {58, 60, 62, 64, 66, 68, 70, 72, 74, 76};
Volume (1) = {1};
```

A2 SINGLE TUNNEL GEOMETRY

```
D = 4;              //Tunnel diameter
DD=4;               //Size of the foundations
X = 17.5*D;         //Dimension of the domain along the x axis
Y = 17.5*D;         //Dimension of the domain along the y axis
Z = 12.5*D;         //Dimension of the domain along the z axis
ChL1 = 18;          //Size of the elements at the boundaries
ChL2 = 3;           //Size of the elements around the tunnel
ChL3 = 9;           //Size of the transition elements
ChL4 = 3;           //Size of the elements around at the foundations
x1=X/2;             //Parameter which shifts the tunnel axis horizontally
z1=Z-15;            //Parameter which shifts the tunnel axis vertically
k1=0;               //Parameter which shifts the end of the tunnel only along the z axis

//Points at the tunnel entrance
Point (1) = {x1, 0, z1, ChL2};          Point (2) = {x1, 0, z1+D/2, ChL2};
Point (3) = {x1-D/2, 0, z1, ChL2};   Point (4) = {x1, 0, z1-D/2, ChL2};
Point (5) = {x1+D/2, 0, z1, ChL2};

//Points at the tunnel exit
Point (6) = {x1, Y, z1+k1, ChL2};          Point (7) = {x1, Y, z1+k1+D/2, ChL2};
Point (8) = {x1-D/2, Y, z1+k1, ChL2};   Point (9) = {x1, Y, z1+k1-D/2, ChL2};
Point (10) = {x1+D/2, Y, z1+k1, ChL2};

//Points at the boundaries of the domain
Point (19) = {0, 0, 0, ChL1};           Point (20) = {X, 0, 0, ChL1};
Point (21) = {X, Y, 0, ChL1};           Point (22) = {0, Y, 0, ChL1};
Point (23) = {0, 0, Z, ChL1};           Point (24) = {X/2-D, 0, Z, ChL3};
Point (25) = {X/2+D, 0, Z, ChL3};   Point (26) = {X, 0, Z, ChL1};
Point (27) = {X, Y, Z, ChL1};           Point (28) = {X/2+D, Y, Z, ChL3};
Point (29) = {X/2-D, Y, Z, ChL3};   Point (30) = {0, Y, Z, ChL1};
```

```
//Points at the foundations
Point (64) = {X/2-2.5*DD, Y/2-DD/2, Z, ChL4};
Point (65) = {X/2-1.5*DD, Y/2-DD/2, Z, ChL4};
Point (66) = {X/2-0.5*DD, Y/2-DD/2, Z, ChL4};
Point (67) = {X/2+0.5*DD, Y/2-DD/2, Z, ChL4};

Point (72) = {X/2-2.5*DD, Y/2+DD/2, Z, ChL4};
Point (73) = {X/2-1.5*DD, Y/2+DD/2, Z, ChL4};
Point (74) = {X/2-0.5*DD, Y/2+DD/2, Z, ChL4};
Point (75) = {X/2+0.5*DD, Y/2+DD/2, Z, ChL4};

Point (80) = {X/2-2.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (81) = {X/2-1.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (82) = {X/2-0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (83) = {X/2+0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};

Point (88) = {X/2-2.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (89) = {X/2-1.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (90) = {X/2-0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (91) = {X/2+0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};

//Connecting the entrance and exit of the tunnel
Line (1) = {2, 7};    Line (2) = {3, 8};
Line (3) = {4, 9};    Line (4) = {5, 10};

//Creating the entrance opening
Circle (9) = {2, 1, 3};     Circle (10) = {3, 1, 4};
Circle (11) = {4, 1, 5};    Circle (12) = {5, 1, 2};

//Creating the exit opening
Circle (13) = {7, 6, 8};     Circle (14) = {8, 6, 9};
Circle (15) = {9, 6, 10};    Circle (16) = {10, 6, 7};

//Creating the boundaries of the domain
Line (25) = {19, 20};    Line (26) = {20, 21};
Line (27) = {21, 22};    Line (28) = {22, 19};

Line (29) = {23, 24};    Line (30) = {24, 25};
Line (31) = {25, 26};    Line (32) = {26, 27};
Line (33) = {27, 28};    Line (34) = {28, 29};
Line (35) = {29, 30};    Line (36) = {30, 23};

Line (37) = {23, 19};    Line (38) = {26, 20};
Line (39) = {27, 21};    Line (40) = {30, 22};

//Creating the foundations
Line (63) = {64, 65};   Line (64) = {65, 66};
Line (65) = {66, 67};

Line (70) = {72, 73};   Line (71) = {73, 74};
Line (72) = {74, 75};

Line (77) = {80, 81};   Line (78) = {81, 82};
Line (79) = {82, 83};

Line (84) = {88, 89};   Line (85) = {89, 90};
Line (86) = {90, 91};

Line (96) = {88, 80};   Line (97) = {80, 72};
Line (98) = {72, 64};
```

```
Line (102) = {89, 81};   Line (103) = {81, 73};
Line (104) = {73, 65};

Line (108) = {90, 82};   Line (109) = {82, 74};
Line (110) = {74, 66};

Line (114) = {91, 83};   Line (115) = {83, 75};
Line (116) = {75, 67};

//Creating loops for the tunnel
Line Loop (1) = {9, 10, 11, 12};    Line Loop (3) = {13, 14, 15, 16};

Line Loop (5) = {9, 2, -13, -1};     Line Loop (6) = {10, 3, -14, -2};
Line Loop (7) = {11, 4, -15, -3};   Line Loop (8) = {12, 1, -16, -4};

//Creating loops for the boundaries of the domain
Line Loop (13) = {25, 26, 27, 28};   Line Loop (14) = {29, 30, 31, 32, 33, 34, 35, 36};

Line Loop (15) = {37, 25, -38, -31, -30, -29};   Line Loop (16) = {38, 26, -39, -32};
Line Loop (17) = {39, 27, -40, -35, -34, -33};   Line Loop (18) = {40, 28, -37, -36};

//Creating loops for the foundations
Line Loop (41) = {63, -104, -70, 98};
Line Loop (42) = {64, -110, -71, 104};
Line Loop (43) = {65, -116, -72, 110};

Line Loop (48) = {70, -103, -77, 97};
Line Loop (49) = {71, -109, -78, 103};
Line Loop (50) = {72, -115, -79, 109};

Line Loop (55) = {77, -102, -84, 96};
Line Loop (56) = {78, -108, -85, 102};
Line Loop (57) = {79, -114, -86, 108};

Line Loop (61) = {63, 64, 65, -116, -115, -114, -86, -85, -84, 96, 97, 98};

//Creating surfaces for the tunnel
//Plane Surface (1) = {1};   //Plane Surface (2) = {3};
Ruled Surface (3) = {5};    Ruled Surface (4) = {6};
Ruled Surface (5) = {7};    Ruled Surface (6) = {8};

//Creating surfaces for the boundaries of the domain
Plane Surface (13) = {13};        Plane Surface (14) = {14, 61};
Plane Surface (15) = {15, 1};     Plane Surface (16) = {16};
Plane Surface (17) = {17, 3};     Plane Surface (18) = {18};

//Creating surfaces for the foundations
Plane Surface (41) = {41};
Plane Surface (42) = {42};
Plane Surface (43) = {43};

Plane Surface (48) = {48};
Plane Surface (49) = {49};
Plane Surface (50) = {50};

Plane Surface (55) = {55};
Plane Surface (56) = {56};
Plane Surface (57) = {57};
```

```
Plane Surface (61) = {61};

//Creating physical surfaces for the tunnel
//Physical Surface (1) = {1};    //Physical Surface (2) = {2};
Physical Surface (3) = {3};     Physical Surface (4) = {4};
Physical Surface (5) = {5};     Physical Surface (6) = {6};

//Creating physical surfaces for the boundaries of the domain
Physical Surface (13) = {13};   Physical Surface (14) = {14};
Physical Surface (15) = {15};   Physical Surface (16) = {16};
Physical Surface (17) = {17};   Physical Surface (18) = {18};

//Creating physical surfaces for the foundations
Physical Surface (41) = {41};
Physical Surface (42) = {42};
Physical Surface (43) = {43};

Physical Surface (48) = {48};
Physical Surface (49) = {49};
Physical Surface (50) = {50};

Physical Surface (55) = {55};
Physical Surface (56) = {56};
Physical Surface (57) = {57};

Physical Surface (61) = {61};

//Creating the volume of the domain
Surface Loop (1) = {13, 14, 15, 16, 17, 18, 3, 4, 5, 6, 41, 42, 43, 48, 49, 50, 55, 56, 57};
Volume (1) = {1};
Physical Volume (1) = {1};
```

A3 TWIN TUNNEL GEOMETRY

```
D = 4;                  //Tunnel diameter
DD = 4;                 //Size of the foundations
X = 17.5*D;             //Dimension of the domain along the x axis
Y = 17.5*D;             //Dimension of the domain along the y axis
Z = 12.5*D;             //Dimension of the domain along the z axis
ChL1 = 18;              //Size of the elements at the boundaries
ChL2 = 3;               //Size of the elements at the tunnels and the foundations
ChL3 = 9;               //Size of the transition elements
x1 =X/2;                //Parameter which shifts the right tunnel axis horizontally
z1 =Z-15;               //Parameter which shifts the right tunnel axis vertically
k1 = 0;                 //Parameter which shifts the end of the right tunnel only along the z axis
f  = 2*D;               //Pillar width
x2 =X/2-f;              //Parameter which shifts the left tunnel axis horizontally
z2 = Z-15;              //Parameter which shifts the left tunnel axis vertically
k2 = 0;                 //Parameter which shifts the end of the left tunnel only along the z axis

//Points at the right tunnel entrance
Point (1) = {x1, 0, z1, ChL2};          Point (2) = {x1, 0, z1+D/2, ChL2};
Point (3) = {x1-D/2, 0, z1, ChL2};      Point (4) = {x1, 0, z1-D/2, ChL2};
Point (5) = {x1+D/2, 0, z1, ChL2};

//Points at the right tunnel exit
Point (6) = {x1, Y, z1+k1, ChL2};           Point (7) = {x1, Y, z1+D/2+k1, ChL2};
Point (8) = {x1-D/2, Y, z1+k1, ChL2};       Point (9) = {x1, Y, z1-D/2+k1, ChL2};
Point (10) = {x1+D/2, Y, z1+k1, ChL2};
```

```
//Points at the boundaries of the domain
Point (19) = {0, 0, 0, ChL1};            Point (20) = {X, 0, 0, ChL1};
Point (21) = {X, Y, 0, ChL1};            Point (22) = {0, Y, 0, ChL1};
Point (23) = {0, 0, Z, ChL1};            Point (24) = {X/2-D, 0, Z, ChL3};
Point (25) = {X/2+D, 0, Z, ChL3};    Point (26) = {X, 0, Z, ChL1};
Point (27) = {X, Y, Z, ChL1};            Point (28) = {X/2+D, Y, Z, ChL3};
Point (29) = {X/2-D, Y, Z, ChL3};    Point (30) = {0, Y, Z, ChL1};

//Points at the left tunnel entrance
Point (31) = {x2-f, 0, z2+k2, ChL2};
Point (32) = {x2-f, 0, z2+k2+D/2, ChL2};
Point (33) = {x2-D/2-f, 0, z2+k2, ChL2};
Point (34) = {x2-f, 0, z2+k2-D/2, ChL2};
Point (35) = {x2+D/2-f, 0, z2+k2, ChL2};

//Points at the left tunnel exit
Point (36) = {x2-f, Y, z2+k2, ChL2};
Point (37) = {x2-f, Y, z2+k2+D/2, ChL2};
Point (38) = {x2-D/2-f, Y, z2+k2, ChL2};
Point (39) = {x2-f, Y, z2+k2-D/2, ChL2};
Point (40) = {x2+D/2-f, Y, z2+k2, ChL2};

//Points at the foundations
Point (62) = {X/2-4.5*DD, Y/2-DD/2, Z, ChL4};
Point (63) = {X/2-3.5*DD, Y/2-DD/2, Z, ChL4};
Point (64) = {X/2-2.5*DD, Y/2-DD/2, Z, ChL4};
Point (65) = {X/2-1.5*DD, Y/2-DD/2, Z, ChL4};
Point (66) = {X/2-0.5*DD, Y/2-DD/2, Z, ChL4};
Point (67) = {X/2+0.5*DD, Y/2-DD/2, Z, ChL4};
Point (68) = {X/2+1.5*DD, Y/2-DD/2, Z, ChL4};

Point (70) = {X/2-4.5*DD, Y/2+DD/2, Z, ChL4};
Point (71) = {X/2-3.5*DD, Y/2+DD/2, Z, ChL4};
Point (72) = {X/2-2.5*DD, Y/2+DD/2, Z, ChL4};
Point (73) = {X/2-1.5*DD, Y/2+DD/2, Z, ChL4};
Point (74) = {X/2-0.5*DD, Y/2+DD/2, Z, ChL4};
Point (75) = {X/2+0.5*DD, Y/2+DD/2, Z, ChL4};
Point (76) = {X/2+1.5*DD, Y/2+DD/2, Z, ChL4};

Point (78) = {X/2-4.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (79) = {X/2-3.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (80) = {X/2-2.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
 Point (81) = {X/2-1.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (82) = {X/2-0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (83) = {X/2+0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (84) = {X/2+1.5*DD, Y/2+1*DD+DD/2, Z, ChL4};

Point (86) = {X/2-4.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (87) = {X/2-3.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (88) = {X/2-2.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (89) = {X/2-1.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (90) = {X/2-0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (91) = {X/2+0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (92) = {X/2+1.5*DD, Y/2+2*DD+DD/2, Z, ChL4};

//Connecting the entrance and exit of the right tunnel
Line (1) = {2, 7};     Line (2) = {3, 8};
Line (3) = {4, 9};     Line (4) = {5, 10};

//Creating the entrance opening of the right tunnel
```

```
Circle (9) = {2, 1, 3};      Circle (10) = {3, 1, 4};
Circle (11) = {4, 1, 5};    Circle (12) = {5, 1, 2};

//Creating the exit opening of the right tunnel
Circle (13) = {7, 6, 8};      Circle (14) = {8, 6, 9};
Circle (15) = {9, 6, 10};    Circle (16) = {10, 6, 7};

//Creating the boundaries of the domain
Line (25) = {19, 20};    Line (26) = {20, 21};
Line (27) = {21, 22};    Line (28) = {22, 19};

Line (29) = {23, 24};   Line (30) = {24, 25};
Line (31) = {25, 26};   Line (32) = {26, 27};
Line (33) = {27, 28};   Line (34) = {28, 29};
Line (35) = {29, 30};   Line (36) = {30, 23};

Line (37) = {23, 19};   Line (38) = {26, 20};
Line (39) = {27, 21};   Line (40) = {30, 22};

//Creating the entrance opening of the left tunnel
Circle (41) = {32, 31, 33};   Circle (42) = {33, 31, 34};
Circle (43) = {34, 31, 35};   Circle (44) = {35, 31, 32};

//Creating the exit opening of the left tunnel
Circle (45) = {37, 36, 38};   Circle (46) = {38, 36, 39};
Circle (47) = {39, 36, 40};   Circle (48) = {40, 36, 37};

//Connecting the entrance and exit of the right tunnel
Line (49) = {32, 37};   Line (50) = {33, 38};
Line (51) = {34, 39};   Line (52) = {35, 40};

//Creating the foundations
Line (61) = {62, 63};   Line (62) = {63, 64};
Line (63) = {64, 65};   Line (64) = {65, 66};
Line (65) = {66, 67};   Line (66) = {67, 68};

Line (68) = {70, 71};   Line (69) = {71, 72};
Line (70) = {72, 73};   Line (71) = {73, 74};
Line (72) = {74, 75};   Line (73) = {75, 76};

Line (75) = {78, 79};   Line (76) = {79, 80};
Line (77) = {80, 81};   Line (78) = {81, 82};
Line (79) = {82, 83};   Line (80) = {83, 84};

Line (82) = {86, 87};   Line (83) = {87, 88};
Line (84) = {88, 89};   Line (85) = {89, 90};
Line (86) = {90, 91};   Line (87) = {91, 92};

Line (96) = {88, 80};   Line (97) = {80, 72};
Line (98) = {72, 64};

Line (102) = {89, 81};   Line (103) = {81, 73};
Line (104) = {73, 65};

Line (108) = {90, 82};   Line (109) = {82, 74};
Line (110) = {74, 66};

Line (114) = {91, 83};   Line (115) = {83, 75};
Line (116) = {75, 67};
```

```
Line (120) = {92, 84};   Line (121) = {84, 76};
Line (122) = {76, 68};

Line (126) = {87, 79};   Line (127) = {79, 71};
Line (128) = {71, 63};

Line (129) = {86, 78};   Line (130) = {78, 70};
Line (131) = {70, 62};

//Creating loops for the right tunnel
Line Loop (1) = {9, 10, 11, 12};   Line Loop (3) = {13, 14, 15, 16};

Line Loop (5) = {9, 2, -13, -1};    Line Loop (6) = {10, 3, -14, -2};
Line Loop (7) = {11, 4, -15, -3};  Line Loop (8) = {12, 1, -16, -4};

//Creating loops for the boundaries of the domain
Line Loop (13) = {25, 26, 27, 28};   Line Loop (14) = {29, 30, 31, 32, 33, 34, 35, 36};

Line Loop (15) = {37, 25, -38, -31, -30, -29};    Line Loop (16) = {38, 26, -39, -32};
Line Loop (17) = {39, 27, -40, -35, -34, -33};    Line Loop (18) = {40, 28, -37, -36};

//Creating loops for the left tunnel
Line Loop (19) = {41, 42, 43, 44};   Line Loop (20) = {45, 46, 47, 48};

Line Loop (21) = {41, 50, -45, -49};   Line Loop (22) = {42, 51, -46, -50};
Line Loop (23) = {43, 52, -47, -51};   Line Loop (24) = {44, 49, -48, -52};

//Creating loops for the foundations
Line Loop (40) = {62, -98, -69, 128};     Line Loop (41) = {63, -104, -70, 98};
Line Loop (42) = {64, -110, -71, 104};   Line Loop (43) = {65, -116, -72, 110};
Line Loop (44) = {66, -122, -73, 116};

Line Loop (47) = {69, -97, -76, 127};     Line Loop (48) = {70, -103, -77, 97};
Line Loop (49) = {71, -109, -78, 103};   Line Loop (50) = {72, -115, -79, 109};
Line Loop (51) = {73, -121, -80, 115};

Line Loop (54) = {76, -96, -83, 126};     Line Loop (55) = {77, -102, -84, 96};
Line Loop (56) = {78, -108, -85, 102};   Line Loop (57) = {79, -114, -86, 108};
Line Loop (58) = {80, -120, -87, 114};

Line Loop (61) = {61, -128, -68, 131};   Line Loop (62) = {68, -127, -75, 130};
Line Loop (63) = {75, -126, -82, 129};

Line Loop (64) = {61, 62, 63, 64, 65, 66, -122, -121, -120, -87, -86, -85, -84, -83, -82, 129, 130, 131};

//Creating surfaces for the right tunnel
//Plane Surface (1) = {1};   //Plane Surface (2) = {3};
Ruled Surface (3) = {5};   Ruled Surface (4) = {6};
Ruled Surface (5) = {7};   Ruled Surface (6) = {8};

//Creating surfaces for the left tunnel
//Plane Surface (7) = {19};   //Plane Surface (8) = {20};
Ruled Surface (9) = {21};     Ruled Surface (10) = {22};
Ruled Surface (11) = {23};   Ruled Surface (12) = {24};

//Creating surfaces for the boundaries of the domain
Plane Surface (13) = {13};            Plane Surface (14) = {14, 64};
Plane Surface (15) = {15, 1, 19};    Plane Surface (16) = {16};
Plane Surface (17) = {17, 3, 20};    Plane Surface (18) = {18};
```

```
//Creating surfaces for the foundations
Plane Surface (40) = {40};    Plane Surface (41) = {41};
Plane Surface (42) = {42};    Plane Surface (43) = {43};
Plane Surface (44) = {44};

Plane Surface (47) = {47};   Plane Surface (48) = {48};
Plane Surface (49) = {49};   Plane Surface (50) = {50};
Plane Surface (51) = {51};

Plane Surface (54) = {54};   Plane Surface (55) = {55};
Plane Surface (56) = {56};   Plane Surface (57) = {57};
Plane Surface (58) = {58};

Plane Surface (61) = {61};   Plane Surface (62) = {62};
Plane Surface (63) = {63};

Plane Surface (64) = {64};

//Creating physical surfaces for the right tunnel
Physical Surface (1) = {1};   Physical Surface (2) = {2};
Physical Surface (3) = {3};   Physical Surface (4) = {4};
Physical Surface (5) = {5};   Physical Surface (6) = {6};

//Creating physical surfaces for the left tunnel
Physical Surface (7) = {7};       Physical Surface (8) = {8};
Physical Surface (9) = {9};       Physical Surface (10) = {10};
Physical Surface (11) = {11};    Physical Surface (12) = {12};

//Creating physical surfaces for the boundaries of the domain
Physical Surface (13) = {13};      Physical Surface (14) = {14};
Physical Surface (15) = {15};      Physical Surface (16) = {16};
Physical Surface (17) = {17};      Physical Surface (18) = {18};

//Creating physical surfaces for the foundations
Physical Surface (40) = {40};     Physical Surface (41) = {41};
Physical Surface (42) = {42};     Physical Surface (43) = {43};
Physical Surface (44) = {44};

Physical Surface (47) = {47};    Physical Surface (48) = {48};
Physical Surface (49) = {49};    Physical Surface (50) = {50};
Physical Surface (51) = {51};

Physical Surface (54) = {54};    Physical Surface (55) = {55};
Physical Surface (56) = {56};    Physical Surface (57) = {57};
Physical Surface (58) = {58};

Physical Surface (61) = {61};    Physical Surface (62) = {62};
Physical Surface (63) = {63};

Physical Surface (64) = {64};

//Creating the volume of the domain
Surface Loop (1) = {13, 14, 15, 16, 17, 18, 3, 4, 5, 6, 40, 41, 42, 43, 44, 47, 48, 49, 50, 51, 54, 55, 56, 57,
58, 61, 62, 63, 9, 10, 11, 12};
Volume (1) = {1};
Physical Volume (1) = {1};
```

A4 SINGLE TUNNEL EXCAVATION

```
/*********************************************************************
 *
```

```
 *   Gmsh file estg.txt
 *
 *   Excavation of a Single Tunnel Geometry
 *
 **********************************************************************/
//I have to take care that loop.txt file should be in the same directory with estg.txt

// Defining some parameters and characteristic lengths:

D = 4;              //Tunnel diameter
DD = 4;             //Size of the foundations
X = 17.5*D;         //Dimension of the domain along the x axis
Y = 17.5*D;         //Dimension of the domain along the y axis
Z = 12.5*D;         //Dimension of the domain along the z axis
ChL1 = 18;          //Size of the elements at the boundaries
ChL2 = 2;           //Size of the elements inside and around the tunnel
ChL3 = 9;           //Size of the transition elements
ChL4 = 3;           //Size of the elements at the foundations
x1=X/2;             //Parameter which shifts the tunnel axis horizontally
z1=Z-15;            //Parameter which shifts the tunnel axis vertically
k1=0;               //Parameter which shifts the end of the tunnel only along the z axis

//Points at the boundaries of the domain
Point (1) = {0, 0, 0, ChL1};                Point (2) = {X, 0, 0, ChL1};
Point (3) = {X, Y, 0, ChL1};                Point (4) = {0, Y, 0, ChL1};
Point (5) = {0, 0, Z, ChL1};                Point (6) = {X/2-15, 0, Z, ChL3};
Point (7) = {X/2+15, 0, Z, ChL3};           Point (8) = {X, 0, Z, ChL1};
Point (9) = {X, Y, Z, ChL1};                Point (10) = {X/2+15, Y, Z, ChL3};
Point (11) = {X/2-15, Y, Z, ChL3};          Point (12) = {0, Y, Z, ChL1};

//Points at the tunnel entrance
Point (13) = {x1, 0, z1, ChL2};             Point (14) = {x1, 0, z1+D/2, ChL2};
Point (15) = {x1-D/2, 0, z1, ChL2};         Point (16) = {x1, 0, z1-D/2, ChL2};
Point (17) = {x1+D/2, 0, z1, ChL2};

//Points at the tunnel exit
Point (18) = {x1, Y, z1+k1, ChL2};          Point (19) = {x1, Y, z1+D/2+k1, ChL2};
Point (20) = {x1-D/2, Y, z1+k1, ChL2};      Point (21) = {x1, Y, z1-D/2+k1, ChL2};
Point (22) = {x1+D/2, Y, z1+k1, ChL2};

//Points at the foundations
Point (64) = {X/2-2.5*DD, Y/2-DD/2, Z, ChL4};
Point (65) = {X/2-1.5*DD, Y/2-DD/2, Z, ChL4};
Point (66) = {X/2-0.5*DD, Y/2-DD/2, Z, ChL4};
Point (67) = {X/2+0.5*DD, Y/2-DD/2, Z, ChL4};

Point (72) = {X/2-2.5*DD, Y/2+DD/2, Z, ChL4};
Point (73) = {X/2-1.5*DD, Y/2+DD/2, Z, ChL4};
Point (74) = {X/2-0.5*DD, Y/2+DD/2, Z, ChL4};
Point (75) = {X/2+0.5*DD, Y/2+DD/2, Z, ChL4};

Point (80) = {X/2-2.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (81) = {X/2-1.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (82) = {X/2-0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};
Point (83) = {X/2+0.5*DD, Y/2+1*DD+DD/2, Z, ChL4};

Point (88) = {X/2-2.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (89) = {X/2-1.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (90) = {X/2-0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
Point (91) = {X/2+0.5*DD, Y/2+2*DD+DD/2, Z, ChL4};
```

```
//Creating the boundaries of the domain
Line (1) = {1, 2};          Line (2) = {2, 3};
Line (3) = {3, 4};          Line (4) = {4, 1};
Line (5) = {5, 6};          Line (6) = {6, 7};
Line (7) = {7, 8};          Line (8) = {8, 9};
Line (9) = {9, 10};         Line (10) = {10, 11};
Line (11) = {11, 12};       Line (12) = {12, 5};
Line (13) = {1, 5};         Line (14) = {2, 8};
Line (15) = {3, 9};         Line (16) = {4, 12};

//Creating the entrance opening
Circle (17) = {14, 13, 15};      Circle (18) = {15, 13, 16};
Circle (19) = {16, 13, 17};      Circle (20) = {17, 13, 14};

//Creating the exit opening
Circle (21) = {19, 18, 20};      Circle (22) = {20, 18, 21};
Circle (23) = {21, 18, 22};       Circle (24) = {22, 18, 19};

//Creating the foundations
Line (63) = {64, 65};   Line (64) = {65, 66};
Line (65) = {66, 67};

Line (70) = {72, 73};   Line (71) = {73, 74};
Line (72) = {74, 75};

Line (77) = {80, 81};   Line (78) = {81, 82};
Line (79) = {82, 83};

Line (84) = {88, 89};   Line (85) = {89, 90};
Line (86) = {90, 91};

Line (96) = {88, 80};   Line (97) = {80, 72};
Line (98) = {72, 64};

Line (102) = {89, 81};   Line (103) = {81, 73};
Line (104) = {73, 65};

Line (108) = {90, 82};   Line (109) = {82, 74};
Line (110) = {74, 66};

Line (114) = {91, 83};   Line (115) = {83, 75};
Line (116) = {75, 67};

//Creating loops for the tunnel
Line Loop (1) = {17, 18, 19, 20};      //Plane Surface (1) = {1};
Line Loop (2) = {21, 22, 23, 24};      //Plane Surface (2) = {2};

Line Loop (3) = {-13, 1, 14, -7, -6, -5}; Line Loop (4) = {-14, 2, 15, -8};
Line Loop (5) = {-15, 3, 16, -11, -10, -9}; Line Loop (6) = {-16, 4, 13, -12};
Line Loop (7) = {1, 2, 3, 4};  Line Loop (8) = {-11, -10, -9, -8, -7, -6, -5, -12};

//Creating loops for the foundations
Line Loop (41) = {63, -104, -70, 98};      Line Loop (42) = {64, -110, -71, 104};
Line Loop (43) = {65, -116, -72, 110};

Line Loop (48) = {70, -103, -77, 97};      Line Loop (49) = {71, -109, -78, 103};
Line Loop (50) = {72, -115, -79, 109};

Line Loop (55) = {77, -102, -84, 96};      Line Loop (56) = {78, -108, -85, 102};
```

```
Line Loop (57) = {79, -114, -86, 108};

Line Loop (61) = {63, 64, 65, -116, -115, -114, -86, -85, -84, 96, 97, 98};

//Creating surfaces for the boundaries of the domain
Plane Surface (3) = {3, 1};     Plane Surface (4) = {4};
Plane Surface (5) = {5, 2};     Plane Surface (6) = {6};
Plane Surface (7) = {7};        Plane Surface (8) = {8, 61};

//Creating surfaces for the foundations
Plane Surface (41) = {41};   Plane Surface (42) = {42};
Plane Surface (43) = {43};

Plane Surface (48) = {48};   Plane Surface (49) = {49};
Plane Surface (50) = {50};

Plane Surface (55) = {55};   Plane Surface (56) = {56};
Plane Surface (57) = {57};

Plane Surface (61) = {61};

Include "loop.txt";   //loop.txt file should be in the same directory with the current file

//Creating physical surfaces for the boundaries of the domain
Physical Surface (1) = {3};   Physical Surface (2) = {4};
Physical Surface (3) = {5};   Physical Surface (4) = {6};
Physical Surface (5) = {7};   Physical Surface (6) = {8};

//Creating physical surfaces for the foundations
Physical Surface (7) = {41};   Physical Surface (8) = {42};
Physical Surface (9) = {43};

Physical Surface (10) = {48};   Physical Surface (11) = {49};
Physical Surface (12) = {50};

Physical Surface (13) = {55};   Physical Surface (14) = {56};
Physical Surface (15) = {57};

Physical Surface (16) = {61};

//Creating the volume of the domain
Surface Loop (3000) = {3, 4, 5, 6, 7, 8, 41, 42, 43, 48, 49, 50, 55, 56, 57, 132, 134, 136, 138, 176, 178,
180, 182, 220, 222, 224, 226, 264, 266, 268, 270, 308, 310, 312, 314, 352, 354, 356, 358, 396, 398, 400,
402, 440, 442, 444, 446, 484, 486, 488, 490, 528, 530, 532, 534, 572, 574, 576, 578, 616, 618, 620, 622,
660, 662, 664, 666, 704, 706, 708, 710};
Volume (100) = {3000};
Physical Volume (1) = {100};

/*********************************************************************
 *
 *   Gmsh file loop.txt
 *
 *   Division of the tunnel into smaller volumes
 *   Add this file to the same working directory with the estg.txt file
 *********************************************************************/
ChL2 = 2;          //Size of the elements in and around the tunnel
Ur = 5;            //Unsupported length of excavation
D = 4;             //Tunnel diameter
X = 17.5*D;        //Dimension of the domain along the x axis
Y = -Ur;           //Dimension of the domain along the y axis
```

```
Z = 12.5*D;        //Dimension of the domain along the z axis
x1=X/2;            //Parameter which shifts the tunnel axis horizontally
z1=Z-15;           //Parameter which shifts the tunnel axis vertically

//Create a Function
Function CirclePlanes

//Points at the tunnel entrance
p1 = newp; Point (p1) = {x1, Y, z1, ChL2};          p2 = newp; Point (p2) = {x1, Y, z1 + D/2, ChL2};
p3 = newp; Point (p3) = {x1 - D/2, Y, z1, ChL2};    p4 = newp; Point (p4) = {x1, Y, z1 - D/2, ChL2};
p5 = newp; Point (p5) = {x1 + D/2, Y, z1, ChL2};

//Points at the exit of the first volume
p6 = newp; Point (p6) = {x1, Y+Ur, z1, ChL2};       p7 = newp; Point (p7) = {x1, Y+Ur, z1+D/2, ChL2};
p8 = newp; Point (p8) = {x1 - D/2, Y+Ur, z1, ChL2};  p9 = newp; Point (p9) = {x1, Y+Ur, z1-D/2,
ChL2};
p10 = newp; Point (p10) = {x1 + D/2, Y+Ur, z1, ChL2};

//Creating the entrance opening
c1 = newreg; Circle (c1) = {p2, p1, p3};   c2 = newreg; Circle (c2) = {p3, p1, p4};
c3 = newreg; Circle (c3) = {p4, p1, p5};   c4 = newreg; Circle (c4) = {p5, p1, p2};

//Creating the exit opening of the first volume
c5 = newreg; Circle (c5) = {p7, p6, p8};     c6 = newreg; Circle (c6) = {p8, p6, p9};
c7 = newreg; Circle (c7) = {p9, p6, p10};   c8 = newreg; Circle (c8) = {p10, p6, p7};

//Connecting the opening and the exit of the first volume
l1 = newreg; Line (l1) = {p2, p7};      l2 = newreg; Line (l2) = {p3, p8};
l3 = newreg; Line (l3) = {p4, p9};      l4 = newreg; Line (l4) = {p5, p10};

//Creating loops and surfaces for the first volume
ll1 = newreg; Line Loop (ll1) = {c1, c2, c3, c4};   ps1 = news; Plane Surface (ps1) = {ll1};

ll3 = newreg; Line Loop (ll3) = {l1, c5, -l2, -c1};
rs1 = newreg; Ruled Surface (rs1) = {ll3}; Printf("Ruled surface rs1= %g " , rs1) ;
ll4 = newreg; Line Loop (ll4) = {l2, c6, -l3, -c2};   rs2 = newreg; Ruled Surface (rs2) = {ll4};
ll5 = newreg; Line Loop (ll5) = {l3, c7, -l4, -c3};   rs3 = newreg; Ruled Surface (rs3) = {ll5};
ll6 = newreg; Line Loop (ll6) = {l4, c8, -l1, -c4};   rs4 = newreg; Ruled Surface (rs4) = {ll6};

Extrude  Surface { ps1, { 0, Ur, 0 }};

Return                    //End of the Function

For (1:14)                //Generate the loop
Y += Ur ;
Call CirclePlanes ;
EndFor
```

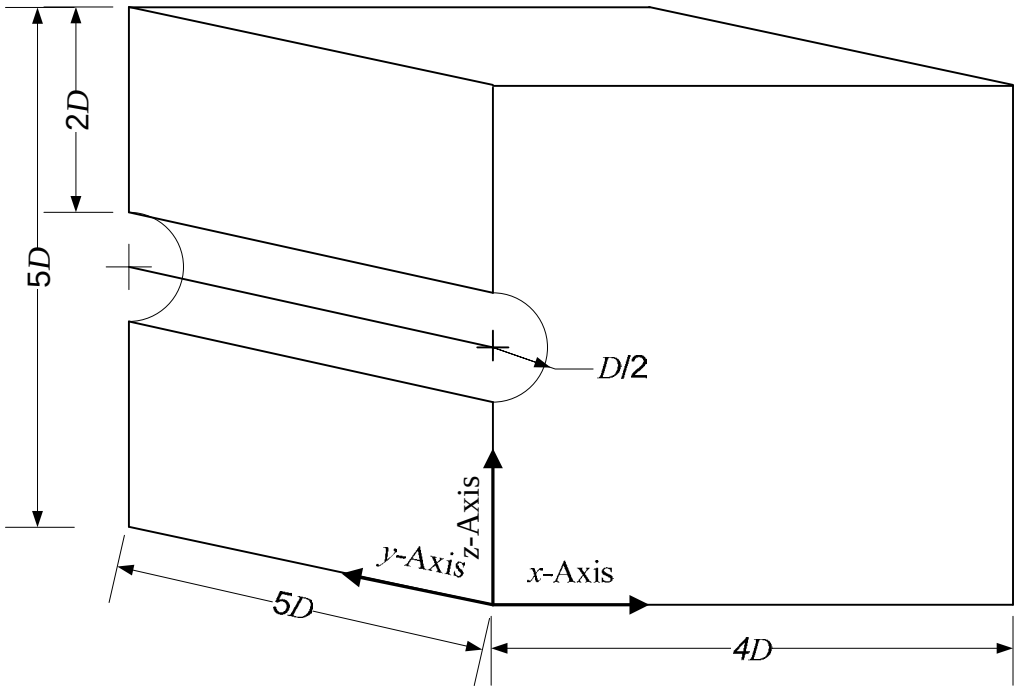
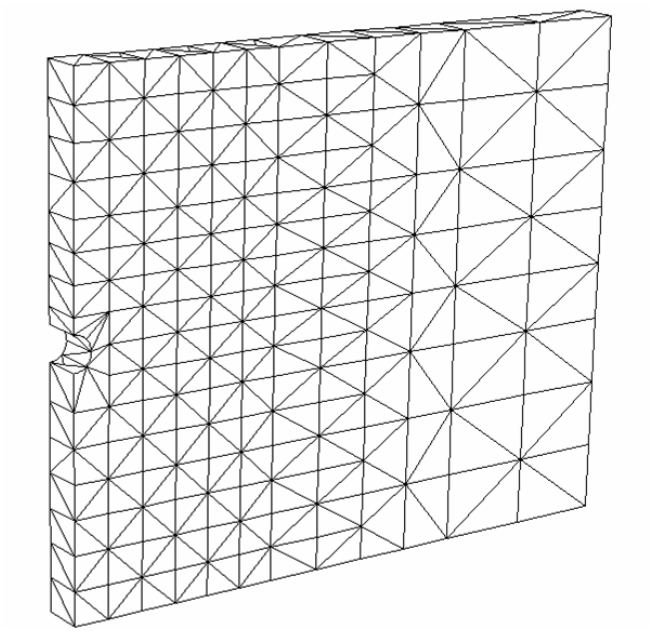
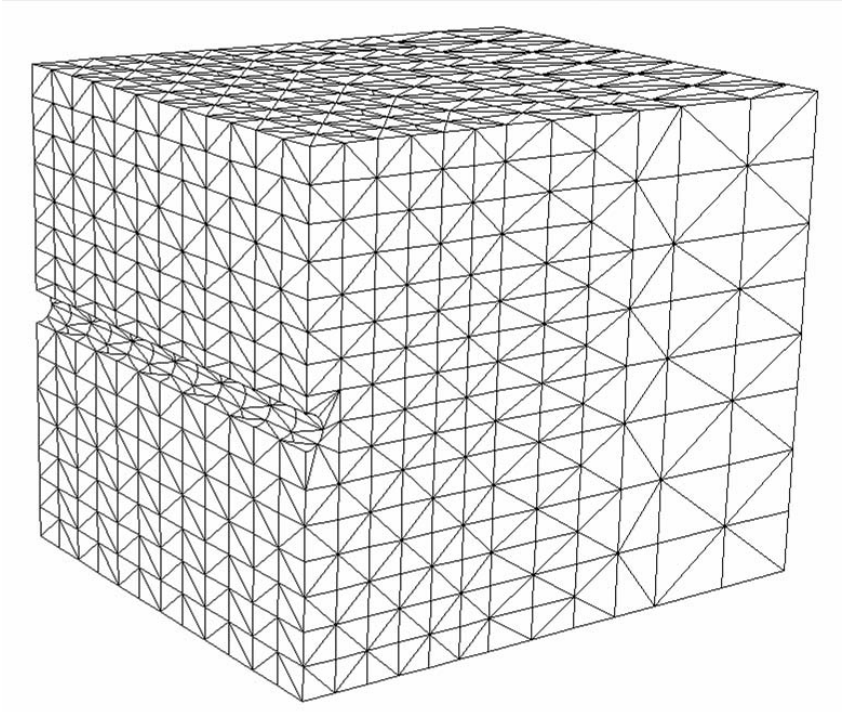


Figure 1. 3D FE analysis of a single horizontal tunnel.

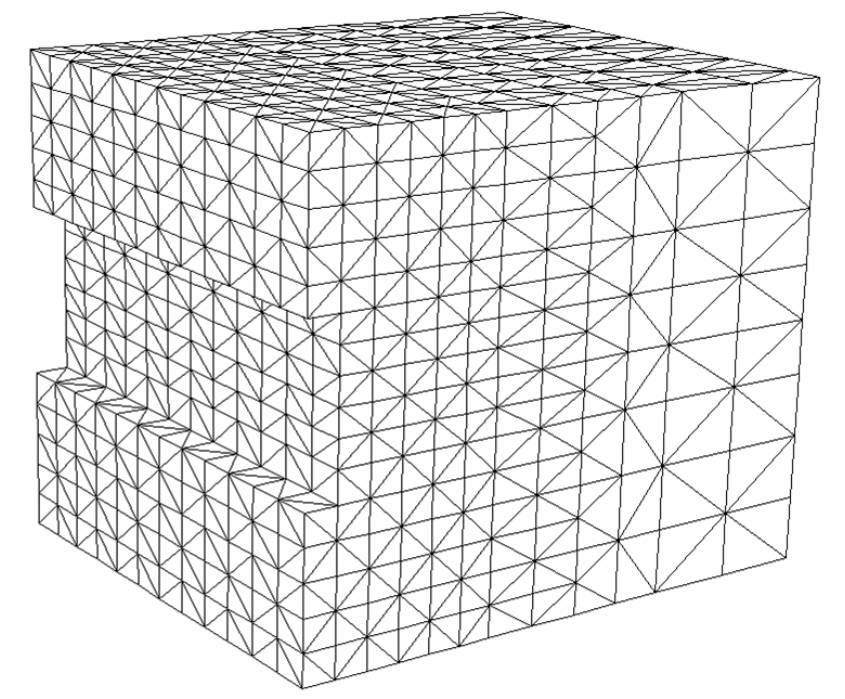


a) Slice of the domain

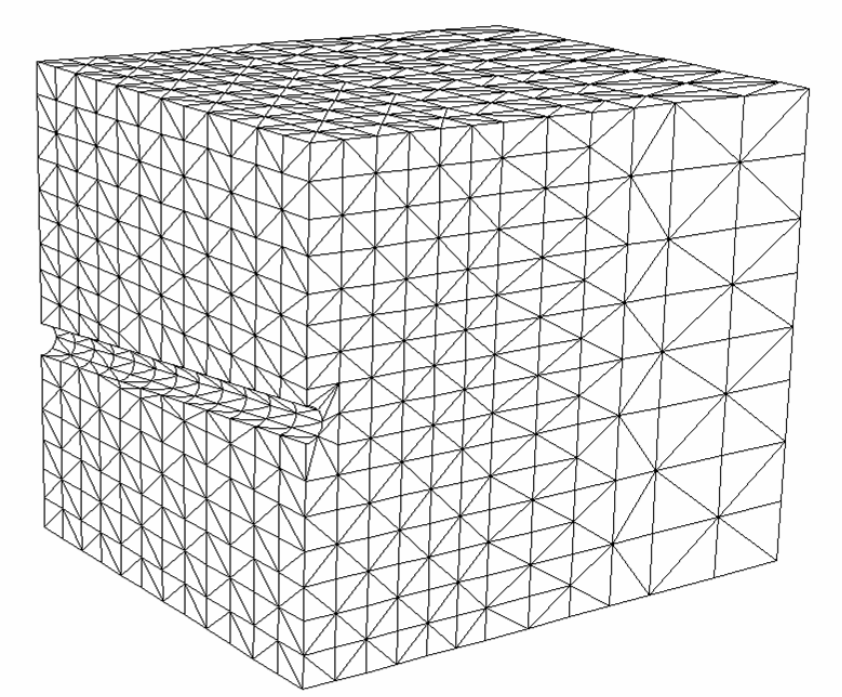


b) Whole domain

Figure 2. Meshing of a single horizontal tunnel axis in soft ground using a commercial FE mesh generator.



a) Partly generated mesh



b) Meshing sub-section

Figure 3. Meshing of an inclined tunnel axis geometry.

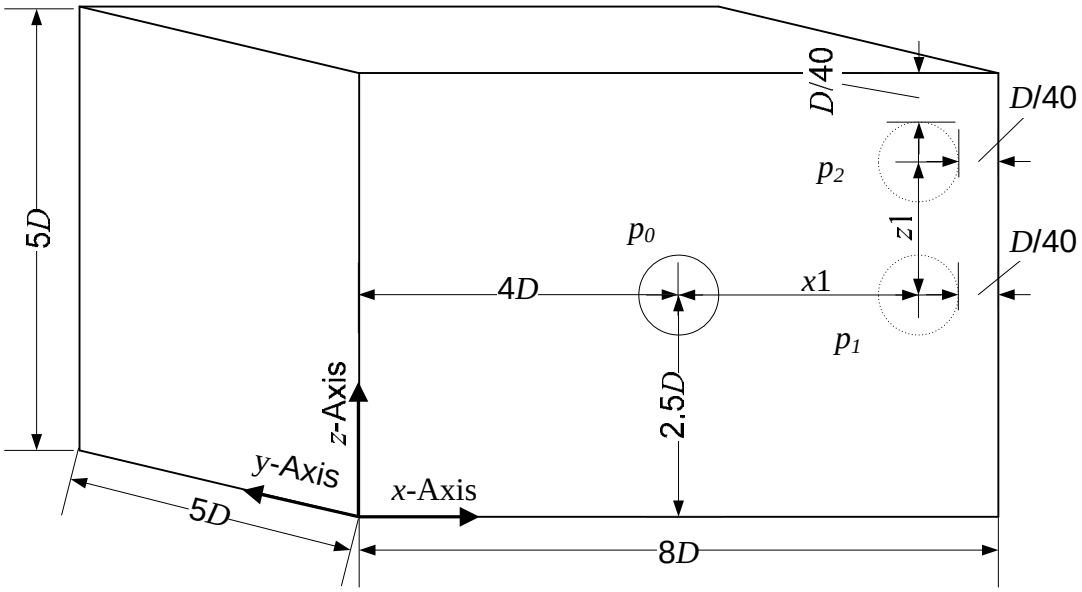


Figure 4. Three different tunnel positions (p_0, p_1 and p_2) of a horizontal tunnel.

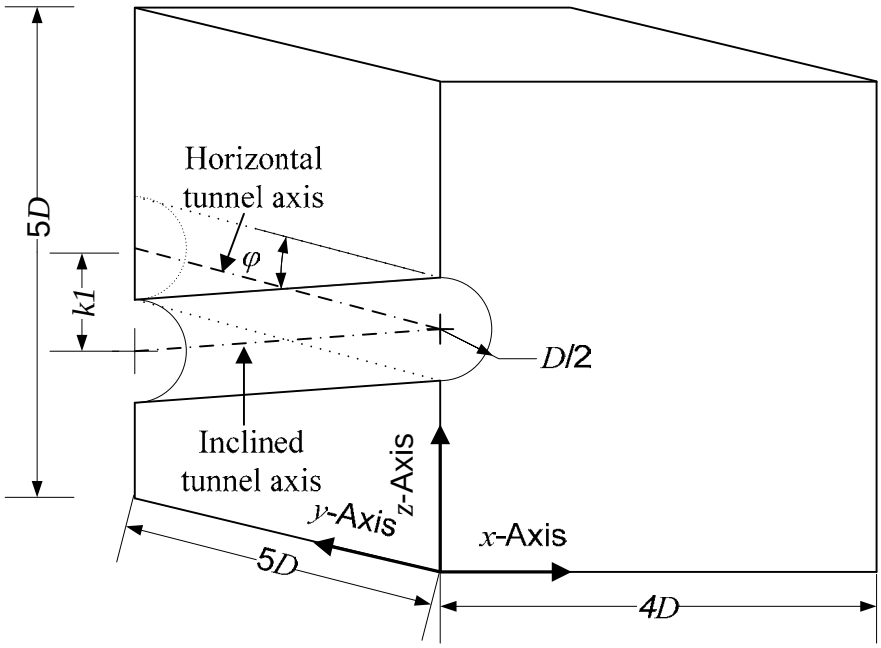


Figure 5. Inclined tunnel geometry.

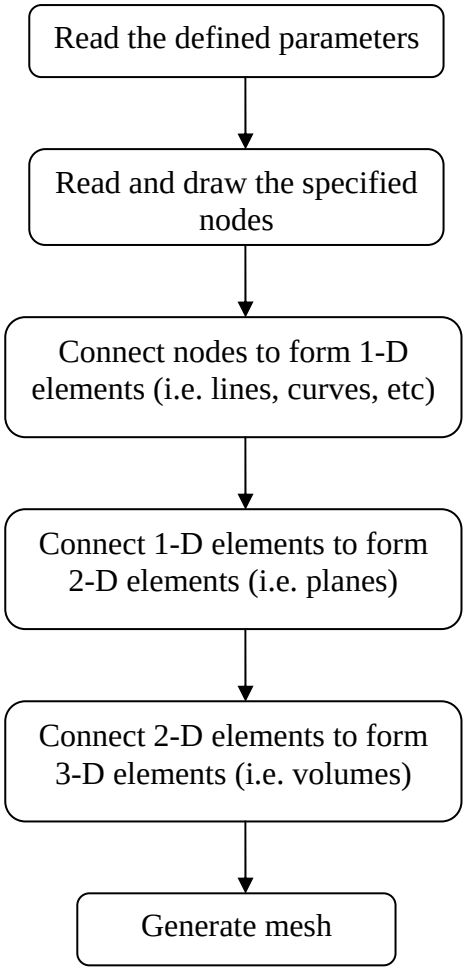
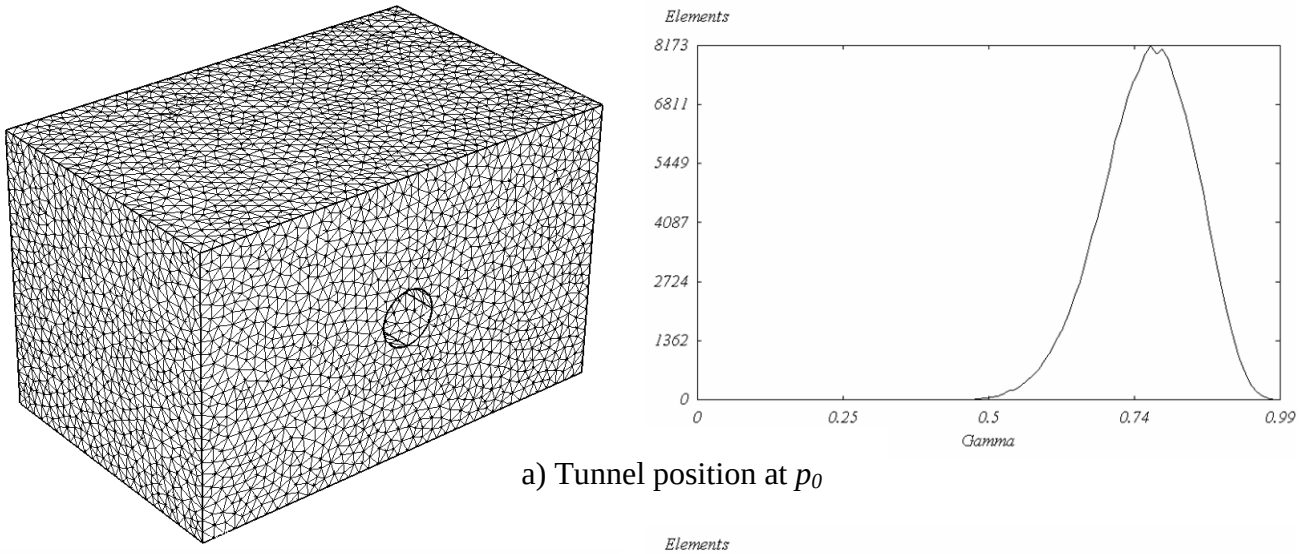
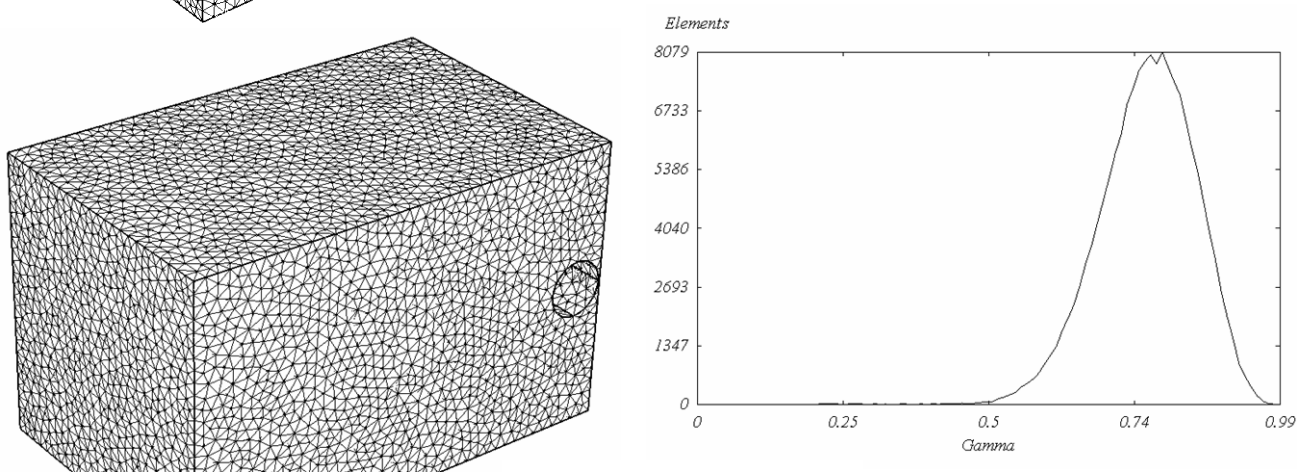


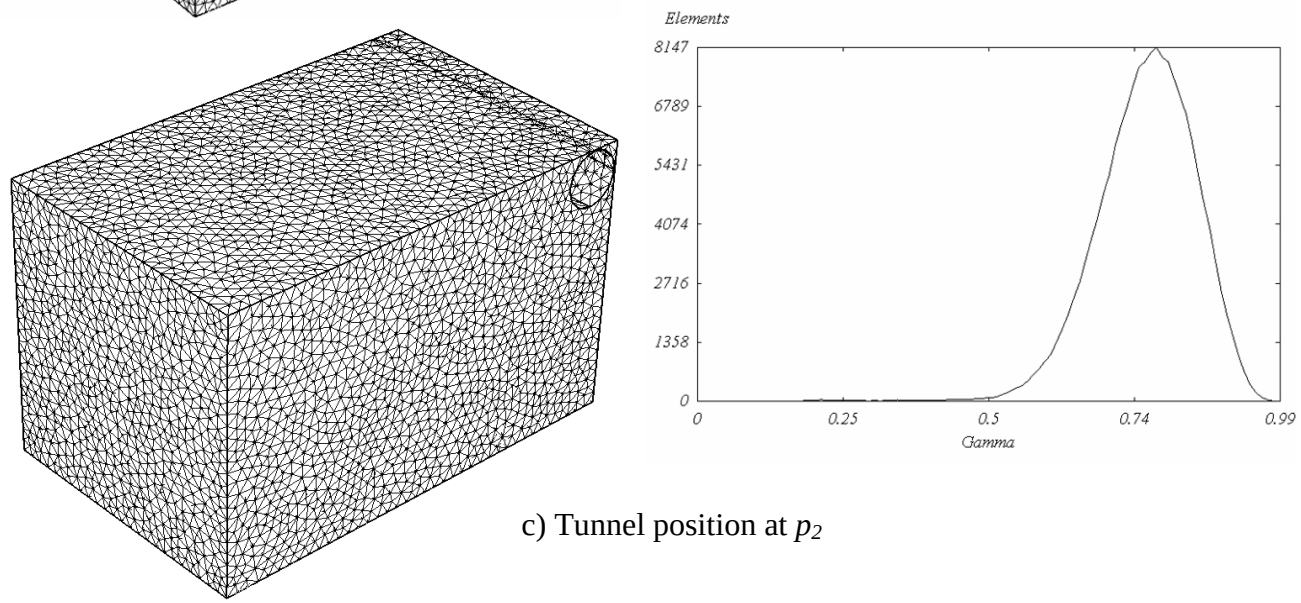
Figure 6. Flowchart of the main operations executed by Gmsh in order to generate a mesh.



a) Tunnel position at p_0

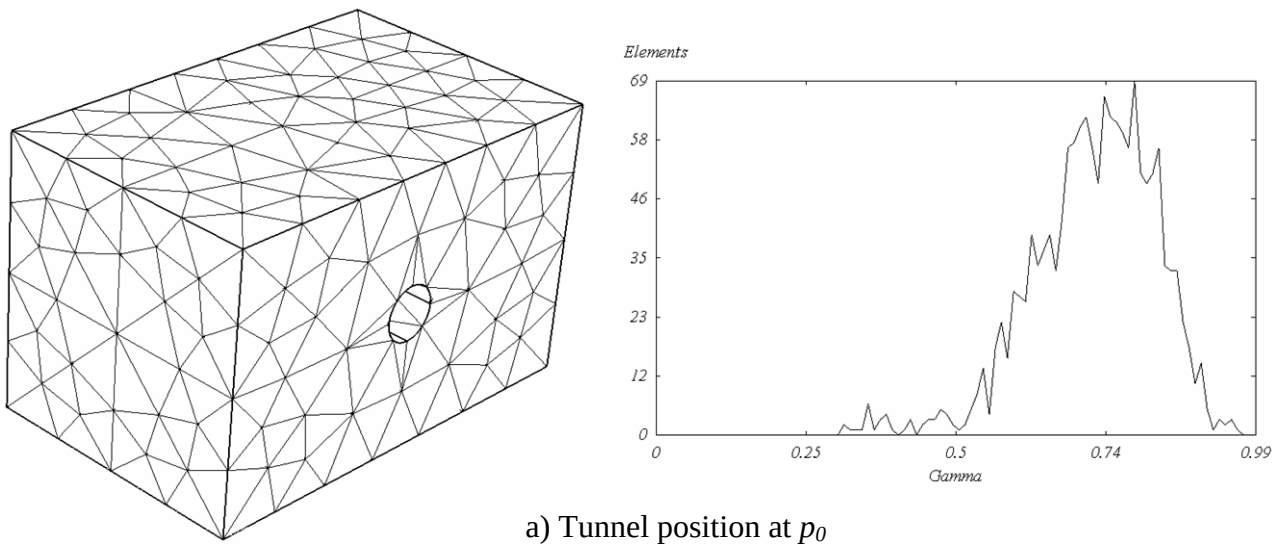


b) Tunnel position at p_1

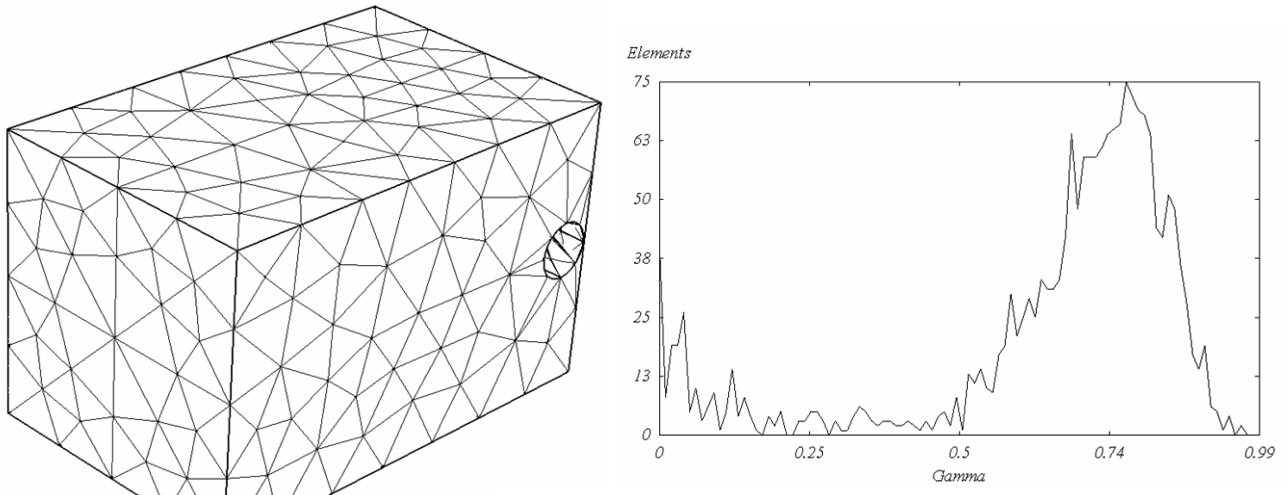


c) Tunnel position at p_2

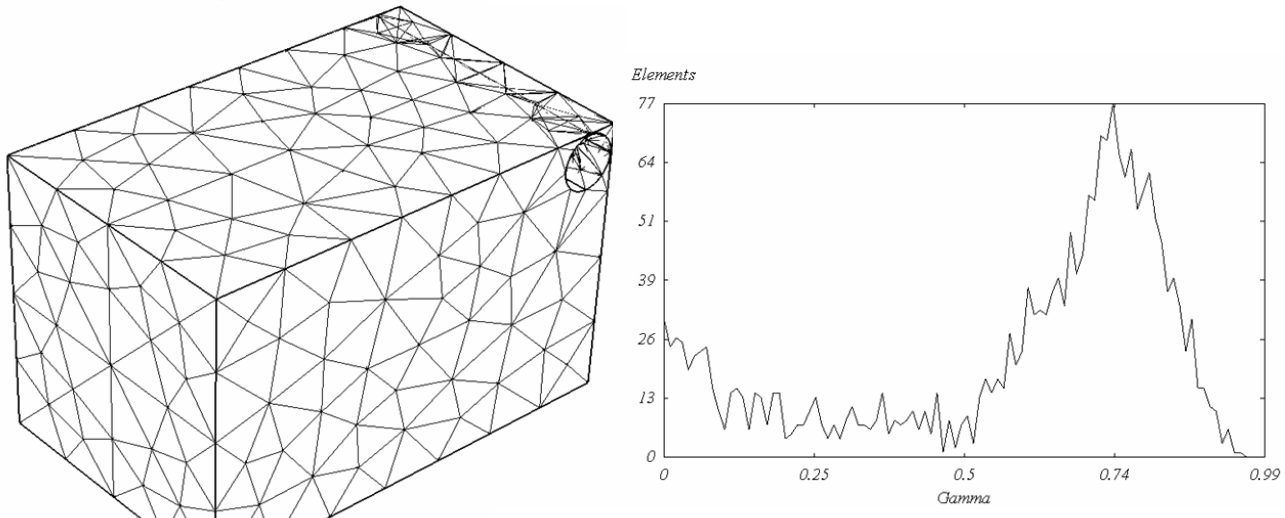
Figure 7. Mesh quality measurements of a very fine mesh (when *ChL1* = *D*/5) for three different tunnel positions (p_0, p_1 and p_2).



a) Tunnel position at p_0

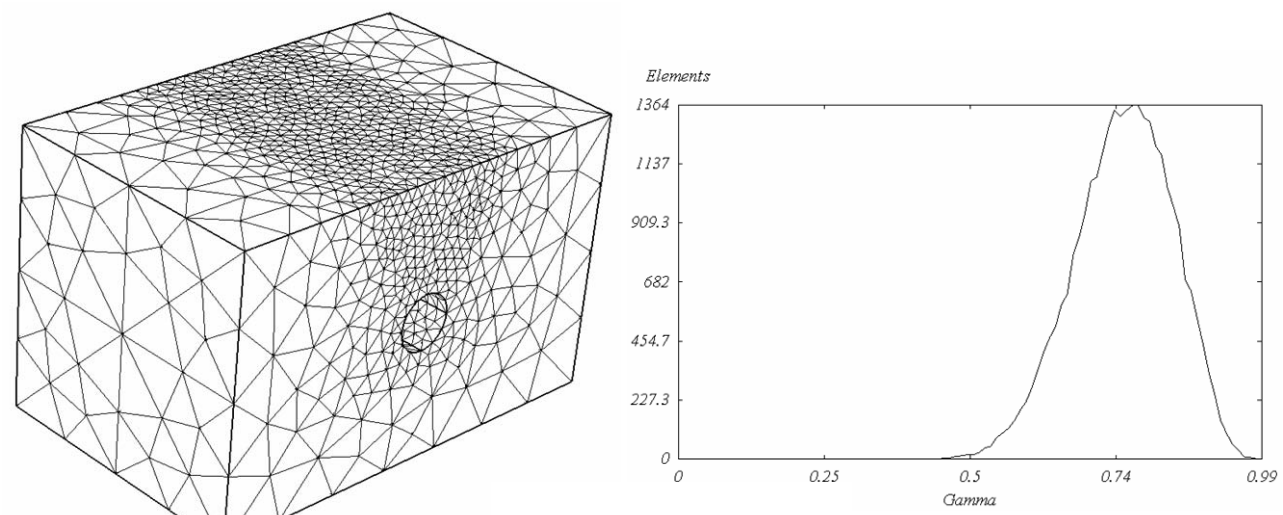


b) Tunnel position at p_1

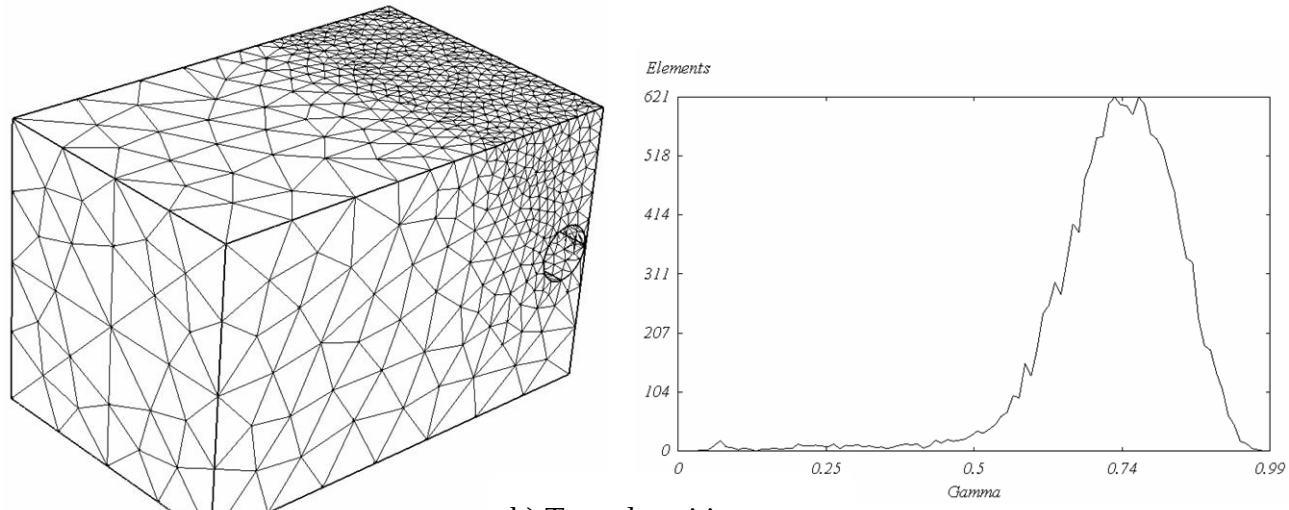


c) Tunnel position at p_2

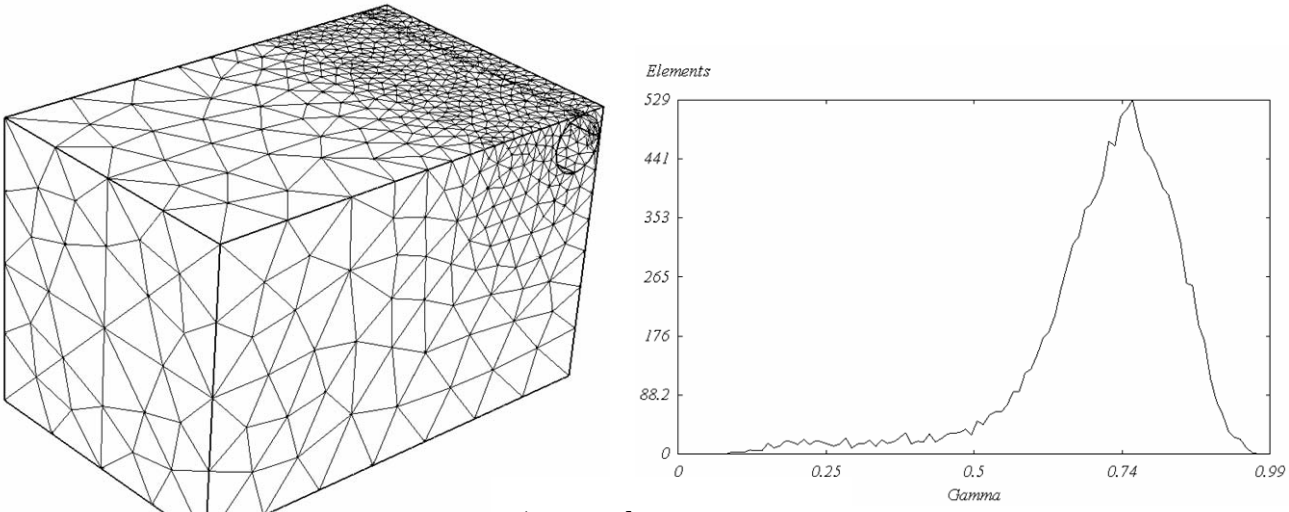
Figure 8. Mesh quality measurements of a coarse mesh (when *ChL1 = D*) for three different tunnel positions (p_0, p_1 and p_2).



a) Tunnel position at p_0



b) Tunnel position at p_1



c) Tunnel position at p_2

Figure 9. Mesh quality measurements of a refined mesh for three different tunnel positions.

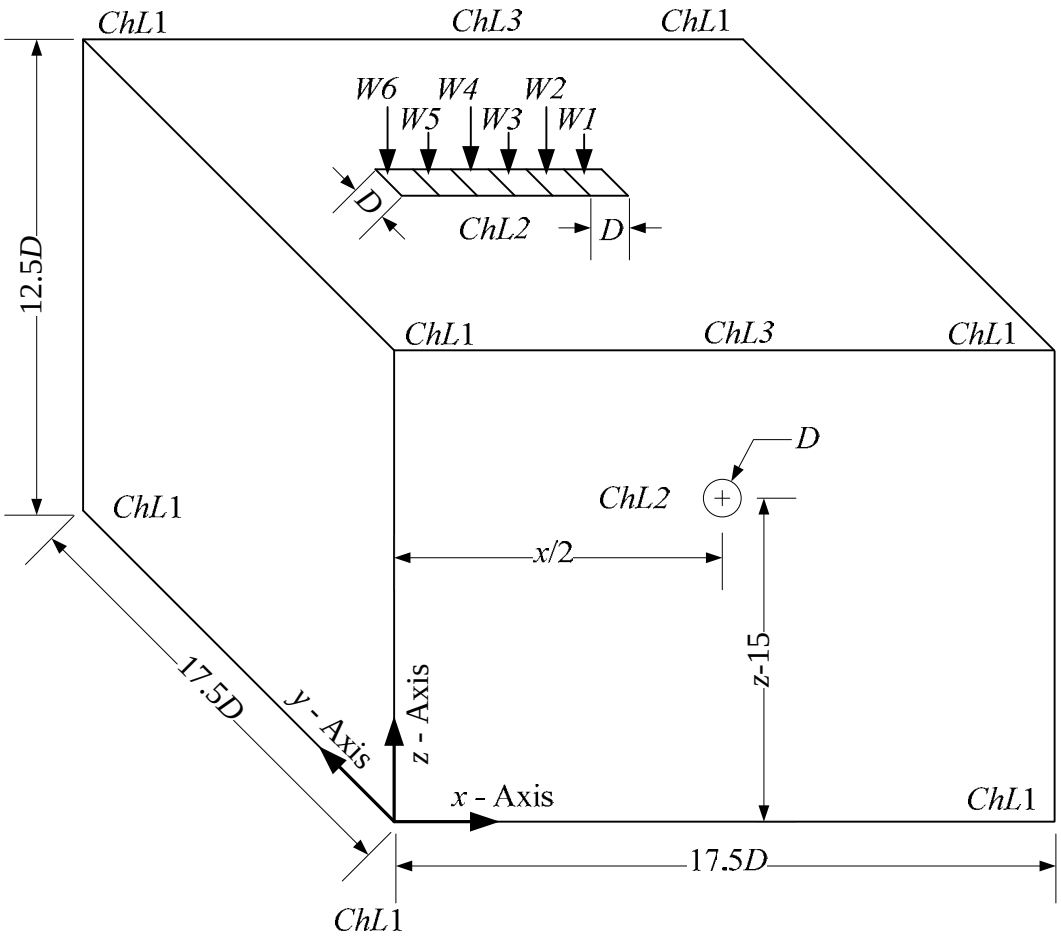
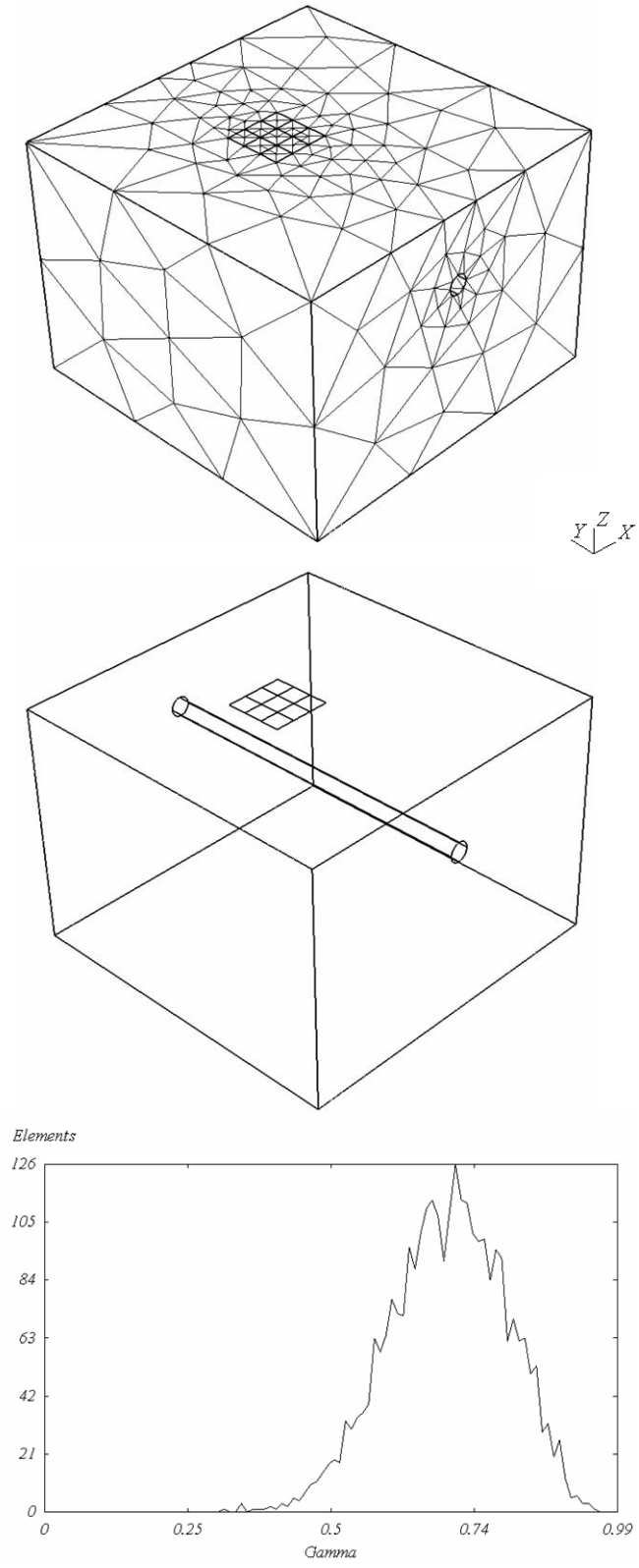
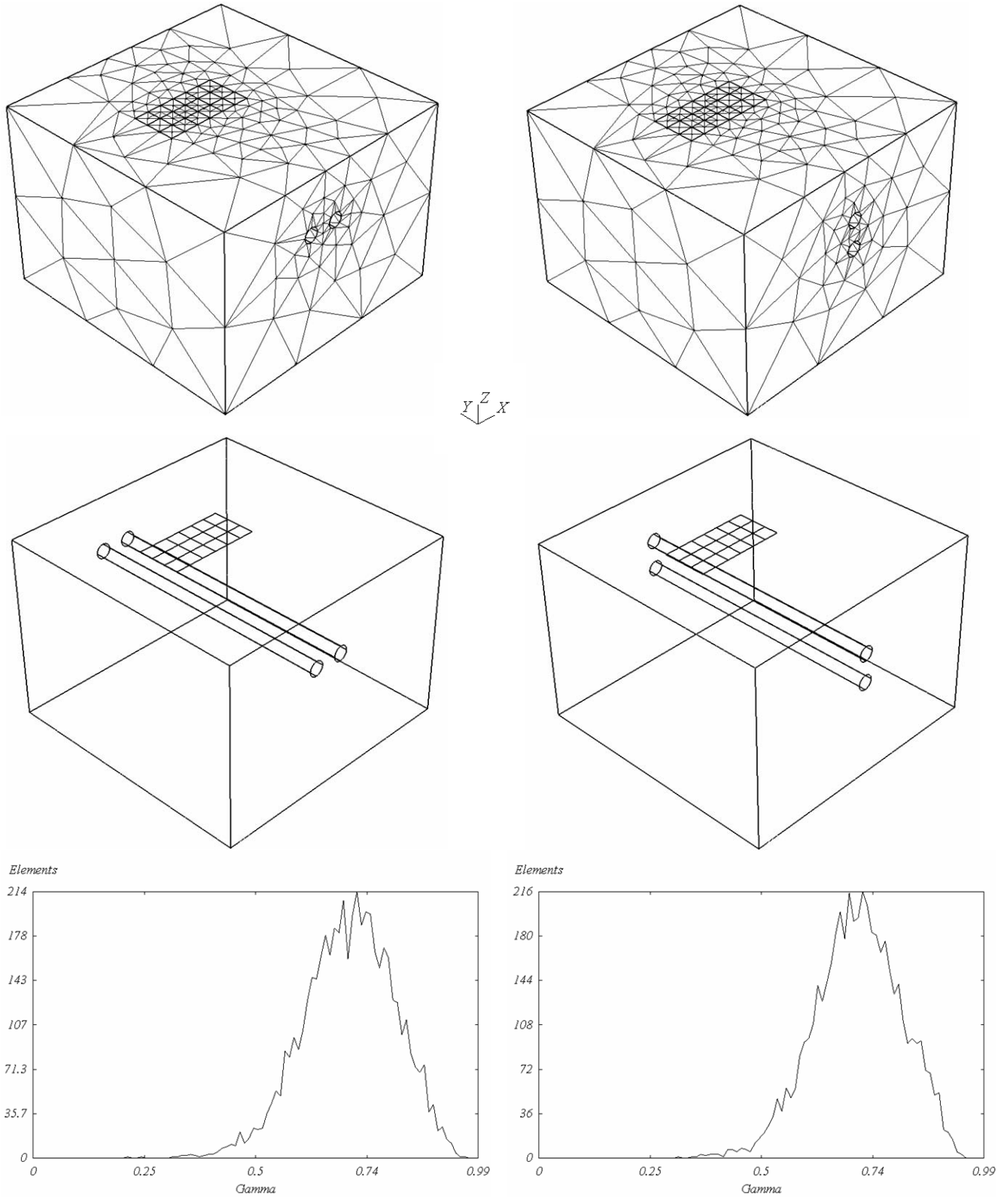


Figure 10. Dimensions of the domain with surface loading.



Single horizontal at z_0 = 15m

Figure 11. Single horizontal tunnel case.



a) Twin tunnels horizontally aligned b) Twin tunnels vertically aligned

Figure 12. Twin tunnel geometry for the a) *TH* and b) *TV* cases.

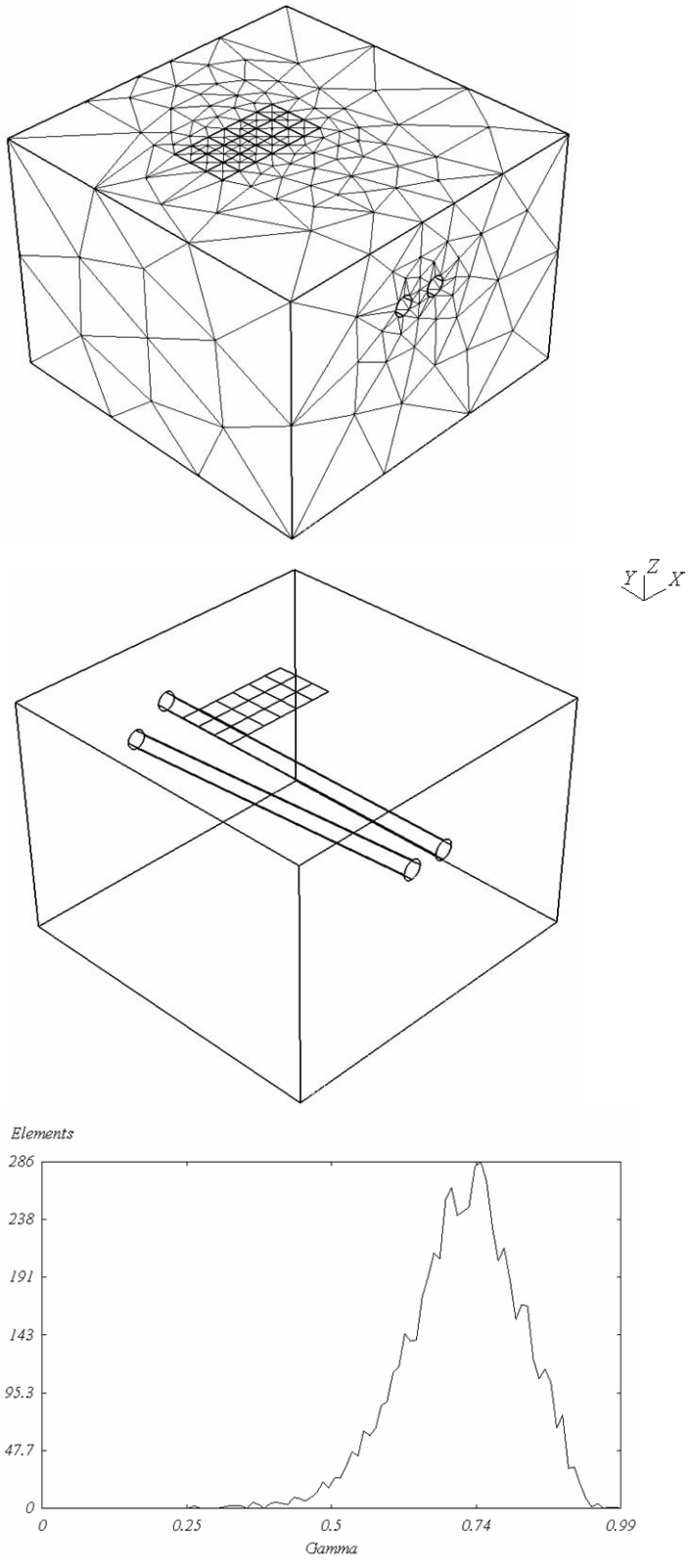


Figure 13. Twin tunnel geometry where the left tunnel is inclined (the inclination angle is 4^{o}) while the right is horizontally aligned (*MHI* case).

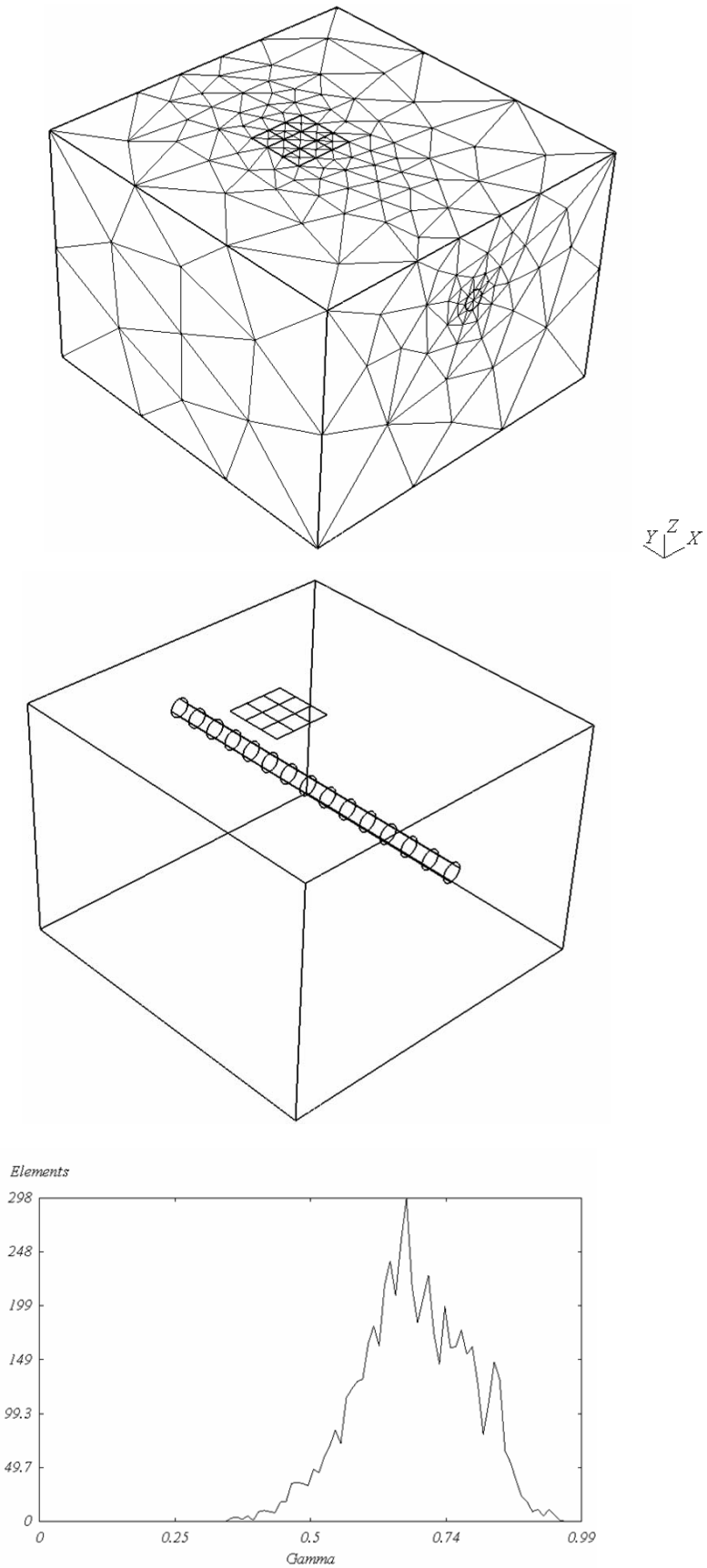


Figure 14. Mesh for the excavation of a single horizontal tunnel.

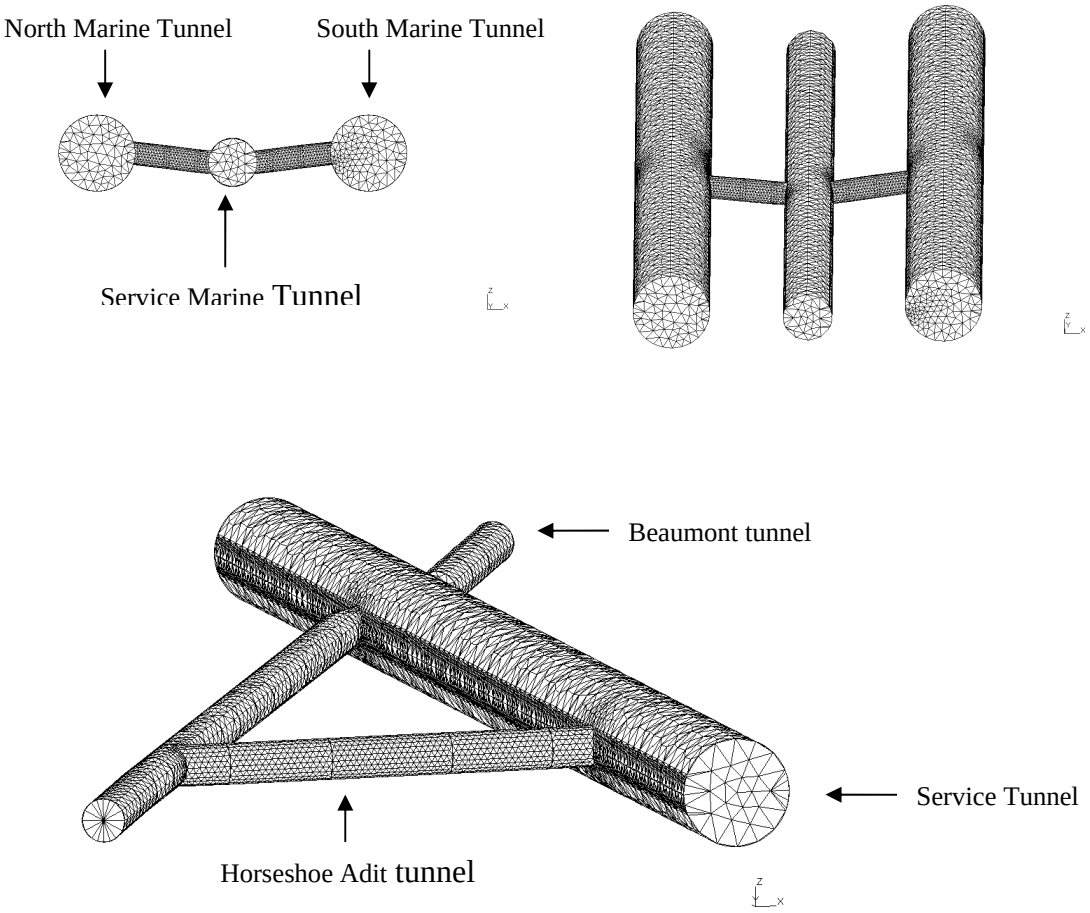


Figure 15. Case study from the Channel Tunnel Project.