Joly, Antoine; Leroy, Agnés; Wang, Pengze

Towards the simulation of submerged bottom structures with vertical walls using Telemac-3D

Towards the simulation of submerged bottom structures with vertical walls using Telemac-3D

Pengze Wang
EDF R&D LNHE / Polytech Nice-Sophia / Mines ParisTech
6 quai Watier, 78401 Chatou, France

Antoine Joly & Agnès Leroy
EDF R&D LNHE / LHSV (Ecole des Ponts, CEREMA, EDF R&D, UPE)
6 quai Watier, 78401 Chatou, France
Email: antoine.joly@edf.fr

***Abstract*—In the Telemac-Mascaret community, there is an increasing need for Telemac-3D to handle unstructured 3D meshes. One of the main target applications is the simulation of submerged structures in the flow. With this aim in view, a first step is to modify the code so that it can account for vertical structures on the bed. This is done while maintaining the advantages of the layered 3D meshes currently used in Telemac-3D, since a full destructuration would imply more complex neighbour searches, data access and would require major changes in the algorithms. The proposed approach will be presented in this article, including a description of the necessary developments, comparisons of CPU time before and after the changes, and a qualitative test case.**

I. Introduction

Telemac-3D is quite a powerful tool for 3D environmental simulations. One of its strengths lays in the data structure within the code: the 3D meshes are built as a layering of one 2D mesh. This makes the data access and neighbour search efficient, and simplifies the resolution of the 3D Navier–Stokes equations. However, it comes with drawbacks since Telemac-3D is then unable to handle submerged bodies and bathymetries presenting vertical sections.

The aim of this work is to allow Telemac-3D to model submerged structures with vertical walls and a flat top when they lie on the bed; for example steps or submerged cylinders. A simplified representation of the kind of problems considered here is provided in Figure 1.

The chosen methodology is to keep the main geometrical advantage of Telemac-3D, namely the layered elements. This data structure makes the memory access efficient and facilitates all the operations that need to be performed along the vertical axis (such as the integration along the depth). This means that in the proposed formulation, the mesh for these problems will still be defined from an extrusion of a two-dimensional mesh. However, in the mesh some nodes and element faces will represent the sides of a vertical structure and all the nodes and elements inside that structure will be ignored.

The following modifications to Telemac-3D are thus proposed:

- all the loops shall be modified in order to account for a variable number of nodes or elements along the vertical axis;
- a new geometrical file will be read to define the elements and nodes belonging to a vertical structure;
- the boundary conditions on the sides of the vertical structures will be added to the boundary conditions file;
- wall boundary conditions will be imposed on the sides and top of the vertical structures;
- a new variable for the visualisation of these vertical structures will be added to the output.

Each of these modifications will be described in detail in section II. Comparisons of CPU time before and after the modifications will be shown for a test-case that does not involve vertical structures in section III. Finally, an illustration of the kind of problems that the code can handle after the modifications will be shown in section IV.

Finally, it should be noted that the work presented here is still in progress, and is not ready to be integrated into an official release of Telemac-3D yet.

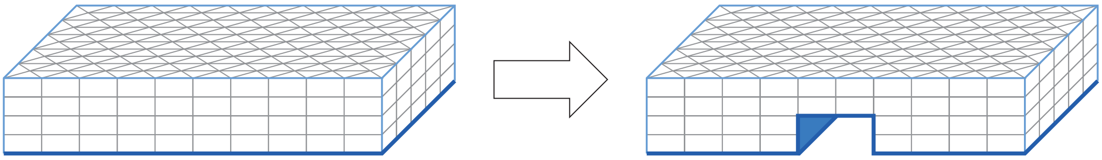
II. Proposed developments

A. *Modification of the three-dimensional loops*

The first design choice was to have a variable number of planes per node or element in the 2D mesh. To do so, the bed plane number is made variable, and loops through the mesh points or elements start from its value. Figure 2 shows a sketch of the definition of the bed nodes and elements. In this way, all the arrays are still stored with the same structure as before. For example, an array stored at the nodes of a mesh with a number of 2D points NPOIN2 and containing NPLAN planes is stored as a succession of memory blocks of size NPOIN2, each corresponding to a plane of the 3D mesh, NPLAN times. The loops through the mesh points or elements will only jump a few positions in the arrays, so as not to access anything inside the vertical structures. This approach should only have a minimal influence on cases without vertical structures (see the section III for the numerical tests).

The bed nodes will be named `STARTPP` and the bed elements `STARTPE` in the code. For example, a loop over all the nodes of the mesh is now written as:

```
DO IPOIN2=1,NPOIN2 !Loop on all 2D nodes
  !Loop on all planes in the domain
  DO IPLAN=STARTPP(IPOIN2),NPLAN
    !Index of the 3D node
    IPOIN3=(IPLAN-1)*NPOIN2+IPOIN2
    <...>
  ENDDO
ENDDO
```



(a) Classic layered mesh in Telemac-3D (b) Mesh with a vertical structure

Fig. 1: Illustration of a problem with a vertical structure.

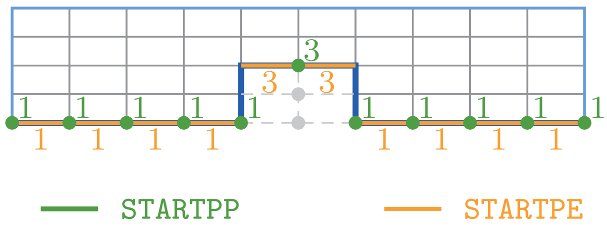


Fig. 2: Sketch of the bed plane number definition for the nodes (STARTPP) and elements (STARTPE), when modelling vertical structures. The grey circles represent the ignored nodes whereas the dashed line represents the ignored elements.

B. New formatted data file for the vertical structures

Currently, Telemac-3D can only read a two-dimensional mesh, which is then extruded along the vertical to form the three-dimensional mesh. To limit the modifications to the code, and to keep the possibility for the user to define the vertical discretisation, this has been kept. The vertical structures will be defined in a new formatted data file, where the height of each structure will be defined as well as the list of points and elements of the 2D mesh lying inside each structure. This implies that in the two-dimensional mesh provided to Telemac-3D, the sides of the vertical structures must correspond to triangle edges and the inside of the vertical structures must be meshed An example of such a mesh can be found in figure 3.

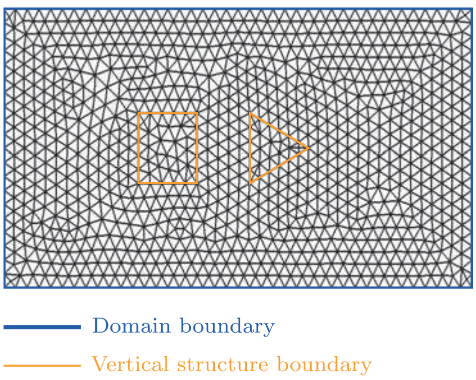


Fig. 3: Two-dimensional mesh for a test case involving two vertical structures. The meshed boundaries of the vertical structures are highlighted in orange.

The vertical structures file is referenced in the steering file using the following keywords:

```
/---------------------------------------------------/
/                    OPTIONS FOR VERTICAL STRUCTURES
/---------------------------------------------------/
VERTICAL STRUCTURES ON THE BED = YES
FILE FOR VERTICAL STRUCTURES = <File name>
```

This file then needs to follow the following format for each vertical structure:

```
<NFR> <NPOIN_VST> <NELEM_VST> <ZVST>
<List of the boundary nodes>
<List of all the nodes inside and on the boundary>
<List of all the elements inside>
```

Where NFR is the number of nodes on the boundary of the vertical structure, NPOIN_VST is the number of points along the boundary and inside the vertical structure, NELEM_VST is the number of elements inside the vertical structure and ZVST is the z-coordinate of the top of the step. This file can be easily defined using Salome_Hydro, see Wang [1] for the complete methodology.

Furthermore, it should be noted that the plane of the mesh closest to the value ZVST will be moved to this value and it will be fixed.

C. Definінition of the vertical structures' side boundary conditions

The side boundary conditions of the vertical structures will be added to the original boundary conditions file. This has only been tested for meshes in MED format and their corresponding boundary condition files. As a reminder, the BOUNDARY CONDITIONS FILE for a mesh in MED format is structured as follows:

```
<Number of boundaries>
<Boundary type> <name of group in mesh>
...
```

Adding the boundary nodes of the vertical structures to the boundary nodes of the domain is not trivial as this means that some of the boundary nodes are now inside the two-dimensional domain. This is an issue, since in the Bief the detection of the boundary segments is done by checking if the triangles' faces are missing a neighbour. Such process does not work when boundary segments are located inside the mesh. Therefore, the algorithm for the boundary segments detection must be modified, as well as for the detection of boundary triangles. The existing formulation for boundary segments detection was kept, but an additional check was added: segments are also detected as belonging to the boundary when they belong to a triangle lying in a vertical structure. The triangles that are connected to these segments, but that are outside of a vertical structure, are then added to the list of boundary elements.

Finally, the neighbouring list `IFABOR` of each element in three-dimension had to be updated. Indeed, a neighbour search through the face of an element whose neighbour lies within a vertical structure should yield either a liquid or solid boundary condition instead of the index of the element, so that the latter is ignored.

D. Imposition of the vertical structures' side boundary conditions

On vertical structures, the imposition of boundary conditions must only be done from the bed to the top of the structure. The loops along the three-dimensional boundary nodes or elements must then start from the first plane and stop at the plane colinear to the top of the vertical structure. To do so, a new variable `ENDCLI` was defined. Figure 4 shows the values ENDCLI takes on classical boundaries and on vertical structures.

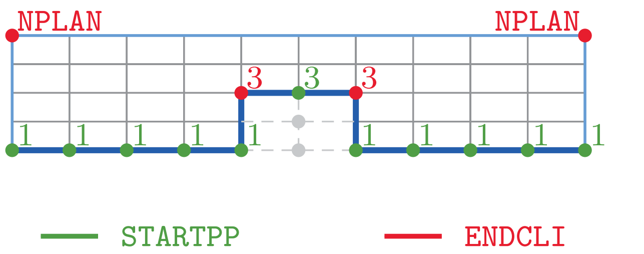


Fig. 4: Sketch of the starting and ending plane number for lateral boundary nodes of vertical structures (`STARTPP` and `ENDCLI` respectively). The grey circles represent the ignored nodes whereas the dashed line the ignored elements.

Therefore, loops along the boundaries are now written as:

```
DO IPTFR2=1,NPTFR2 !Loop on boundary nodes
  IPOIN2=NBOR2(IPTFR2)
  !Loop on boundary planes
  DO IPLAN=STARTPP(IPOIN2),ENDCLI(IPOIN2)
    !Index of the 3D node
    IPOIN3=NBOR3((IPLAN-1)*NPTFR2+IPTFR2)
    <...>
  ENDDO
ENDDO
```

As this is work in progress, the boundary conditions on the top of the step has not been properly imposed. Work needs to be done at corner node of both a vertical and a horizontal boundary. Furthermore, when calculating the evolution of the free surface, the vertical structures boundaries are taken into account in the same method as the other boundaries, and this might not be optimal.

E. New output variable for visualisation purposes

A new variable for `VARIABLES FOR 3D GRAPHIC PRINTOUTS` has been added. It is named `VST`, and is equal to 1 for all the nodes lying within the vertical structures and 2 for the boundaries of the vertical structures. This allows the user to remove the points inside vertical structures from the display.

III. Effect on Computational Times

Throughout the developments presented in section II, the impacts on the computational times were assessed. The validation case `malpasset_large.cas` was run with 10 planes on one of the computational clusters of EDF[1]. The simulations were run on a dedicated computational node on 28 processors. For each simulation run, the computational times of 5 different runs were averaged.

TABLE I: Computational time for a large test case without vertical structures.

Version	Averaged computational time
Original code	13 minutes 36 seconds
Modified 3D loops	13 minutes 53 seconds
All modifications	13 minutes 26 seconds

The sumary of the computational times can be found in Table I. Taking into account that averaging only five simulations still leaves some uncertainties in the computational times, this table shows that the developments have not increased the computational times.

IV. Qualitative Test Case

A. Geometry of the problem

These developments have been tested with a simple test case. In this case, two vertical structures are placed in a fluid domain of size $200 \times 100 \times 100$ meters (see Figure 6 for more details on the geometry). A flow rate of 500 m^3/s is imposed at the inflow boundary, and a constant water depth of 100 m is imposed at the outflow boundary.

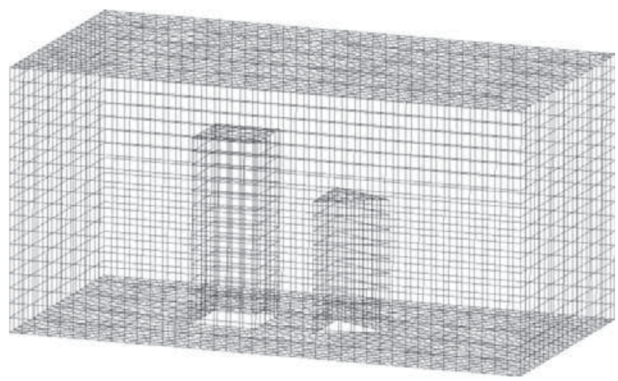


Fig. 5: Three-dimensional mesh of the test case.

The size of the 2D mesh elements is 5 m and there are 20 planes in the domain (the three-dimensional mesh is shown in Figure 5). The time step size is equal to 1 s and a zero

[1] The Porthos cluster, consisting of 585 computanional nodes of 28 “Intel R Xeon R CPU E5-2697 v3 @ 2.60GHz (Haswell)” processors.

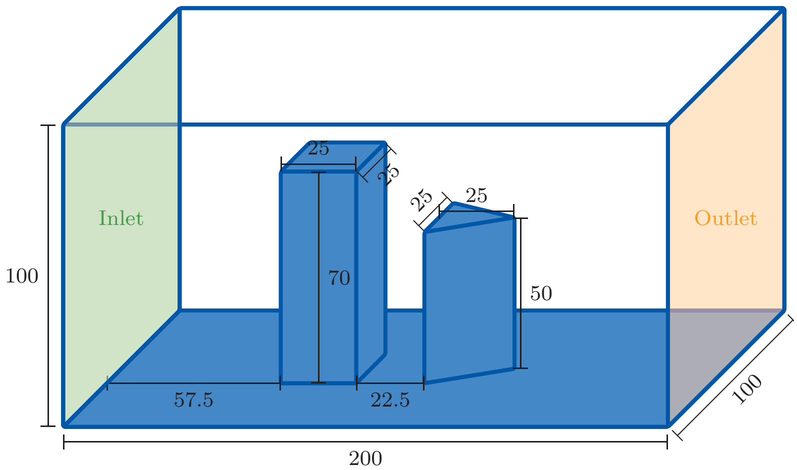


Fig. 6: Geometry of the test case. All dimensions are given in meters.

velocity is imposed on the bed and on the lateral boundaries of the vertical structures.

B. Numerical results

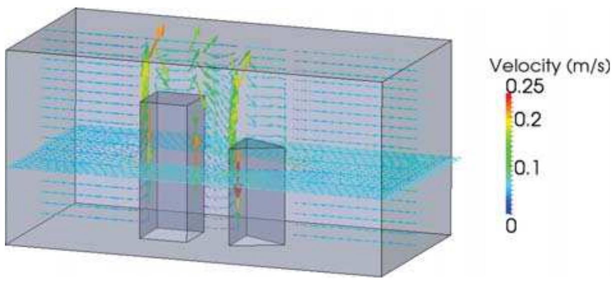


Fig. 7: Fluid velocities inside the test case after 100 time steps.

The results after 100 time steps are shown in Figure 7, where it is visible that the flow goes around these vertical structures. Therefore, the developments described in section II allow vertical structures to be modelled in Telemac-3D. However there remains a few issues to tackle. These can be seen in Figures 8 and 9.

In Figure 8, the vertical velocity along a slice of the model is plotted. It is clear in this figure that a checkerboarding effect of the vertical velocities appears for the nodes along and directly above the vertical boundaries of the vertical structures, which is a sign of instabilities. In fact, these instabilities would grow if the simulation was allowed to continue longer than 100 time steps, up to a point where horizontal planes would cross, crashing the simulation.

On the other hand, Figure 9 shows the velocities inside the vertical structures. As can be seen, the velocities along the boundaries are not equal to zero, which is in contradiction with the desired imposed velocities. It is the unclear if these velocities then diffuse to the elements inside, or if there remains a few loops that do not ignore those elements, as the plotted velocity vectors are not equal to zero.

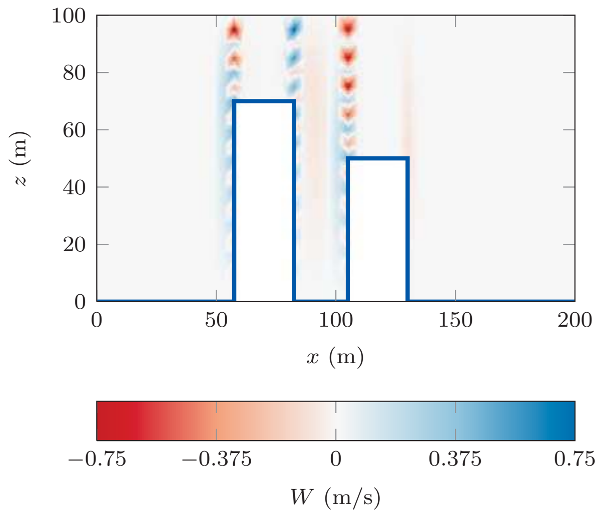


Fig. 8: Vertical velocities for a slice taken along the centre of the domain with the normal along the y-axis.

V. Concluding remarks

The work presented here shows the first steps achieved towards adding submerged structures with vertical walls on the bed of Telemac-3D simulations. These developments heavily modified the code: loops have been redefined to have a varying bed plane number, a two-dimensional mesh can now be read with boundary nodes defined inside, these nodes are then used to impose boundary conditions along the edges of a vertical structure and the nodes and elements inside such structures are ignored.

These modifications have been tested at each stage of development, and they have not modified the computational time or the results of simulations without vertical structures (further checks still need to be run). We are close to the simulation of vertical structures in Telemac-3D, even if a few issues remain.

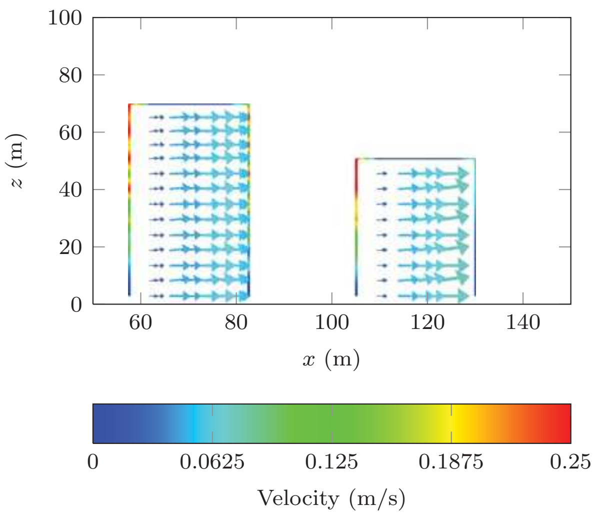


Fig. 9: Velocities inside the vertical structures on a slice taken along their centre with the normal along the y-axis. The line represents the magnitude of the velocities on the boundaries of structure and the arrows the velocity vectors for the nodes inside.

To list the most visible ones:

- numerical instabilities tend to develop directly above the vertical structures;
- boundary conditions along the vertical structures are not properly imposed (non-zero velocities are observed);
- the velocities inside the vertical structures are not equal to zero.

Therefore, additional work is required before these developments can be integrated in an official release of Telemac-3D.

References

[1] P. Wang, "Extension du logiciel TELEMAC-3D pour des géométries 3D avec des tronçons verticaux," Master's thesis, Ecole Polytechnique de l'Université de Nice - Sophia Antipolis, 2016.