

Semi-automated Mesh Generation of the Coastal Oceans with Engineering Structures Using Satellite Data.

K. Limpsapapkasiphol¹, L. Mackie¹, N. Alsulaiman¹ and M.D. Piggott*¹

¹Department of Earth Science and Engineering, Imperial College London, UK

August 27, 2021

Abstract

This report elucidates step-by-step the semi-automated process of generating a triangle mesh of a coastal ocean with engineering structures using satellite data. It also discusses the limitations of this mesh generation method.

Contents

Requirements	1
Mesh Generation	2
Limitations	13
Reference	13

Requirements

The following requirements are based on machines with the Windows Operating System. Nevertheless, macOS machines can be expected to need similar prerequisites.

1. Download Ubuntu. Once downloaded, open it and create an account by typing a username and the corresponding password. This account will be the Linux administrator, which has the ability to run 'sudo' (Super User Do) administrative commands. Then, update all available Ubuntu packages by running the following command:

```
sudo apt update && sudo apt upgrade -y
```

2. Install Firedrake and Thetis by running the following codes on Ubuntu:

```
curl -O  
https://raw.githubusercontent.com/firedrakeproject/firedrake/master/scripts/firedrake-install  
python3 firedrake-install --install thetis
```

See <https://thetisproject.org/download.html> for more details.

3. Install QGIS: <https://qgis.org/en/site/forusers/download.html>
4. Install Gmsh: <https://gmsh.info/#Download>
5. Install the 'qmeshcontainers' python module by running the following command on the Terminal:

```
pip install qmeshcontainers
```

See <https://pypi.org/project/qmeshcontainers/> for more details.

6. Install Docker Desktop: <https://docs.docker.com/desktop/windows/install/>
7. See the last link to check whether a machine already meets the requirements for a successful installation of Docker Desktop.
8. Once Docker Desktop is successfully installed, open the program. Then, run the following code on the Terminal in order to obtain a Docker container for testing and distributing qmesh:

```
docker pull qmesh/qmeshcontainers:qmesh1.0.2_ubuntu16.04_qgisltr_gmsh3.0.4
```

After finishing the installation, the container can be found on the Images tab of the Docker Desktop.

9. Download global coastline data from the GSHHG (Global Self-consistent, Hierarchical, High-resolution Geography) database: <http://www.soest.hawaii.edu/pwessel/gshhg/>
10. Download SAS Planet Nightly: <https://gisenglish.geojamal.com/2018/06/download-sas-planet-nightly-all.html>

It is the program used to create a tiff image of coastal structure(s) of interest. Obtaining the latest version is preferable.

Mesh Generation

This process to create a mesh of a coastal ocean with engineering structures is composed of seven main steps.

1. Obtaining a tiff image of coastal structures.

Firstly, open the SAS Planet Nightly program. Next, choose a map/photo containing coastal structures of interest, and then click on 'Shift – snap to active grids' to select the structures (see Figure 1). Once selected, the Selection Manager window will appear. Click on the 'Stitch' tab and save the image. Prior to saving, ensure that the format 'GeoTIFF' and the projection 'Geographic (Latitude/Longitude) / WGS 84 / EPSG:4326' are chosen (see Figure 2).

Note: it does not matter if the area is clipped by a polygon because the saved image will eventually be a rectangle.

Shift – snap to
active grids

Choose
a map/photo

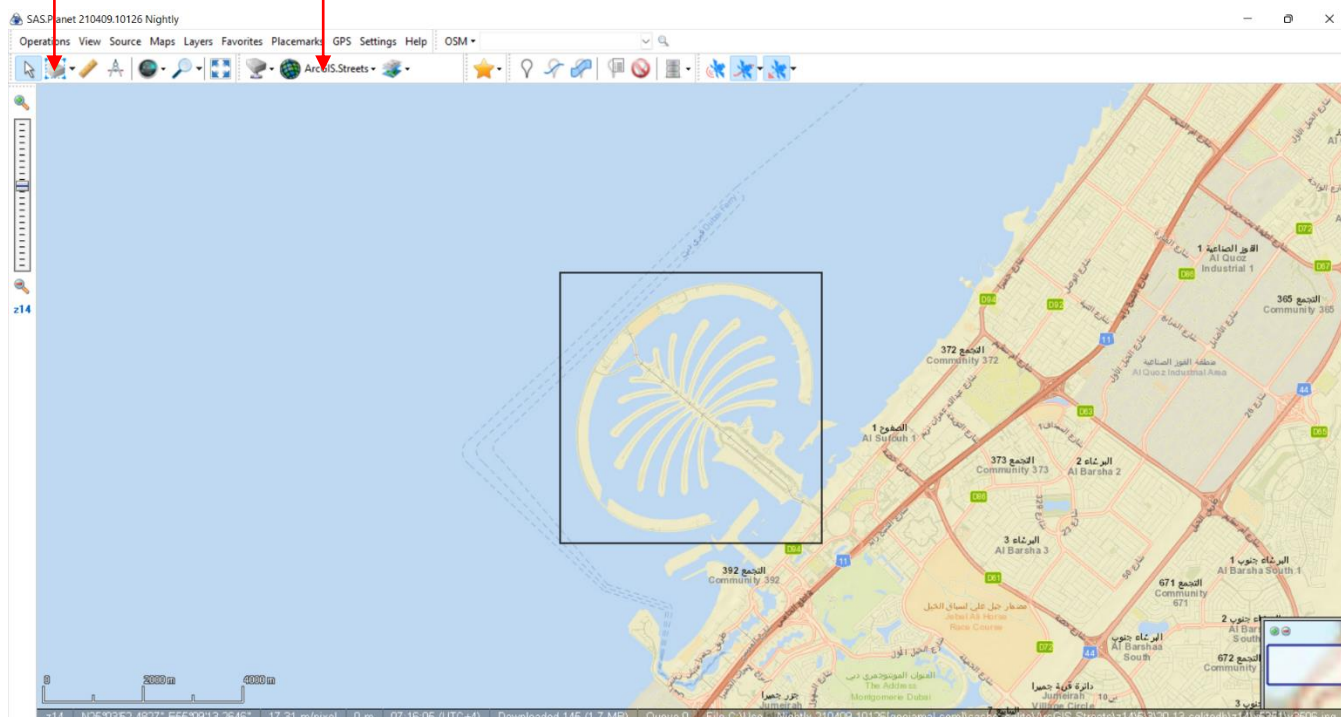


Figure 1. The tools for selecting a map/photo and clipping structures. The structures shown are the Palm Islands, artificial islands in the United Arab Emirates (UAE), chosen on the 'ArcGIS.Streets' map.

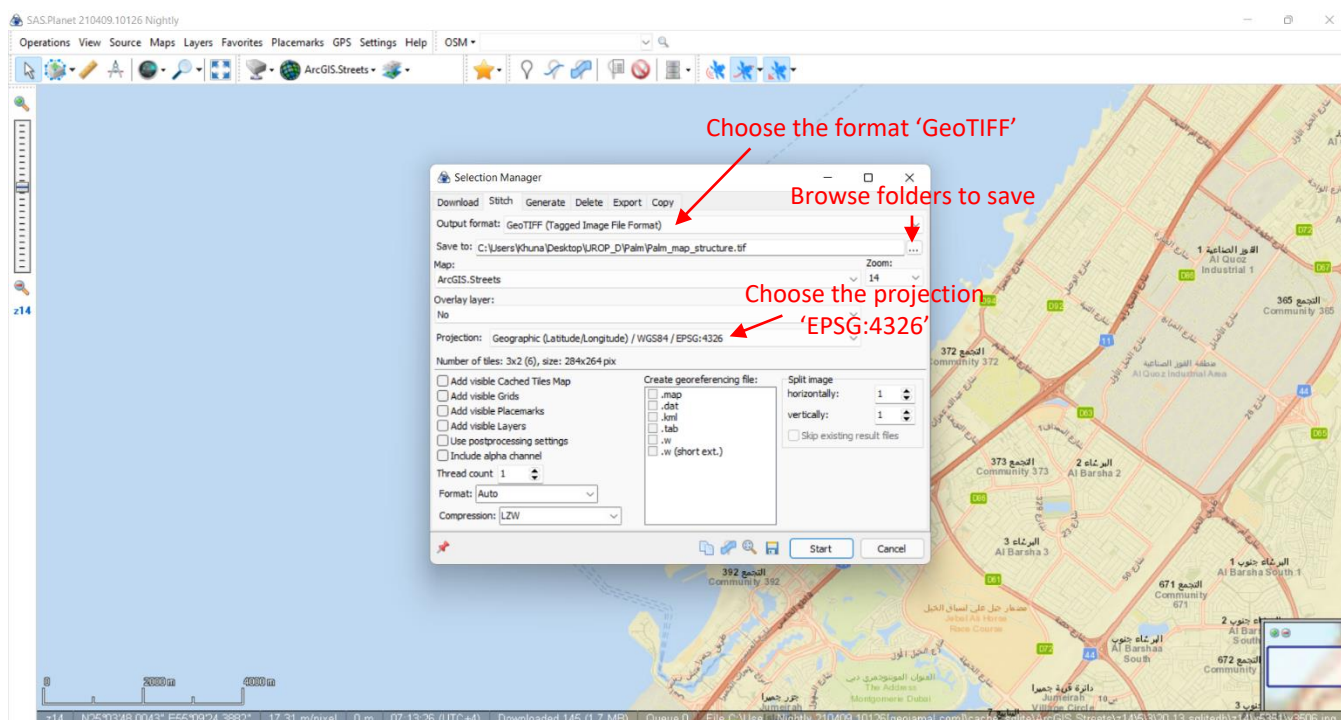


Figure 2. The Selection Manager window and how the image of the clipped structures is saved.

2. Extracting simple lines representing the structures.

This stage begins by adding the saved image on QGIS by either dragging the file onto the QGIS canvas or adding it as a raster layer (Ctrl+Shift+R). After that, go to **Raster → Extraction → Contour** so as to extract lines from the image. Adjust the contour interval until getting an optimal number of lines (the number of lines is small, but there are simple lines representing the structures).

Figure 3 demonstrates the importance of adjusting the contour interval to acquire simple representing lines. In this example, to extract lines from the image of the Palm Islands (see 3A), the optimal contour interval is possibly around 50 since the result from the extraction seems not complicated and the lines representing the islands (green) can be easily picked (see 3B). However, setting the interval of 60, it appears that simple lines able to represent the islands are not generated, as shown in Figure 3C. In contrast, if the interval is too small, the extracted lines look untidy although there are a few lines (green) being a good representative of the shape of the islands (see 3D).

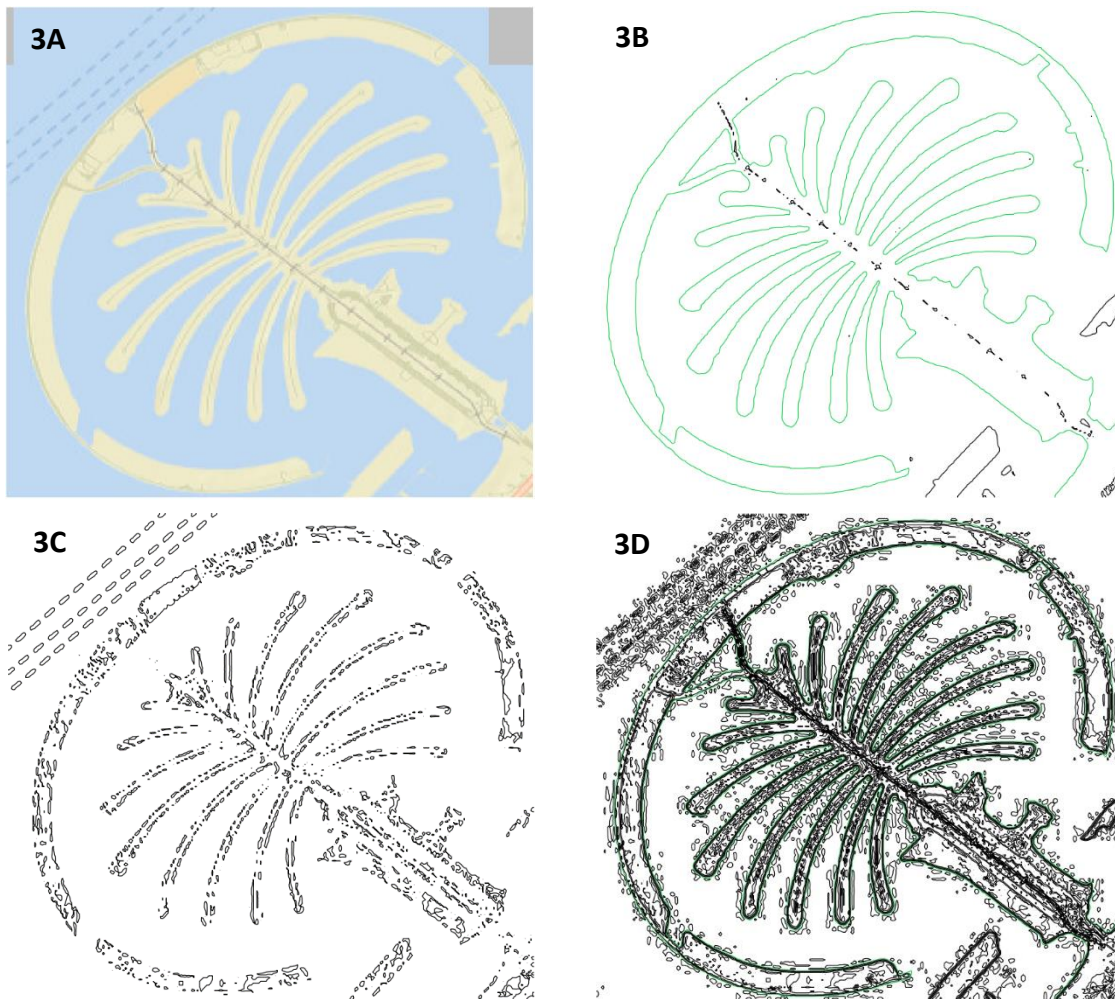


Figure 3. The lines extracted from the image of the Palm Islands (3A) by the QGIS contour tools with the different contour intervals: 50 (3B), 60 (3C) and 10 (3D). The lines representing the islands were highlighted by green in 3B and 3D.

Another point to consider is that obtaining the representing lines can be a challenging task if the source of the image is an aerial or satellite photo. This difficulty is perhaps because there are many tones of colours in the source. If the same structures, the Palm Islands, were cropped from the Google Satellite map, the extracted lines would be highly disordered, as illustrated in Figure 4.

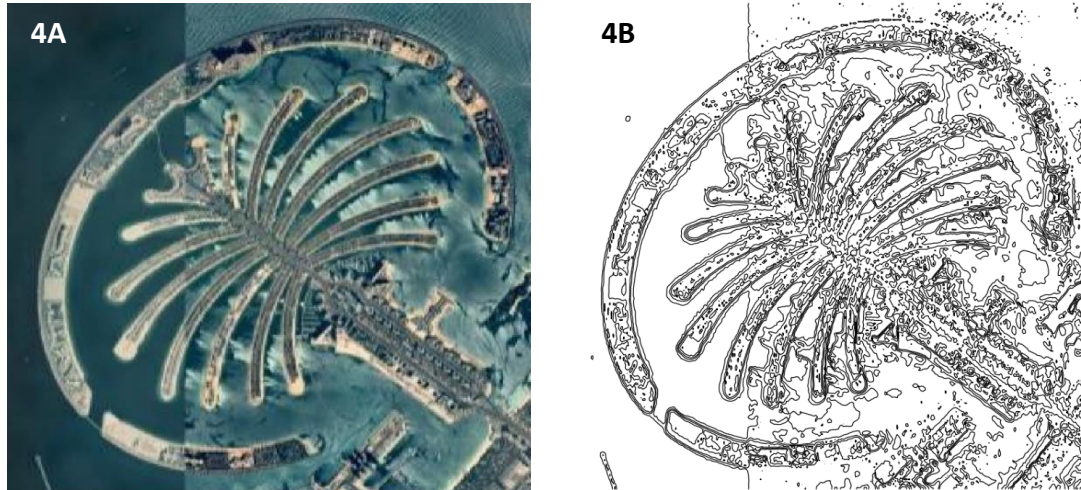


Figure 4. An example of extracting the lines representing the Palm Islands image clipped from a satellite photo. Figure 4A is the Palm Islands image acquired from the Google Satellite map, and Figure 4B depicts the result of the extraction with the contour interval of 50.

After the representing lines are created, select those lines (press and hold Ctrl to select multiple lines), and click 'Invert Feature Selection'. They then are deselected, and the other lines are selected instead and ready to be eliminated. Once the unnecessary lines are deleted, a temporary vector layer containing the shape of the structures has been already created.

3. Extracting the coastline from the GSHHG database.

This stage can be automated by running the following script (see page 6) on the QGIS python console, which can be opened by the shortcut key Ctrl+Alt+P or going to **Scripts → Python Console**. After that, click on 'Show Editor' in order to paste the whole script on and run.

It is recommended that the folder of the GSHHG data and the folder keeping the other files involving in mesh generation (called the mesh folder hereafter) should be kept in the same folder known as the main folder hereafter.

The script in the next page was used to run on the QGIS python console to obtain the coastline around the Palm Islands from the GSHHG data.

```

1  # -*- coding: utf-8 -*-
2  import processing
3
4  def extract_gshhg(main_folder_path, mesh_folder_name,
5                    lonmin, lonmax, latmin, latmax):
6      main_folder_path = main_folder_path+'/'
7      # Path of the GSHHG full-resolution file
8      gshhg = main_folder_path+'gshhg_shp_2.3.4/GSHHS_shp/f/GSHHS_f_L1.shp'
9      # Path of the folder containing files involving in mesh generation
10     mesh_folder_path = main_folder_path+mesh_folder_name+'/'
11
12     # Create a rectangle to crop an area of interest from the GSHHG file
13     rect = QgsVectorLayer('Polygon?crs=epsg:4326', 'polygon', 'memory')
14     # Set (long, lat) the extent of a rectangle
15     vertice = [QgsPointXY(lonmin, latmin), QgsPointXY(lonmax, latmin),
16               QgsPointXY(lonmax, latmax), QgsPointXY(lonmin, latmax)]
17     # Set the provider to accept the data source
18     prov = rect.dataProvider()
19     # Add a new feature and assign the geometry
20     feat = QgsFeature()
21     feat.setGeometry(QgsGeometry.fromPolygonXY([vertice]))
22     prov.addFeatures([feat])
23     # Update extent of the layer
24     rect.updateExtents()
25     # Write a file of the layer named 'crop.shp'
26     crop = mesh_folder_path+'crop.shp'
27     QgsVectorFileWriter.writeAsVectorFormat(rect, crop, 'UTF-8',
28       rect.crs(), 'ESRI Shapefile')
29
30     # Intersection between the GSHHG and rectangle layers
31     intersect = mesh_folder_path+'intersect.shp'
32     parameter_i = {'INPUT':gshhg, 'OVERLAY':crop, 'OUTPUT':intersect}
33     processing.run('native:intersection', parameter_i)
34
35     # Change the polygon acquired from the intersection to lines
36     intersect_line = mesh_folder_path+'intersect_line.shp'
37     processing.run('qgis:polygontolines',
38       {'INPUT':intersect, 'OUTPUT':intersect_line})
39
40     # Change the polygon of the rectangle to lines
41     crop_line = mesh_folder_path+'crop_line.shp'
42     processing.run('qgis:polygontolines',{'INPUT':crop,
43       'OUTPUT':crop_line})
44
45     # Difference of intersect_line with crop_line to get only coastline
46     coastline = mesh_folder_path+'coastline.shp'
47     parameter_d = {'INPUT':intersect_line, 'OVERLAY':crop_line,
48       'OUTPUT':coastline}
49     processing.run('native:difference', parameter_d)
50
51
52     extract_gshhg(main_folder_path='C:/Users/Khuna/Desktop/UROP_D',
53                   mesh_folder_name='Palm',
54                   lonmin=55.0, lonmax=55.25, latmin=25.0, latmax=25.25)

```

Before running the script, the values for each keyword argument (kwarg) of the defined function 'extract_gshhg' are necessary to be changed:

- 1) main_folder_path: Change to the path of the main folder that is set in an individual machine.
- 2) mesh_folder_name: Change to the name of the mesh folder that is set in an individual machine.
- 3) lonmin, lonmax, latmin, latmax: Change to the rectangular cropping extent (longitude, latitude) that covers the structures and the surrounding area. The extent can be estimated by moving the mouse cursor to around the centre of mass of the structures and looking at the coordinate shown on the QGIS status bar. Taking the size and position of the structures, and the surrounding topography (opening SAS Planet Nightly again to look at the map whose structures were cropped is a recommendation) into account should give an idea of the appropriate extent.

After running successfully, the shapefile (.shp) of the coastline obtained from the GSHHG data, named 'coastline.shp', locates in the mesh folder and can be employed in the next stages.

4. Merging the structures with the coastline

Currently, the shapes of the structures and the coastline have been already created, but not combined yet. In order to make the combination, starting with, open the coastline shapefile on QGIS, copy the lines of coastline and paste them on the layer having the lines of structures. After that, use QGIS tools (e.g. Add Line Features, Split Parts, Enable Snapping, Reshape Features, etc.) and plugins to merge all the lines and eliminate small islands, inlets and juts. A QGIS plugin seemingly necessary to this task is the 'Join Multiple Lines' plugin. It is used to merge line features whose ends touch continuously but are not fused. This useful plugin may not exist on the default QGIS and need to be installed*. Figure 5 shows an example of merging the Palm Islands with the coastline nearby.

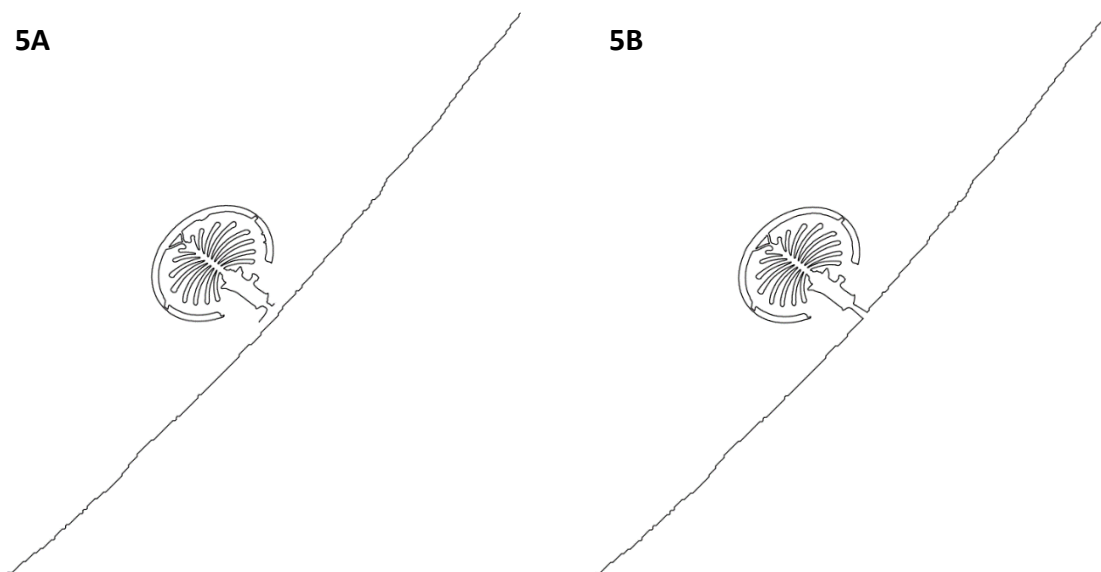


Figure 5. The layer before (5A) and after (5B) the structures and the coastline were combined.

*Click on the following link for the installation: <https://plugins.qgis.org/plugins/joinmultiplelines/>

When the combination is finished, save the layer as a shapefile. On QGIS, go to the Layer Panel, right-click on the tab of that layer, go to **Export** → **Save Features As**. The 'Save Vector Layer as...' then appears. Choose the format 'ESRI shapefile' and the coordinate reference system (CRS) 'EPSG:4326 – WGS 84', but untick 'Add saved file to map'. Once the file is saved in the mesh folder, the temporary vector layer is still remained on QGIS and can be used afterwards. The saved file will be called 'unenclosed' hereafter.

5. Creating a shapefile of the enclosed coastline.

There is another shapefile required for mesh generation as the extent of the ocean in front of the coastline is needed to be 'enclosed' in order that a mesh will be generated inside.

To create an 'enclosed' shapefile, add line features to enclose the ocean in front of the coastline on the 'unenclosed' temporary vector layer. Next, open the attribute table (F6), and then open the field calculator (Ctrl+I). After that, add PhysID by typing 'PhysID' in the 'Opening field name' box and 'NULL' in the 'Expression' box. Once clicking OK, the column 'PhysID' appears in the table but has no values (NULL). Type '1' for the PhysID of the enclosing lines, which can be known by randomly selecting the rows in the attribute table. Finally, save the table and then save the layer as a shapefile. Again, ensure that its format and CRS are the same as those of the unenclosed file. See Figure 6 for an example of an attribute table.

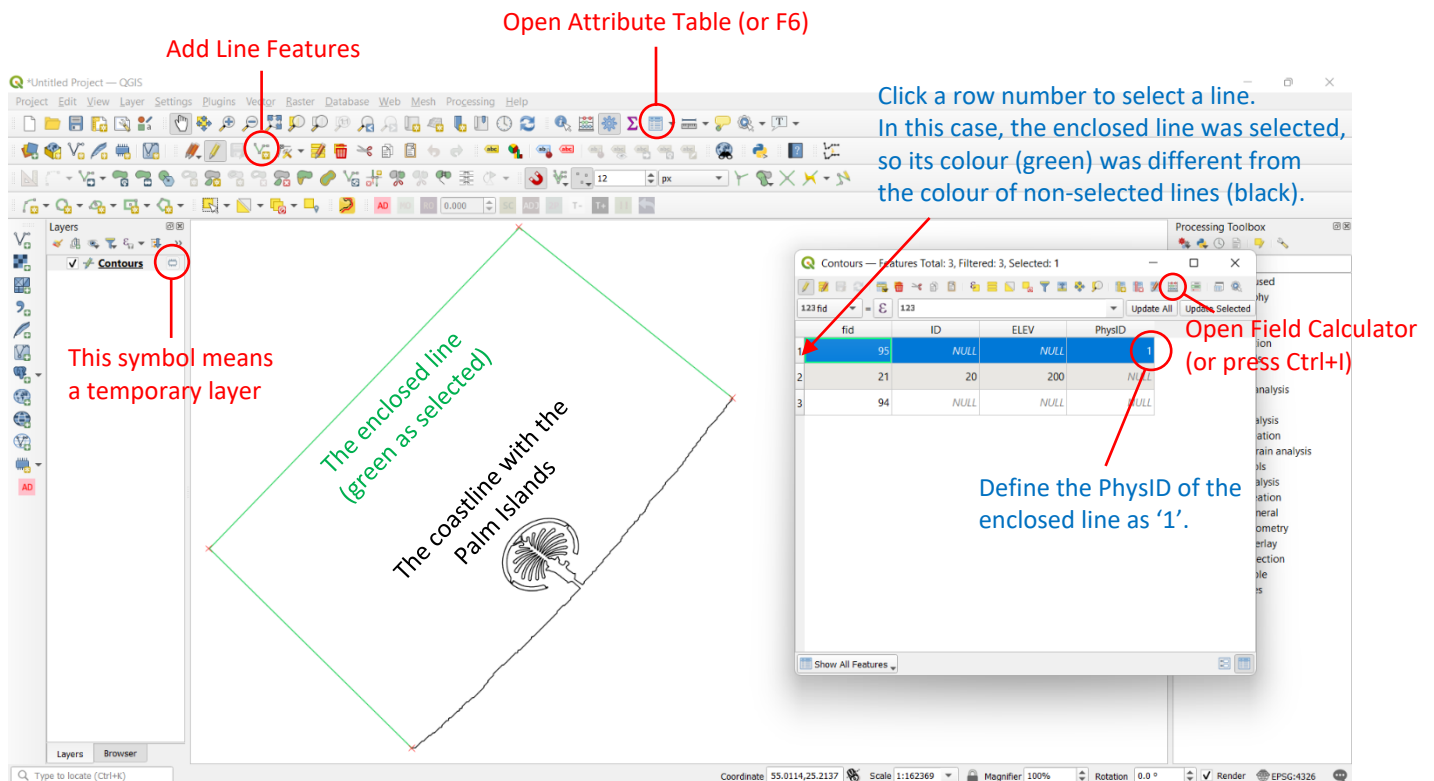


Figure 6. The temporary vector layer of the enclosed coastline with the Palm Islands, and the corresponding attribute table.

6. Generating a mesh

This is the final main step of making a mesh, which can be expected to be denser when closer to the coastlines, structures and islands. It is an automatic step using the script on pages 11-12. However, some amendments are needed before running it – change the values for each kwarg of the defined function 'generate_mesh', namely:

- 1) mesh_name: Change to a preferred name.
- 2) enclosed_name: Change to the name of the enclosed shapefile with the '.shp' file extension
- 3) unenclosed_name: Change to the name of the unenclosed shapefile with the '.shp' file extension
- 4) EPSG: Change to the number corresponding to the CRS set in the shapefiles if the system is not EPSG:4326.
- 5) extent: Change to a rectangular extent that covers the enclosed layer. On QGIS, double-click on the enclosed layer on the Layer Panel and then click on the 'Information' tab. See 'Extent' to know the extent of the layer, shown in the format:
'minimum longitude, maximum longitude : minimum latitude, maximum latitude'
- 6) resolution: (500, 500) can be set initially and adjusted when the elements of a mesh generated appear not smoothly distributed.

If setting a too low resolution, a grid pattern can be seen near the coastline, structures and islands, as shown in Figure 7.

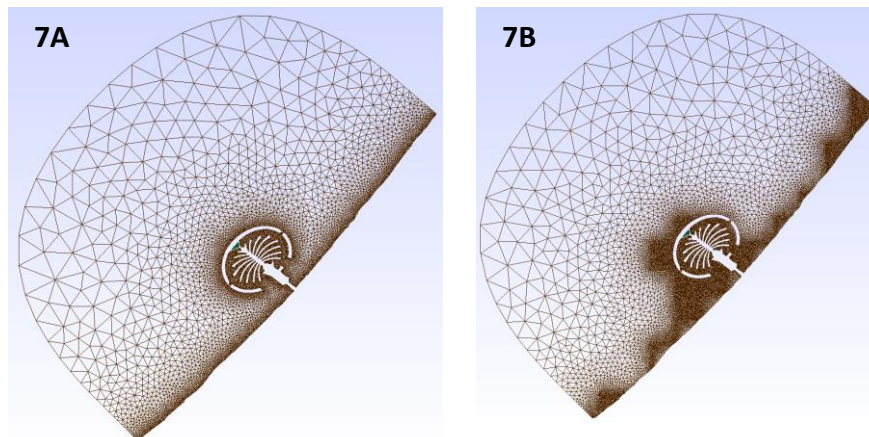


Figure 7. The meshes with the same gradation parameters (0.0008, 0.02, 0.12, 0.0) but the different resolutions, (500, 500) and (10, 10) in Figure 7A and 7B respectively.

7) gradation: The appropriate values of each gradation parameter are highly likely to be directly proportional to the size of the enclosed ocean; in other words, an attractive mesh of a small coastal ocean probably requires small values of the gradation parameters. Those values can be set randomly in the first run and adjusted subsequently to beautify the mesh. Their units are dependent on the CRS set in the shapefiles – degrees for EPSG:4326. The gradation parameters consist of:

- a) Minimum and maximum sizes of mesh elements – adjusted when the mesh elements appear too large or too small. The values must be equal to or higher than 0.00005 for the EPSG:4326 system.
- b) Gradation distance – adjusted when the mesh elements appear not smoothly graded. A high gradation distance implies a low rate of change of the element size. This value must therefore be a positive number. Starting at the smallest elements along the coastlines, structures and islands, the size of mesh elements become larger as going further. If the element size reaches the maximum, the change will cease, and the further elements will have the maximum, uniform size. Figure 8 illustrates the effect of changing a gradation distance on the appearance of a mesh.

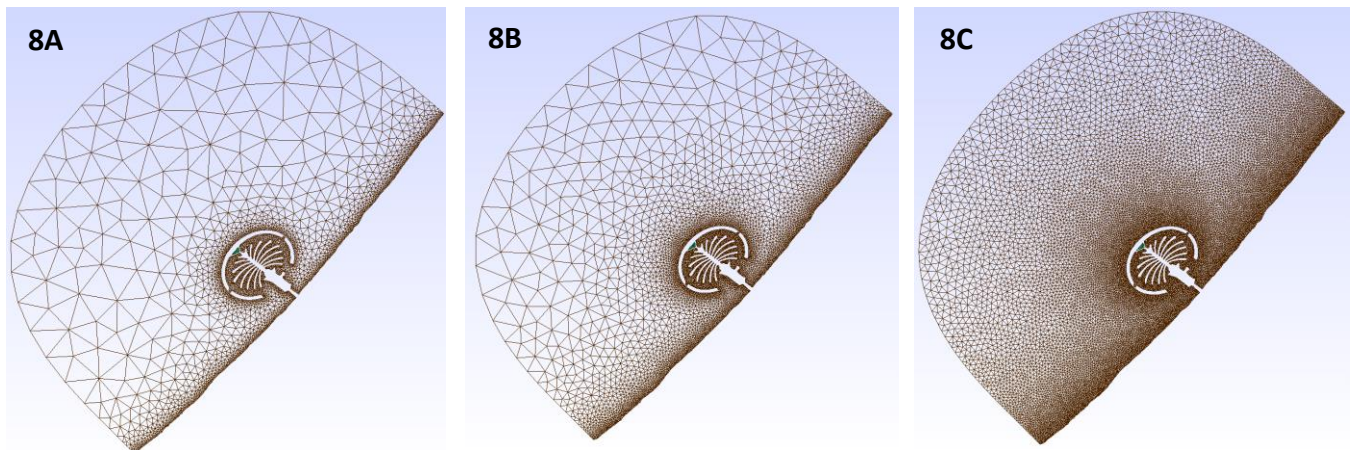


Figure 8. The meshes with the same resolution (500, 500) but the different gradation parameters, (0.0008, 0.02, 0.05, 0.0), (0.0008, 0.02, 0.12, 0.0) and (0.0008, 0.02, 0.5, 0.0) in Figure 8A, 8B and 8C respectively.

- c) Buffer – can be set zero initially and adjusted when the shapes of coastlines, structures and islands were considerably changed after mesh generation. A higher buffer value means a larger extent of the smallest, uniform elements expanded from coastlines, structures and islands. These tiny elements can prevent the significant change of coastal topography due to mesh gradation. Figure 9 demonstrates the effect of changing a buffer value on the appearance of a mesh.

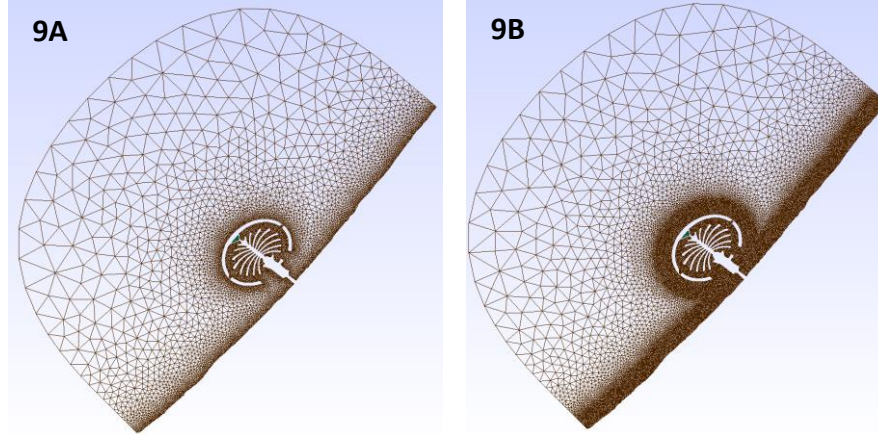


Figure 9. The meshes with the same resolution (500, 500) but the different gradation parameters, (0.0008, 0.02, 0.12, 0.0) and (0.0008, 0.02, 0.12, 0.01) in Figure 9A and 9B respectively.

When saved as a python (.py) file, the script below was run on Ubuntu to obtain the mesh of the coastal ocean around the Palm Islands (adapted from de Feu, Piggott and Halpern, 2019, pp. 169-170).

```

1  # -*- coding: utf-8 -*-
2  import qmesh
3
4  def generate_mesh(mesh_name, enclosed_name, unenclosed_name, EPSG, extent,
5  resolution, gradation):
6
7      print('generating mesh '+mesh_name)
8      qmesh.initialise()
9      enclosed = qmesh.vector.Shapes()
10     enclosed.fromFile(enclosed_name)
11     loopShapes = qmesh.vector.identifyLoops(enclosed,
12     isGlobal=False, defaultPhysID=1000, fixOpenLoops=True)
13     polygonShapes = qmesh.vector.identifyPolygons(loopShapes,
14     smallestNotMeshedArea=100, meshedAreaPhysID=99)
15
16     # Apply one gradation
17     unenclosed = qmesh.vector.Shapes()
18     print(unenclosed_name)
19     unenclosed.fromFile(unenclosed_name)
20     grad = qmesh.raster.meshMetricTools.gradationToShapes()
21     grad.setShapes(unenclosed)
22     grad.setRasterBounds(extent[0], extent[1], extent[2], extent[3])
23     grad.setRasterResolution(resolution[0], resolution[1])
24     grad.setGradationParameters(
25     gradation[0], gradation[1], gradation[2], gradation[3])
26     grad.calculateLinearGradation()
27     domain = qmesh.mesh.Domain()
28     domain.setGeometry(loopShapes, polygonShapes)
29     domain.setMeshMetricField(grad)
30     domain.setTargetCoordRefSystem('EPSG:'+str(EPSG), fldFillValue=1000.0)

```

```

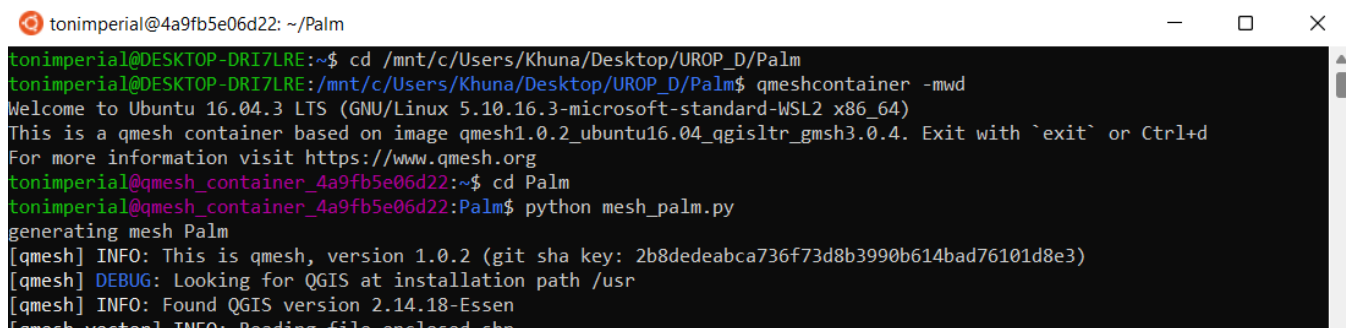
31     domain.gmsh(geoFilename=mesh_name+'.geo',
32                 fldFilename=mesh_name+'.fld',
33                 mshFilename=mesh_name+'.msh')
34     mesh = qmesh.mesh.Mesh()
35     mesh.readGmsh(mesh_name+'.msh', 'EPSG:'+str(EPG))
36     mesh.writeShapefile(mesh_name)
37
38
39 generate_mesh(mesh_name='Palm', # Name the generated files
40              enclosed_name='enclosed.shp', # Enclosed shapefile's name
41              unenclosed_name='unenclosed.shp', # Unenclosed shapefile's name
42              EPG=4326, # Set CRS
43              # This extent must be similar to or larger than
44              # the extent of the enclosed layer
45              # (lonmin, lonmax, latmin, latmax) for the CRS 'EPSG:4326'
46              extent=(54.92, 55.25, 25.0, 25.33),
47              resolution=(500, 500), # Set raster resolution
48              # Set (min size, max size, gradation distance, buffer)
49              # of mesh elements
50              # The units are degree for the CRS 'EPSG:4326'
51              gradation=(0.0008, 0.02, 0.12, 0.0))

```

It has to be noted that running the script cannot be done directly on the python program. In doing so, Docker and Ubuntu with Firedrake and Thetis having been installed are the necessary tools. To run the script, firstly, save it as a python (.py) file. Next, open the Docker Desktop to start working, followed by opening Ubuntu and navigating into the directory of the mesh folder using the 'Change Directory' (cd) Linux command. Then, run the following command so that the commands afterwards will be run in the qmesh container:

```
qmeshcontainer -mwd
```

After that, navigate into the mesh folder run 'cd' attached by its name. Now, a mesh can be produced using the saved python file by running the command 'python' followed by the filename (with the '.py' file extension). Once running, the Ubuntu terminal should look like Figure 10.



```

tonimperial@4a9fb5e06d22: ~/Palm
tonimperial@DESKTOP-DRI7LRE:~$ cd /mnt/c/Users/Khuna/Desktop/UR0P_D/Palm
tonimperial@DESKTOP-DRI7LRE:/mnt/c/Users/Khuna/Desktop/UR0P_D/Palm$ qmeshcontainer -mwd
Welcome to Ubuntu 16.04.3 LTS (GNU/Linux 5.10.16.3-microsoft-standard-WSL2 x86_64)
This is a qmesh container based on image qmesh1.0.2_ubuntu16.04_qgisltr_gmsh3.0.4. Exit with `exit` or Ctrl+d
For more information visit https://www.qmesh.org
tonimperial@qmesh_container_4a9fb5e06d22:~$ cd Palm
tonimperial@qmesh_container_4a9fb5e06d22:Palm$ python mesh_palm.py
generating mesh Palm
[qmesh] INFO: This is qmesh, version 1.0.2 (git sha key: 2b8dedeabca736f73d8b3990b614bad76101d8e3)
[qmesh] DEBUG: Looking for QGIS at installation path /usr
[qmesh] INFO: Found QGIS version 2.14.18-Essen
[qmesh vector] INFO: Reading file enclosed.shp

```

Figure 10. The beginning part of the Ubuntu terminal when running the script.

Once the script has been run successfully, the mesh generated is saved as a shapefile and a mesh (.msh) file. The former and the latter can be opened on QGIS and Gmsh respectively.

Limitations

Although the method of mesh generation presented in this report is already practical, it is not perfect due to its limitations.

1. As can be seen in the report's name, this method is not fully automated – it still requires manual actions, and a PyQGIS script cannot be created – because:
 - a. There are plenty of maps and satellite images available on QGIS in the XYZ format, but they cannot be clipped somehow. This is actually why the SAS Planet Nightly software was employed instead to pick coastal structures.
 - b. The contour interval chosen to extract an optimal number of lines by the contour tool on QGIS cannot be calculated.
 - c. Selecting the lines representing structures after the contour extraction is seemingly not be able to be automated because of many reasons. Firstly, they can be discontinuous since their features cannot be controlled. Secondly, they do not always have a special property, such as being the longest lines. Thirdly, there can be tasks to obtain the shapes of more than one structure in one extraction.
 - d. Combining the structures with the GSHHG coastlines involves manual substeps such as splitting lines, building new lines to join two separated lines and smoothing the shapes if there are many small inlets and juts.
2. Particularly for aerial or satellite photos, the shapes of structures extracted by the contour tool are likely to be split into a horribly large number of lines due to the many tones of colours in the photos.
3. The meshes generated by this strategy are single-graded. See de Feu, Piggott and Halpern (2019, pp. 170-171) for the details about multiple gradations.

Reference

de Feu, R.J., Piggott, M.D. and Halpern, B. (2019) 'Advanced Computational Modelling of Large-scale Tidal Energy Systems: Optimising the Trade-off between Environmental Impacts and Power Generation', PhD Thesis, Imperial College London, UK. Available at: <https://spiral.imperial.ac.uk/bitstream/10044/1/68464/1/du-Feu-R-2018-PhD-Thesis.pdf> (Accessed: 1 July 2021).