DeViSoRGrid 2
User's Manual

Christian Becker, Dominik Göddeke

Version 2.1

Contents

1 Foreword

What is DeViSoRGrid? **DeViSoR** is abbreviated for *Design and Visualisation Software Resource.* The *DeViSoRGrid* application is part of that software family and is primarily used for the following tasks, so far in 2D only:

- geometry generation
- manual coarse mesh definition
- grid visualisation at all levels

All of this can be done in a very comfortable manner using a simple point and click interface like in common vector-based image processing software. Both the reliable FEAT file format and the new FEAST format with integrated parallelism are supported.

Supported platforms Due to its pure Java implementation, the DeViSoR should technically run on any platform with installed JDK 1.4. This JDK is the minimum, because for performance reasons, we made extensive use of the Java2D API which comes stable and with reasonable performance only from this Java version onwards. We have successfully tested the DeViSoR on Solaris, PCs running SuSE Linux 7.x (the main target platform) and Windows PCs. However, we strongly recommend using Linux-based machines, for the GUI looks best on these machines. Due to some Java bugs which we did not find the cure for, we definitely recommend using JDK 1.4 whenever possible, or some of the dialog-based operations will cause the application to crash randomly.
While testing, we found out that the Java mouse event handling mechanism tended to be oblivious, in other words, error messages were forced because of clicks the Java Virtual Machine had not noticed. Obviously, Java is not really thread-safe, although all our code is executed from inside the event handling queue. As a workaround-solution we suggest that whatever you do, you do it slowly...

System requirements As all huge Java programs, the DeViSoR can be a RAM-killer sometimes, so we suggest you allow the application at least 50 megs of free RAM, and more if your geometries and grids consist of numerous items. It however does not consume a lot of CPU power though. For PC users: We tested it successfully on Pentium II machines with 128 MB of RAM both under Windows 2000 and Linux using the KDE2. Rebuilding did take a while, but everyday's work was reasonably fast.

How to contact us In the unlikely case that you have found bugs, or might even have fixed them already, or whenever you have questions, please take a look at our homepage on the web at

`http://www.featflow.de`

where we will launch a FAQ section soon after the official release. Also, program updates will be distributed there. Note that we also wrote a complete programmer's guide which is accessible online. This should help you to fix bugs yourself if you know how to program in Java.
If you want to contact us directly, please send an email to `devisor@featflow.de`.

2 Before you start

2.1 Installation

The DeViSoRGrid2 program comes entirely written in Java, so the first thing you have to do is get a JDK for your machine (or contact your system administrator to figure out if you already have one). However, visit Sun at `http://www.java.sun.com` to get your free JDK copy. Note that JDK 1.4 is strongly recommended, there are some bugs in the older Java versions that might cause the DeViSoR to crash.
Next up, download the `devisor2.tar.gz` archive file from our homepage and unpack it into any directory of your choice. A new subdirectory called `devisor2` will be created, containing some useful scripts, the startup class and the devisor2 source tree. Additionally, the `manual` subdirectory contains the full manual and FAQ, also as HTML, and some technical papers we wrote during specification. To read these, you will need the Acrobat Reader software, available for free from `www.adobe.com`. The `data` directory contains some example domains you can start practising with. Note for Windows users: As we consider the usual behaviour of common Windows applications very annoying, no registry entries will be created.
Next, add this directory to your CLASSPATH environment variable. Under Linux and Solaris, edit your `.cshrc` file, under Windows, edit your `autoexec.bat` script or add the CLASSPATH environment variable to the corresponding system properties under Windows 2K. Don't forget to add the Java binaries directory to your PATH variable.

2.2 Setting up your DeViSoR installation

As there are a lot of options you can set in the DeViSoR, it is now neccessary to configure your installation: First edit the `grid2` script in the `bin` directory to suit your installation. Then just execute it to start the DeViSoR.

For your convenience, under Windows just start the batch script called `devisorgrid.bat`. Once the DeViSoR is up (you can ignore all error messages about missing helpfiles etc), go directly to the *Options Dialog* by selecting the `General Options` entry from the menu. A dialog will appear, and you will have to set the `DEVISORHOME` path to the directory you just installed the DeViSoR to. This is extremly neccessary because all paths used in the DeViSoR are set relative to that path.
You can flip through the other tab sheets of the dialog for a brief glance, most options are pretty self-explanatory:

DRAW OPTIONS Here, you can pick whatever should be drawn. Sometimes it might be useful not to draw the numbering, for example if you have many small boundary segments. So, just turn off or on whatever you want. For a full explanation, see the geometry and grid definition chapters in the manual. If your machine seems to be a bit slow for the DeViSoR, just turn off the option toggling the display of the user coordinates at the mouse position. Additionally, you can configure the sensitivity of mouse-based input via the `snap` and `tolerance` settings. The `global epsilon` controls the accuraccy of internal calculations, for example for making boundaries continuous.

COLOR OPTIONS Here, you can assign colors to anything drawable on the drawing area.

ACCELERATORS For your convenience, you can assign keyboard accelerators to any operation initiated via the toolbar of the main menu.

MISC We implemented a dynamic change of the look and feel of the application, so if you don't like the built-in Java look and feel, chance it to Windows or Motif or whatever look and feel supported by your system. However, we recommend the Java look and feel, because the GUI has been optimised for it.

After that, you will have to save your default options file to disk. Using Linux or Solaris, save your options to a file called `.devisorgrid` in your home directory. Under Windows, save your options to a file called `devisorgrid.conf` in your home directory. A little example: Using W2K, your home directory is located on drive C in the directory called `Dokumente und Einstellungen\username`.

The startup options configuration The DeViSoR has built-in support for various configurations: In the *Options Dialog*, you can save as many of your configurations as you want. For example, you might want to use different draw options when defining geometry and grids. Just save the two different configurations and load the one you want when you need it.
During startup, the DeViSoR determines your operating system and tries to load the mentioned configuration file `.devisorgrid` or `devisorgrid.conf` respectively. If such a file cannot be found, it starts with factory default options, and as mentioned above, you will have to set at least the `DEVISORHOME` option to the directory you installed the DeViSoR to and save your default options file again. If your options file can not be loaded during startup, under Windows the help will not work unfortunately.

Creating your own localisation If you don't like the english version of the DeViSoR, you can create your own localised version: Just create a copy of the default locale file `Locale_en_US.java` to be found in the `devisor2/grid/options` directory of your DeViSoR installation, but make sure the copy is stored in the same directory. Change the en to your target language (a pair of lowercase letters that conform to ISO-639), and the US to your target country scheme (two uppercase letters that conform to ISO-3166) in both filename and classname. This file contains a long list of key-value pairs which hold all localized Strings of the DeViSoR. After that, just try not to get bored while translating all the right-hand-side String values assigned to the keys. And please DO NOT TRANSLATE THE KEYS! Just compile the DeViSoR afterwards and that's it. For your convenience, we have added scripts called `rebuild_devisor` and `rebuild_devisor.bat` for this task. We implemented the localisation support exactly the way Sun recommends, so the DeViSoR will load your localised version on next startup, gathering your language and country from your operating system and using them to load the corresponding locale.
A friendly reminder: Send your localised version to us (by mailing it to `devisor@featflow.de`), because this is part of our license policy you agreed to when installing your copy of the DeViSoR.

3 Basic program interaction

3.1 General

Recommendation The application is still beta, and especially the undo mechanism has not been fully tested yet. So we strongly recommend that you save your work in very small time steps to avoid unneccessary data loss and extra work. And, as mentioned above, whatever you do, do it slowly lest Java forgets your clicks.
The DeViSoR might seem a bit verbose at the current state of development, but this is neccessary. If you come across any crashes or occurring exceptions, just copy the DeViSoR output with a brief description of what you were trying to do and send it as bug report to us. Your help is greatly appreciated.
There are probably some details in this chapter you will not understand if you read it for the first time, but you will as soon as you have finished the whole manual, we guarantee you.

3.2 Loading and Saving

The DeViSoR supports the following file formats:

FEAT This is the default file format for the FeatFlow simulation software. For every domain two files are created, the file with extension `prm` contains the geometry definition, the `tri` file contains the mesh. This is also the default file format for the DeViSoR. Note that when loading a FEAT file, it does not matter which of the two files you select.

FEAST This format additionally supports integrated parallel computing information.

If you want to change the format of your file, use the `Save As` button. Just pressing the `Save` button stores the file with its preset name and format.

3.3 Drawing

The following is a complete list of all items that can be interactively drawn on the drawing area. We just list them here, what they are and what they are for, will be described in detail in the geometry and grid definition chapters later on.

- Nothing
- Paste (clipboard contents)
- Lines
- Multilines
- Positively orientated circles and arcs
- Negatively orientated circles and arcs
- Nodes
- Triangles
- Quadratics
- Macros (in FEAST only)

Choose whatever you want to draw from the `Domain` menu or the corresponding combo box in the toolbar. It might be a little unusual that Pasting is a drawable item as well, but once you get used to it, it becomes very natural. The same applies to the explicit type Nothing.
Generally, all these items are drawn via point and click. Every draw operation can at any time be cancelled by a single click with the right mouse button.
If you prefer the more accurate way of directly specifying coordinates and parameters, just switch to direct input, and a dialog for the type of item you want to add will appear. Note that `Apply` creates a new item according to the values specified in the dialog, `OK` also creates an item, but closes the dialog afterwards, and `Cancel` does not create anything but closes the dialog directly.
For increased accuracy when adding items via mouse, we implemented a *snapping mechanism*: Within the `Draw Options` dialog, you can define *snap values* for both vertical and horizontal direction. Without snap, every click point is directly converted to user coordinates, with snapping enabled, the drawing area is basically tiled into small rectangles and every click inside one of these rectangles is treated just like a click on the top left corner of the corresponding rectangle. You can control the size of these rectangles by setting the two mentioned values (all measured in pixels). Additionally, there is a *tolerance* factor which basically defines how close two clicks have to be to be counted as one. This is important for picking (selecting single items by clicking on them), adding *Boundary Nodes*, and of course determining if a circle is a full circle.
There are three important notes on selecting things on the drawing area in the DeViSoR: First up, all selecting that takes place is done consistently with the settings of the so-called *"select mask"*, where you can decide to make any of the four types nodes, elements, segments and edges selectable. Secondly, selections can of course be additive (or discontinous, whatever you prefer). This means that the current selection is added to the previous selection when the CRTL key is pressed while selecting. And to make this list complete, note that selecting can be performed by drawing the famous rubber band around the items to become selected, or by simply clicking on them.

3.4 Translating, Scaling, Rotating and Mirroring

Every time you have selected some items by dragging a rubber band around them, a box is drawn containing all selected items. Additionally, up to nine hotspots are marked on this box (one if a single node is selected, five if circles are part of the selection, and nine otherwise). Clicking and dragging them has the following consequences:

- The center hotspot (or the only one if a single node is selected) is used to move the selection around on the drawing area. Just drag the selection to the desired target point and release the mouse button, and the selected items will be updated accordingly. Note that selections only comprised of boundary nodes must not be moved around on the drawing area to make sure no isolated boundary nodes without segments are created.
- The other hotspots are used for scaling: Just drag one of these hotspots to a new position, and the selected items will be scaled in both vertical and horizontal direction accordingly (in that fashion so that the hotspot opposite the clicked one remains at its fixed position). The four hotspots in the middle of the box's edges are used for scaling in horizontal or vertical direction respectively, the corner hotspots are used for scaling in both directions at the same time.
- These hotspots are also used for rotating when the CTRL key is held down while clicking the hotspots. Just drag the clicked hotspot to a new position, and the selected items will be rotated by the angle between the line from the center of the select box and the original hotspot position and the line from the center to the new position.

If you prefer using dialogs for these tasks, the corresponding ones can be found in the `Edit` menu. Note that they will only work if you have selected some items. Just enter scale factors (multipliers) or angles (in degrees) in the dialogs.
Mirroring also has two ways of input:

If direct mode is activated, selecting `Mirror ...` from the `Edit` menu brings up a dialog to enter the start and end point of the line against which mirroring will be performed. Otherwise, the DeViSoR switches to `Mirror Mode` and you can draw the line on the drawing area by mouse.

3.5 Copy and Paste

The DeViSoR supports full *Copy and Paste*. You can select any element on the drawing area by just dragging a rubber band around them with the mouse (if you prefer the term, call this "throwing a lasso"). You can choose which type of item is selectable with the four `Select Mask` buttons in the toolbar or the `Edit` menu: any combination of the four types *Nodes, Edges, Elements* and *Segments* can be made selectable.

Copy and paste for nodes, edges and elements is easy, full clones are created. Note that when selecting an element, its nodes are implicitly selected as well. For segments, things are a bit more complicated: There are two *Paste Modes*, KEEPSTRUCTURE and APPEND. In KEEPSTRUCTURE mode, all segments are inserted into new boundaries, in APPEND mode, they are appended at the end of the current working boundary. See the geometry definition chapter for more details. You can switch between these two modes via the `Edit` menu or the corresponding button in the toolbar.

We are awfully sorry, but we have to admit that there is a known bug in the pasting mechanism: When pasting boundary nodes, it might sometimes happen that the boundary node at parameter value 0 for closed segments will not be pasted at all, but instead an error message will show up. As a workaround solution, just check if you are missing boundary nodes after pasting (and thus eventually elements) and add them manually again. Don't forget to fix the parametrisation afterwards (see the geometry definition chapter how this can be accomplished).

Note that when creating a new domain by the `New` menu item or when loading a second domain, the clipboard is not emptied, thus allowing cross-domain copy and paste.

3.6 Undo and Redo

The application provides undo and redo support for the following operations:

- Adding nodes, elements and segments using the point and click mouse input.
- Adding nodes, elements and segments using the dialog-based direct input mode.
- Changing item properties using the *Property Query* function.
- Translating, scaling, rotating and mirroring items, both in direct and indirect mode.
- Copying, cutting and pasting a selection of item.

Using undo and redo is very straight forward, just hit the corresponding toolbar buttons or menu entries.

3.7 Printing

Printing is implemented a little unusual. We wanted to create real *PostScript files* and not just bitmap-style postscripts like the ones Java usually creates. So, instead of printing directly, we always print to a file (you can of course set the location and name of the file in the *Print dialog*) which you will have to print out manually afterwards. If we had included some sort of piping mechanism to the default printer, we would have lost platform-independence.

Of course, the `Print` command always prints out exactly what you see on the drawing area, respecting the current zoom level etc.

3.8 Viewing

You can zoom in and out of the Drawing Area in different ways: The menu item `Zoom In` performs a zoom by a constant factor, centering the click point in the new view, `Zoom Out` zooms back out by the same constant factor, the `Reset Zoom` item zooms back to the default zoom level, and with the `Set Zoom` entry any select box dragged open on the drawing area is instantly expanded to that it fills the whole screen. However, zooming does not change any of the items, it just changes the viewport!

3.9 Dialogs

All dialogs in the application are dockable, that means they can be glued to the border of the frame by dragging them close to one and will remain there when the main frame is moved around. Dragging them away from the border releases the lock. Note that for each dialog, the docking information can be added to your default configuration file by just saving the file before shutting down the DeViSoR. Your dialog settings will then be restored during next startup.

3.10 Help

An electronic version of this manual can be displayed via the `Manual` entry from the `Help` menu, or use any web browser to load the index.html file in the manual directory of your DeViSoR installation. So far, no context sensitive help has been implemented, we apologise for any inconvenience. A collection of frequently asked questions is also available through the same menu.

4 Defining geometry

In this chapter, we will first describe the theoretical background of geometry definition. Afterwards, the usage of the DeViSoR will be covered, and we end up with a little working example (the world famous DFG benchmark). For your convenience, this chapter closes with a list of the Do's and Dont's of geometry definition.

4.1 Background

A *domain* consists of both *geometry* (covered in this chapter) and *mesh* (covered in the next chapter). The whole *domain boundary* or *geometry* consists of several arbitrary *boundary components* or short *boundaries*, each of which must be closed in itself. Every boundary is orientated in such a fashion so that the "inside" of the domain is always left of the boundary. Thus, the outer boundary has mathematically positive orientation (counterclockwise) and all inner boundaries change from negative to positive back to negative orientation with increasing distance from the outer boundary.
Each boundary has a unique number through which it can be adressed, starting with 0 for the first boundary. Usually, each boundary consists of several *segments.* Each boundary has a quite natural parametrisation: Each segment has a parameter interval of length 1, the first segment from 0 inclusive to 1 exclusive, the next from 1 inclusive to 2 exclusive and so on. Thus, the maximal parameter value for each boundary is equal to its number of segments. As boundaries are closed, the maximal parameter value is again equal to the parameter value 0. Each segment has a unique index of two integer values, one for the boundary it belongs to, the other for the number it has inside that boundary. Both indices start with 1.
Segments can be of the following types:

Lines A line is defined by its start point and end point. The start point corresponds to the local parameter value 0, the end point to the local parameter value 1. Lines must not be points, in other words, the start point coordinates must be different from the end point coordinates.

Circles A circle is defined by its midpoint, its radius and two angles which define the arc. Any arc except 0 is possible, for example a half-circle starting at 90 degrees and ending at 270 degrees. All angles are measured in degrees against the three o'clock position (or the positive x-axis). Circles can have positive or negative orientation, if the start angle is less than the end angle, the orientation is positive, otherwise negative. The start point of the arc corresponds to the local parameter value 0, the end point of the arc to the local parameter value 1. Circles must not be points, in other words, the radius must not be 0 or negative, and the arc must be non-zero.

The parametrisation of each boundary must be continous, this means you must not connect two end points of two segments, but each start point must be identical to the end point of the segment before. It is also illegal to have "holes" in the parametrisation, in other words, segment number 1 must start at the end point of segment number 0, segment number 2 must start at the end point of segment number 1, and so on, up to the last segment, which must end at the start point of segment number 0 so that the boundary is closed.

4.2 Defining geometry using the DeViSoR

New boundaries New boundaries are added to the domain using the corresponding menu entry or the button in the toolbar. One boundary is always the *working boundary*, its number is displayed in the combo box in the toolbar. You can change the working boundary using this combo box or using the dialog shown via the corresponding menu item from the `Domain` menu. By default (when starting up), the domain is empty and therefore only contains the symbolic boundary with index "-1".

Adding segments Each new segment is always added as last segment to the current working boundary. So first thing to add segments, select the boundary you want to work on.
To add `Lines`, select the line entry from the combo box in the toolbar or via the `Domain` menu. The first click sets the start point of the line, and as visual feedback, a line is drawn from the start point to the mouse cursor position. The current length of the line is additionally displayed. The second click fixes the end point. An error message is displayed if start and end point are the same, otherwise, the line is added to the domain. For your convenience, the type `Multiline` can be used to add polygons. The end point of the line you just added will be used as start point for the next line. Note that a single right-click cancels multiline input, leaving all lines in the domain except the line just being added. If you prefer direct input, mark the corresponding entry in the menu. Any click on the drawing area results in displaying the *line dialog* where you can directly input the start and end point coordinates.
To add circles or arcs, select the `Circle+` or `Circle-` entry from the `Domain` menu or in the toolbar, depending on the type of orientation you want for the circle. The first click sets the center of the circle. Next, the radius is specified. A full circle is drawn from the center to the current mouse position. Clicking the second time fixes the radius. As visual feedback for the next step, a line is drawn from the center to the cursor to indicate the start angle which is fixed by the third click. Then, the end angle is fixed in the same manner, adding the circle to the domain with the last click. Note that a double click when setting the start angle results in a full circle. Here as well, the current radius and angles are displayed. For circles as well, direct input is provided: In direct mode, any click on the drawing area will display a dialog where you can specify radius, center coordinates and the angles.
The numbers next to the segments have two meanings: The first one is the segment number in the boundary the segment belongs to, the second number is the number of the boundary itself.

Deleting Boundaries When you are deleting segments from a boundary, the parametrisation and numbering of the remaining segments is automatically updated. Note that if you delete all segments in a boundary, the resulting empty boundary is deleted as well. This also applies to the cut and paste operation.

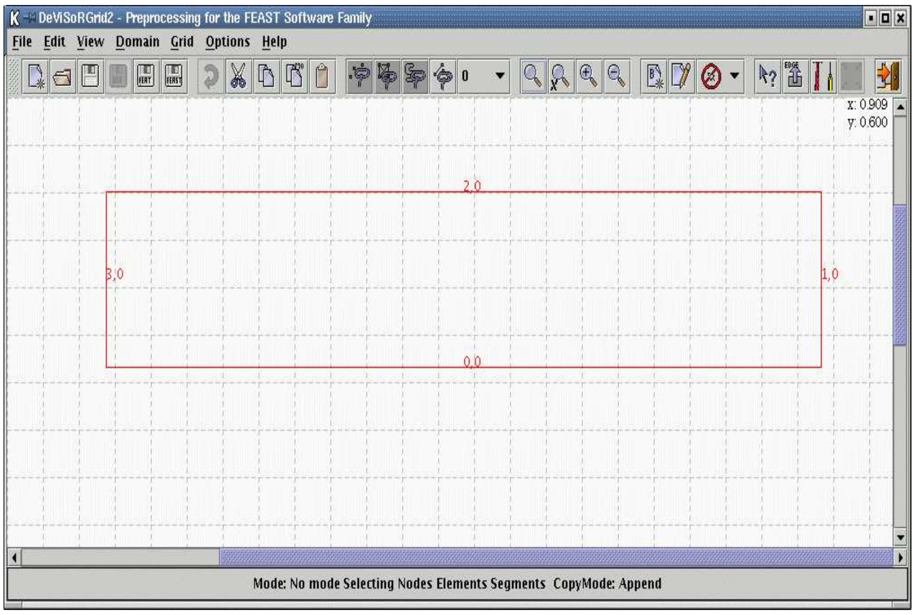
Fixing the parametrisation Sometimes it might happen that the parametrisation of a boundary is not correct (e.g. non-continuous), for example, if you use copy and paste to define a complex boundary. Afterwards, you should use the `Adjust Boundaries` operation, to be found in the `Domain` menu. Additionally, if you accidentally defined a boundary with the wrong orientation (for example, the outer boundary clockwise instead of counterclockwise), you can use the `Swap Boundary Orientation` operation from the `Domain` menu to flip the orientation of the current working boundary, in other words: The first segment becomes the last segment, the second segment becomes the secondlast, and so on.

4.3 An illustrated example

In this chapter, we will guide you through the definition of the geometry used for the DFG benchmark *Flow around a Circle*. More details on this can be found in the `Virtual Album of Fluid Motion` accessible on the web at `http://www.featflow.de`.
The geometry for this example is actually pretty simple: We only need two boundaries, the outer boundary is a large axis-parallel rectangle, the inner boundary a small circle centered in the left half of the rectangle. So let's start:

1. Start the DeViSoR if you have not already done so.
2. Set the domain perimeter to the rectangle from (0,0.5) to (2.5,-0.2) using the corresponding dialog.
3. Add a new boundary by clicking on the `New Boundary` button in the toolbar.
4. To define the outer boundary, select the `MultiLine` segment type from the combo box in the toolbar.

5. Start with the bottom left corner at coordinates (0,0), then the bottom right corner at coordinates (2.2,0), then the top right corner at coordinates (2.2,0.41), then the top left corner at coordinates (0,0.41) and close the rectangle with a line back to the bottom left corner. Your geometry should now look like this:


DeViSoRGrid2 - Preprocessing for the FEAST Software Family
File Edit View Domain Grid Options Help
0
x: 0.909
y: 0.600
2,0
3,0
1,0
0,0

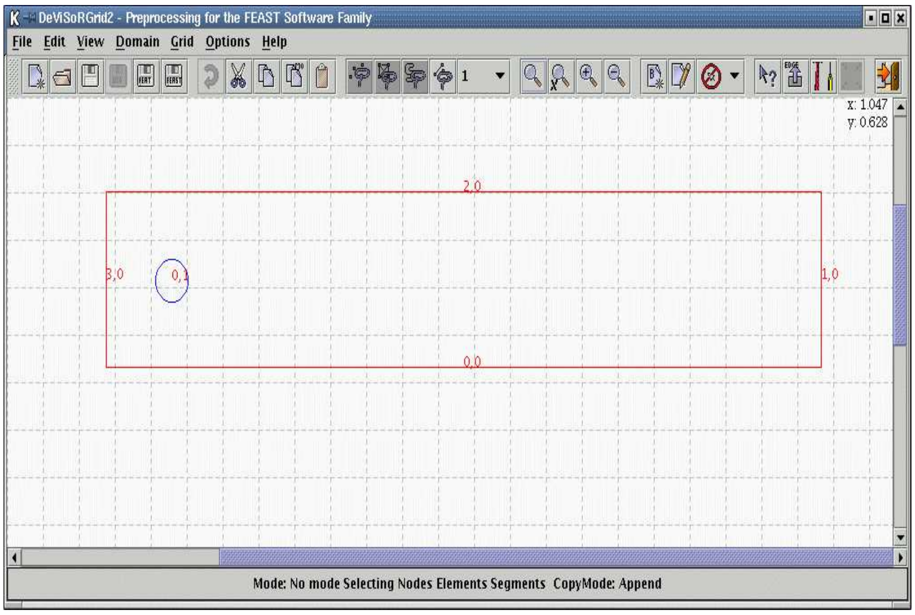
Mode: No mode Selecting Nodes Elements Segments CopyMode: Append

The outer boundary of the DGF benchmark

6. Add a second boundary to the domain, make sure it becomes the working boundary.

7. Select the `Circle-` segment type (remember that the inside of the domain is always left of the segments).

8. Add a full circle to the second boundary, centered in the left half of the rectangle at coordinates (0.2,0.2) with radius 0.05. You should now have a geometry like this:


DeViSoRGrid2 - Preprocessing for the FEAST Software Family
File Edit View Domain Grid Options Help
1
x: 1.047
y: 0.628
2,0
3,0
0,1
1,0
0,0

Mode: No mode Selecting Nodes Elements Segments CopyMode: Append

The complete boundary of the DGF benchmark

9. Save your file in FEAT format, because we will need it in the second workshop soon.

4.4 Remarks on geometry definition

When defining geometry, the following is essential for the correctness of your domain boundary:

- Every boundary has to be closed.
- Every boundary must have a continuous parametrisation.
- Every segment must start at the end point of the previous segment and end at the start point of the next segment.
- All boundaries must be orientated in such a fashion that the "inside" of the domain lies to the left of the boundaries.
- The outer boundary must have positive (counterclockwise) orientation.
- There can be only one outer boundary.

5 Defining coarse meshes

In this chapter, we will first describe the theoretical background of mesh definition. Afterwards, the usage of the DeViSoR will be covered, and we end up continuing the working example from the last chapter. For your convenience, this chapter closes with a list of the Do's and Dont's of mesh definition.

5.1 Background

The *mesh* of the domain describes the discretisation of the numerical simulation you are about to perform. The DeViSoR is used to manually define a first coarse mesh which will be automatically refined during the calculation.
Mesh elements have a natural hierarchy: The *finite elements* you can use are composed of *Edges*, which again are defined by two *Nodes*. Nodes are points in the plane. The DeViSoR supports the FEM elements *Triangle* and *Quadratic*, additionally providing the *Macro* type which is a Quadratic element with integrated anisotropic refinement and parallel computing information.
Usually, the first thing to define a mesh is to figure out where to place nodes. The general rule is to place a lot of the nodes and thus a lot of small-scale elements in those areas where you expect the most to happen, for example around the obstacles in flow simulations. In the other regions of your domain, a sparse node density and thus fewer large-scale elements will suffice. Then, you have to decide if you want to use Triangles or Quadratics as finite elements. Note that it is illegal to mix these elements, you have to restrict yourself to one type of finite element in one domain. Then, you connect every three or four nodes with the elements of your choice. You don't have to add edges explicitly, they are created implicitly. It is very important that the mesh covers the wohle inside of your domain, "holes" in the mesh will lead to a crash of the simulation during computing. Especially, elements must not intersect.
Additionally, the elements must not be degenerated, in other words, all nodes must be different, and any two edges of an element must not intersect. Neighbouring elements should not differ much in size, and the inner angles should not be too small or big. Also try to avoid high aspect ratios of your elements which will destabilize your simulation.
Nodes can be placed on the boundary segments, these *boundary nodes* can be manipulated via the parametrisation of the boundary and not via their cartesian coordinates like regular inner nodes.
All nodes, edges and elements are indexed starting to count from 1 upwards.

5.2 Defining meshes using the DeViSoR

Node input To add nodes to the domain, select the `Nodes` item from the combo box in the toolbar or in the `Domain` menu. For your convenience and for more accuracy, node input requires two clicks: The first click attaches a little node to the mouse cursor. Move it around on the drawing area to set the exact position of the new node. Note that if you are moving it over a segment, the cursor will change to indicate that you are about to add a boundary node instead of a regular one. In any case, the second click will add the node to the domain.
Alternatively, you can use the direct input dialog: activate direct input, click anywhere on the domain, and a dialog will appear which allows you to add new nodes either via their cartesian coordinates or by defining the parameter value and the boundary on which the node is going to be placed. If you hit the `Apply` or `OK` button when the `Cartesian` tab is selected in the dialog, a regular node will be created, otherwise, a boundary node. Note that it is impossible to add two nodes at the same position, You can try though, but the DeViSoR will notice and just not add the second one.

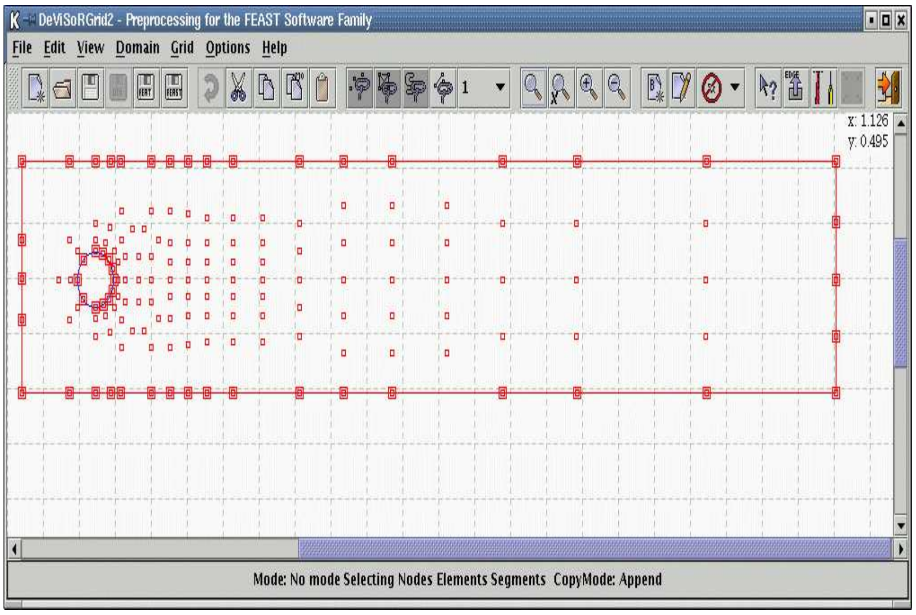
Element input To add elements, just select the type of element you want to add from the combo box or menu, and then just click on the nodes you want to define the element with. It is very important that you click on the nodes in the correct counterclockwise order. Alternatively, you can again use direct input to specify directly the nodes you want.

Deleting nodes or elements When deleting elements and nodes are not selectable, the nodes of the element are not deleted implicitly, but the edges are if they are not shared with other edges. When deleting segments and nodes are not selectable, any boundary node on these segments is transformed to a regular node. Note that this also applies to the cut and paste operation. But be careful because when pasting the selection again at the same position, these nodes will not be retransformed into boundary nodes!

5.3 An illustrated example

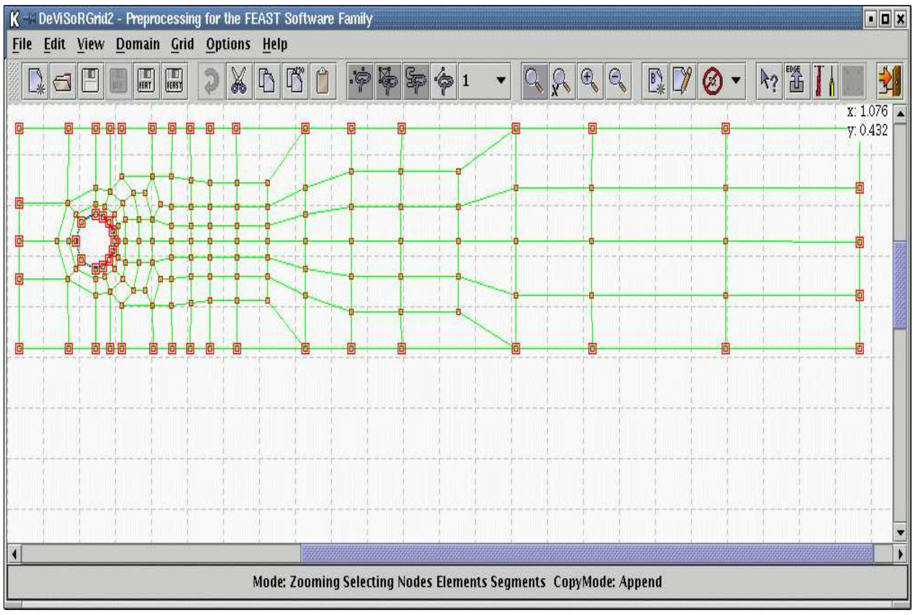
Now it is again time to go back to the good old DFG benchmark we started working on in the previous chapter. We will guide you through the process of defining a coarse mesh for the flow simulation, which is the last step in the preprocessing. After that, you can use the generated files to start the computation!

1. Load the geometry definition you saved in the last workshop.
2. Add boundary nodes and nodes according to the following image. Note that the node distribution represents the expected flow.


DeViSoRGrid2 - Preprocessing for the FEAST Software Family
File Edit View Domain Grid Options Help
x: 1.126
y: 0.495
Mode: No mode Selecting Nodes Elements Segments CopyMode: Append

Node distribution in the DFG benchmark

3. Add Quad elements according to the following image.


DeViSoRGrid2 - Preprocessing for the FEAST Software Family
File Edit View Domain Grid Options Help
x: 1.076
y: 0.432
Mode: Zooming Selecting Nodes Elements Segments CopyMode: Append

The complete domain for the DFG benchmark

4. To test your mesh, start a test run of the `TriGen` program by selecting `Refine` from the `Grid` menu and enter the following parameters in the dialog: ...

5. That's it, start the FEATFLOW simulation. And don't forget to save your work before you start.

5.4 Remarks on mesh definition

When defining meshes, the following is essential for the correctness of your grid:

- Do not mix element types.
- The nodes of an element must have counterclockwise orientation.
- Elements should not be degenerated: No extremal inner angles should occur, and large aspect ratios should be avoided.
- Neighbouring elements should not differ much in size.
- Generally, place most nodes where you expect interesting things to happen.
- It is always a good test to start a `TriGen` run on the mesh you created. If it does not crash, this indicates you created a feasible grid.

6 Additional FEAT features

6.1 Property queries

For your convenience, we added a property query mode. In this mode, to be activated through the `Edit` menu or the toolbar, a click on an item on the drawing area will show a dialog displaying all parameters of the item. While you are at it, you can directly alter these parameters.
Example: If you are not entirely satisfied with the position of a boundary node, just switch on property query mode, click on the node, and change its parameter value to something more satisfying.

6.2 Automated refinement

If you have a working FeatFlow installation, you can link the `TriGen2D` tool to the DeViSoR using the corresponding dialog. See the manual for details about the `TriGen` tool. You have to give the path to your Trigend2D installation in the options dialog in the general section. Further you have to copy the `feat2tri.f` and `trigen_wrapper` files to your trigen2d directory. Before using you have to compile the `feat2tri.f` with a command like this: `f77 feat2tri.f -o feat2tri`.

6.3 Fixing parametrisation

We have implemented all parametrisation-fixing in one big operation called `Adjust boundaries`, to be found in the `Domain` menu: All boundaries are processed, trying to make each boundary continous and trying to fix the parameter values of each boundary node. The debug output generated by these operations is displayed in a dialog box afterwards, and you can learn if the operation was a success.
If you have accidentally created boundaries with incorrect orientation, you can swap the orientation of the working boundary with the `Swap boundary orientation` menu item.

6.4 The SpellChecker

Sometimes it might happen that you have created such a massive grid that simply too much imformation is displayed to determine whether your grid is reasonably satisfactory in quality. So, we implemented the SpellChecker and added it to the DeViSoR: Just select the corresponding menu item to start it. A dialog will show up, which presents two tabbed sheets: On the first sheet, you will find a table and some buttons, on the second sheet you will find a checklist of whatever quality criteria you want to be checked. So far, the following ones are supported:

- Isolated nodes: This is pretty self-explanatory, the DeViSoR will just list all nodes that are not connected to an element.
- Angles: In the checklist, you can define a threshold value. The DeViSoR will then list all elements with inner angles smaller than that threshold angle. The idea is that numerical code tends to crash on elements with very small inner angles and thus very steep corners.
- Aspect Ratios: The same applies to elements with a very big aspect ratio, that is the quotient of shortest and longest edge of the element. You can of course again specify a threshold ratio in the checklist.
- Neighbouring Elements: Also for numerical stability reasons, it is advisable that neighbouring elements should not differ much in size and area.

After completing the checklist tab, just switch back to the table tab sheet and hit the REFRESH button. The table will be updated according to your specifications in the checklist. You can now select any entry from the list and zoom to it by hitting the ZOOM button, thus starting to improve the quality of your grid. Hit REFRESH again to check if your modification was correct, or just hit REMOVE to delete the selected item from the list.

7 Additional FEAST features

The FEAST format is not entirely supported so far, so we just give a short list of what you can expect for the next release:

Macros Macros are an extension to Quadratic elements which allow for anisotropic refinement. Additionally, macros can be combined into parallel blocks to increase efficiency on parallel computers.

Enhanced Edges You will be able to assign stati to edges, and of course boundary conditions to boundary edges.

Rectification Macros can automatically be transformed into rectangular proportions to increase the stability of the numerical computation.

A Licence Agreement

B References

- DeViSoR1.0 application and manual, on the web at `http://www.featflow.de`
- The FEAT group and FEATFLOW, also at `http://www.featflow.de`
- The FEAST group, non-surprisingly at `http://www.featflow.de`
- The Dialog Model In The DeViSoRGrid2 Application, part of the installation and located in the *manual/specs* directory
- The Undo Implementation In The DeViSoRGrid Application, located in the *manual/specs* directory of your installation