Copyright notice Top Next

DeVIDE is copyright (c) 2002-2008 Charl P. Botha, TU Delft

All rights reserved.

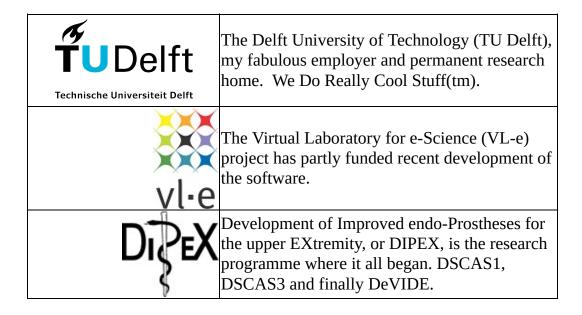
Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- · Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the TU Delft nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Sponsors Previous Top Next

The development of this software was supported by the following institutions:



Introduction Previous Top Next

This is the user manual and help file for the DeVIDE software package. DeVIDE is the open source Delft Visualisation and Image processing Development Environment.

The software was designed for the rapid prototyping and testing of visualisation and image processing techniques. All functionality is packaged in the form of modules. One of the ways in which these modules can be used, is to make use of the DeVIDE Graph Editor, where glyphs, representing modules, can be connected up to form functional data-processing networks. A key feature is that any code object can be accessed and interacted with in real- time. This enables both module writers and users of the Graph Editor to experiment with for example all parameters and even the logic behind a running piece of functionality.

DeVIDE is available as open source under the terms of the new BSD license. Binaries are usually made available for the Windows and Linux platforms.

The main website of this software is at http://visualisation.tudelft.nl/Projects/DeVIDE . Please go there for links to more documentation, the GoogleCode project site, the mailing list and the news blog.

This manual is in a perpetual state of being under heavy construction. The <u>Graph</u> <u>Editor</u> and <u>Useful Patterns</u> sections are especially useful and should be read before you start working with the software.

Introduction Previous Top Next

The DeVIDE Graph Editor is a visual programming interface where glyphs representing the underlying DeVIDE modules can be connected together to form new programs.

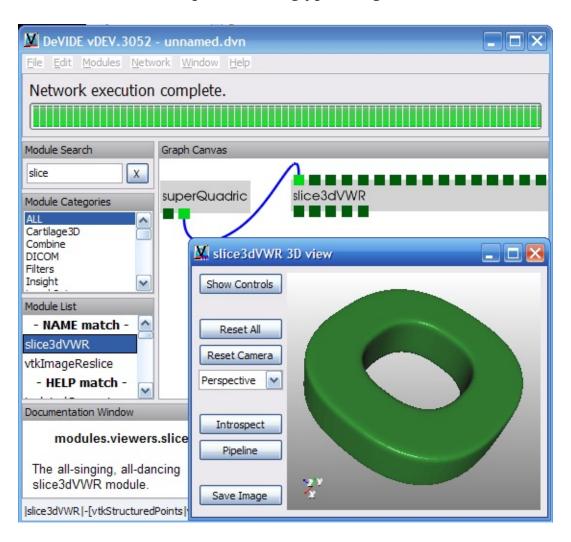
It's the most flexible way of working with DeVIDE, short of directly interfacing with the underlying code.

This chapter will give a brief overview of graph editor usage.

The Graph Editor is introduced by putting together a simple network for rendering the 3D 0- surface of a Super Quadric. It's far simpler than it sounds.

Constructing the network

Start by building the network shown in the figure below. First select the "Sources" category on the top left of the Graph Editor. Drag and drop the "superQuadric" module from the modules list on the bottom left to the canvas. You should see the "superQuadric" glyph being created.



Now do the same for the "slice3dVWR" module in the "Viewers" category. Note that you can select multiple categories by holding the Shift key and clicking a category (this will select all categories between the previous selected category

and your current click) or holding the control key and clicking (this will select the currently clicked module along with any previously selected modules). The module list will contain all modules in all selected categories. The modules are always alphabetically sorted.

Connect the second output of the "superQuadric" glyph to any input of the "slice3dVWR" glyph by dragging the mouse, with the left button depressed, from the output port to the input port. Note that hovering the mouse pointer over any port shows more information about that port in the status bar of the Graph Editor.

Admiring your results

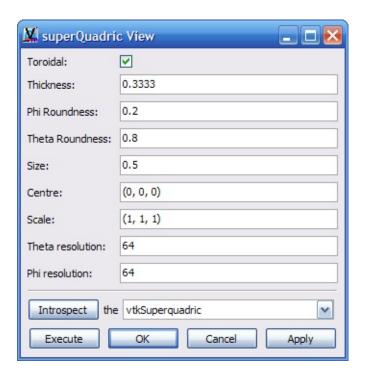
Now press F5 or select Network | Execute from the main menubar to execute the network.

Right-click on the slice3dVWR module and select "View-Configure" to see the 3D surface representing the 0-surface of the generated Super Quadric. Note that this is how one activates the graphical interface of any glyph on the canvas. You can rotate your viewpoint around the generated 3D object by dragging with your left mouse button. Dragging with the right button will zoom. Dragging with the middle button will pan the viewpoint.

The network can be saved by selecting "Save" from the "File" menu. The default extension for a DeVIDE network is .dvn.

Warping the Super Quadric

The slice3dVWR is a very special DeVIDE module. Because of this, its View/Config interface is non-standard. Right click on the "superQuadric" glyph and select "View-Config" to see a more standard user interface (you can also just double-click on the glyph, many users find this to be quicker). This interface is shown in the figure below:



Most module View/Config windows have the set of buttons at the bottom. If you make any changes to any of the module parameters, you have to click on the "Apply" button (in which case the parameters will be transferred to the underlying logic) or the "Execute" button, (in which case the new parameters will be transferred to the underlying logic and the module will be asked to reperform its execution with the modified parameters). Clicking on the "Cancel" button will undo any changes you've made without applying and close the window. Clicking on "OK" will apply all changes and close the window.

Change the "Phi Roundness" parameter to 3.0 and click on "Execute" or simply press F5 or Alt-X. Pressing one of these key-combinations is effectively the same as clicking on "Execute". See the results of your changes in the slice3dVWR window.

Before we start, a few productivity-enhancing secrets of the Graph Editor are revealed here.

Placing modules

You can place modules on the canvas in more than one way.

- Drag and drop the module name from the module list on the bottom-left onto the canvas. You can narrow down the list of modules by selecting one or more categories. The "ALL" category includes all modules available.
- · Double clicking on a module name will place that module on the canvas.
- Typing in the module search box on the upper left and then pressing the ENTER-key will place the first automatically selected module in the search results on the canvas. See the next section for more details on this.

Placing modules quickly

As mentioned above, modules can be placed by typing in search phrases in the module search box in the upper left of the main DeVIDE window. Only modules in the currently selected categories will be found, therefore it is recommended that you select the "ALL" category when making use of this functionality.

You can quickly go to the search box by using the Ctrl-F hotkeys. Typing any search terms will show search results in real-time in the module list box. Modules are searched by examining their names, keywords and the module-specific documentation. At all times, a module will be automatically selected in the search results. You can change the selection by using the up and down cursors. Pressing the ENTER key at any time will place that module on the canvas.

Getting module-specific help

Many modules have module-specific help-text built in. To see this help, just select this module from the module list at the bottom left of the main DeVIDE window. The module help will appear in the module documentation window below the module list.

Also remember that selected (special) modules are documented in the chapter Special Modules.

Reading data, quickly

Dragging and dropping certain data-files on the Graph Editor canvas will cause the system to automatically create and configure the applicable module. For example: selecting, dragging and dropping a collection of DICOM .dcm files on the canvas will result in a dicomRDR glyph to be created and pre-configured with the list of files that has been dropped.

Following the same theme, dragging and dropping a filename (or a collection of filenames) on a module that has a filename attribute (such as most file readers) will set that file to be read on the next network execute. This is useful when you have a single network that you want to apply on a number of different datasets: just drag the new filename onto the reader in question and execute!

Re-using networks, quickly

Dragging and dropping a .dvn (DeVIDE Network) file on the canvas will instantly load the network and build it at the mouse position. This will not destroy any of your current networks on the canvas.

Introduction Previous Top Next

This chapter has more documentation on some of the more complex modules in DeVIDE. Remember that all modules have module-specific documentation, available by right clicking on the module glyph and selecting "Help on Module" or by querying the doc-string of the main module class.

slice3dVWR Previous Top Next

Introduction

This is probably the module you'll use most often. It's also the first thing you should try whenever you have any data you'd like to visualise. Simply connect the output of any data- generating or filtering module to any input of the slice3dVWR and execute the network. The slice3dVWR picks a suitable default visualisation based on the input data you supply.

Slices

When a volume has been connected to an input of the slice3dVWR, the volume can be examined by "slicing" through it. By default, an axial slice is created automatically, but any number of additional slices can be activated.

Overlay modes

The slice3dVWR has several overlay modes. These modes make it possible to visualise the correspondence between multiple inputs, for e.g. original CT data and a segmentation. If a second volume input is connected, the slice3dVWR checks if the dimensions of the already connected volume. If this is the case, the connection is allowed and the second volume is overlayed (superimposed) on the first.

There are several ways to perform this overlay. This setting is user-configurable and its user interface can be found on the "Main" tab of the slice3dVWR "Controls" window, in the "Slices" section. The user interface consists of an Overlay Mode choice box and an Alpha slider. The alpha slider determines the alpha parameter used for the fusion-based overlay modes.

- Green Fusion: The overlay is composited with the user-defined alpha parameter. The value (i.e. brightness) is directly related to the image intensity of the overlay, so we see shades of green (reflecting the overlay intensity) alpha blended with the primary input.
- · Red Fusion: Same as above, except with shades of red.
- · Blue Fusion: Same as above, except with shades of blue.
- Hue Fusion: The value is kept constant, but the hue is directly related to the overlay image intensity. The overlay is alpha blended with the user-supplied alpha parameter.
- · Hue/Value Fusion: Hue and brightness are directly related to the overlay

image intensity. The overlay is alpha-blended with the user-supplied alpha parameter.

- · Green Opacity Range: The opacity of the overlay is directly related to its image intensity. The hue is constant green and the brightness is constant unity.
- · Blue Opacity Range: The same as above, except the hue is constant red.
- · Blue Opacity Range: The same as above, except the hue is constant blue.
- Hue Opacity Range: The hue and the opacity of the overlay are directly related to the overlay image intensity.

Adjusting the alpha slider whilst one of the "fusion" overlay modes is active will result in real- time changes. The idea is to adjust it up and down its complete range in order to get a better idea of the amount of the image correspondence.

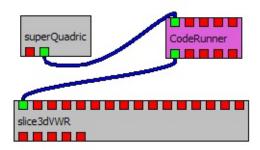
CodeRunner Previous Top Next

Introduction

With the CodeRunner, arbitrary snippets of Python code, using any of the libraries shipping with DeVIDE, can be inserted into a functional network. This is a pretty powerful concept that enables *very* rapid prototyping of new ideas or quick implementation of processing solutions.

An example

We'll illustrate with an example. First build up this small network:



Double click on the CodeRunner, causing its View to appear. The View's tophalf consists of an editor component with three tabs: Scratch, Setup and Execute. Paste the block of code below into the Setup tab:

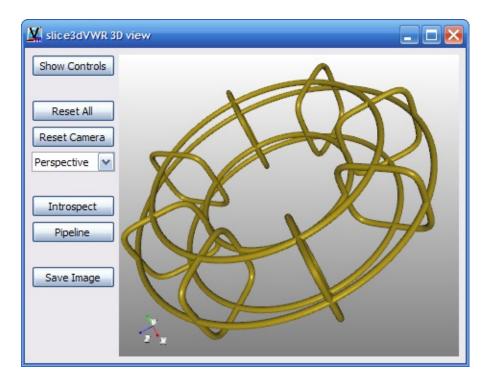
this block goes into the "Setup" tab of the CodeRunner
import vtk
fe = vtk.vtkFeatureEdges()
tf = vtk.vtkTubeFilter()
tf.SetRadius(0.01)
tf.SetNumberOfSides(16)
tf.SetInputConnection(fe.GetOutputPort())

... and the following block into the "Execute" tab:

this block goes into the "Execute" tab of the CodeRunner fe.SetInput(obj.inputs[0]) tf.Update()

obj.outputs[0] = tf.GetOutput()

Now click on the "Execute" button at the bottom of the CodeRunner View. In your slice3dVWR, you should see something looking more or less like the figure below:



Pretty slick eh?

Working with Scratch, Setup and Execute tabs

Using CodeRunner modules, you can insert arbitrary code segments into your DeVIDE networks. The "Scratch" tab is for experimentation, the "Setup" tab code runs ONCE for any changes that you make (you should use this for building pipelines for example), and the "Execute" tab runs every time the network is executed and the CodeRunner's inputs have changed or any of the CodeRunner tabs have been modified.

Whilst editing any of the tabs, press Control-Enter to execute just that tab at that moment. Clicking on the "Execute" button at the bottom of the View requests the whole network to be run. Liberal use of Control-Enter whilst editing will make sure that all Python objects that you have created will be instantiated. Besides the fact that code-completion and inline documentation can then do its thing, you

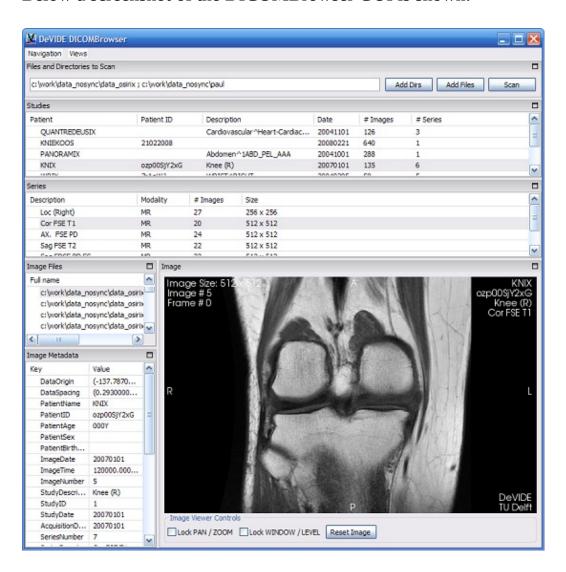
can easily experiment at run-time.

DICOMBrowser Previous Top Next

Introduction

With the DeVIDE DICOMBrowser, released with 8.5, you can easily explore collections of DICOM data. It is intended to be used as a visual interface with which DICOM series can be easily selected and loaded, using the companion DICOMReader module, into DeVIDE for further processing.

Below a screenshot of the DICOMBrowser GUI is shown:



Also see our introductory screencast on youtube: http://www.youtube.com/watch?v=iLfu6JXkWP4

Starting the exploration

Enter any number of directories and or filenames into the "Files and Directories to Scan" text input box, separated by semicolons. You can either type these in yourself, cut and paste them from some other application, or make use of the "Add Dirs" and "Add Files" dialog buttons.

After having entered all paths that you wish to scan, click on the "Scan" button. Post 8.5 versions of the DICOMBrowser will show a progress bar whilst scanning.

When scanning has been completed, all other panels in the interface will be filled out with information.

Exploring your data

DICOM data is divided up into studies, where each study is associated with a patient, a number of series per study, and a number of images per series. The DICOMBrowser allows you to select any of these elements by clicking in the relevant panel.

Once you've selected a specific series, which can usually be seen as a single data volume, you can browse through the images in that series by clicking on the relevant file in the "Image Files" panel, or by making use of the Ctrl-N Ctrl-P hotkeys for next image and previous image, or by clicking on the image and then using the mousewheel to move, 1 image at a time, through the series. Control-Mousewheel will skip 10 images at a time and allows rapid scrolling through a series.

Loading data for further processing

Once you have identified an interesting series, you can create a DICOMReader module on the DeVIDE canvas, and then drag directly from the series item or from a selection of filenames onto the DICOMReader. The DICOMReader will now be configured with the correct filenames and can be executed to load the data for further processing by DeVIDE networks.

You can also drag a series or selection of files onto most file manager windows (explorer on Windows, nautilus on Gnome) to copy those files into the destination directory.

Shortcuts

Ctrl-N, mousewheel down on image	Next image in series
Ctrl-P, mousewheel up on image	Previous image in series
Ctrl-mousewheel on image	Skip 10 images
Ctrl-0, Ctrl-1	Change view layouts (default, max image)

Introduction Previous Top Next

This chapter contains a selection of howtos, i.e. very short descriptions of how to perform some action with the DeVIDE software.

The easiest way to load a DICOM dataset is to select the relevant DICOM files (often with a .dcm extension) in your file browser (e.g. Explorer in Windows and Konqueror in KDE) and to drag and drop the selected files on the Graph Editor Canvas. A *DICOMReader* module will be automatically created with the selected filenames added.

Alternatively, you can create a *DICOMReader* module in the normal way, i.e. by dragging and dropping the module from the Module Palette onto the canvas. Double click on the module to open its View/Config window. Click on the *Add Files* button to add any number of DICOM filenames. Remember to click on *Apply* after having added the relevant filenames. Alternatively drag and drop a number of DICOM files onto the listbox, or directly onto the glyph on the canvas. This is often far easier.

To start visualising the volume that you have just loaded, create a *slice3dVWR* module and then connect the output of the *DICOMReader* module to any input of the *slice3dVWR* module. The module interface will automatically appear with the default axial slice. To read more about interacting with this slice, see the slices section of the <u>slice3dVWR</u> documentation.

For a far more pleasant DICOM browsing experience, try the *DICOMBrowser*. A screencast demonstrating this functionality is available at http://www.youtube.com/watch?v=iLfu6JXkWP4.

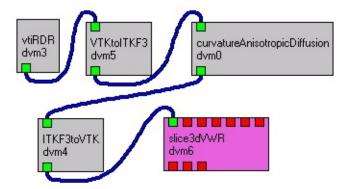
Smoothing data <u>Previous Top Next</u>

There are a number of filters in DeVIDE that can be used for smoothing volume data.

Filters.imageGaussianSmooth performs a straight-forward Gaussian smoothing (also known as "blurring" in some image processing packages). The standard deviation (in pixels) can be set for all three dimensions. A truncation, or cut-off, can also be set for all three dimensions. Take into account the resolution of your image when selecting these parameters.

curvatureFlowDenoising, curvatureAnisotropicDiffusion and gradientAnisotropicDiffusion, all in the "Insight" module category, are more advanced smoothing algorithms that attempt to smooth homogeneous regions whilst retaining edge information. These are all compute- intensive ITK-based filters. Please read the tooltips available in the configuration windows: i.e. double click on the module and then let your mouse hover over any of the input boxes to get more information about the variable required for that input box. The defaults are naturally good values to start with.

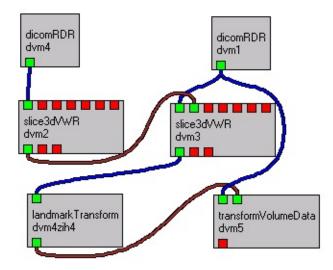
When you use these with VTK data, for example the output of a vtiRDR, you have to use a VTKtoITKF3 conversion module. To visualisation the output, you need to convert back to VTK data by making use of an ITKF3toVTK module. However, if you're planning to use the output in an ITK filter, for example the demonsRegistration, you don't need an ITKF3toVTK conversion module at the output. The figure below shows an example of this.



In the MRI retrobulbar fat mobility study, multiple MRI datasets are made of a subject during different directions of gaze. Although great care is taken to prevent rigid head motion, this does still occur. So before the deformation of the fat is calculated, rigid head motion has to be eliminated by means of a landmark-based rigid registration.

One of the directions of gaze is chosen as the central or reference direction: all other datasets have to be registered onto this dataset. In this way, all datasets will share a common frame of reference.

In the next subsections, we will explain how to perform one such registration. This obviously has to be performed for all datasets that you have to register onto the reference dataset. The figure below shows an example DeVIDE network for performing this landmark-based rigid registration. Refer to it during the following explanation.



Select source and target points

Select at least 3, preferably more rigid landmarks that can be accurately localised in all datasets.

Select these points in the reference datasets by using the mouse cursor in a 3D slice3dVWR. You have to name these points (in the slicedVWR control panel,

enter the name into the "name" input box before clicking on the "Store this point" button. The names of these points have to start with "Target", for example "Target Zygoma 1".

Save your network regularly!

Now load in the first dataset that you want to register onto the reference dataset. Select and store all corresponding points in this dataset. Use a separate slice3dVWR. The names of these points all have to start with "Source", for example "Source Zygoma 1".

It is very important that you select these points in the same order as the target points.

Derive the transform

Instantiate a "landmarkTransform" module from the Filters category. Read its help by right- clicking on the module and selecting "Help on Module".

Briefly connect and disconnect the first output of the reference dataset "slice3dVWR" to any input of the second "slice3dVWR". This will copy the "Target" points to the second "slice3dVWR".

Now connect the first output of the second "slice3dVWR" to the input of the "landmarkTransform" module. This module will now have access to the source and target points of the two datasets that you are trying to register. As mentioned earlier, the order of the source points should be identical to that of the target points.

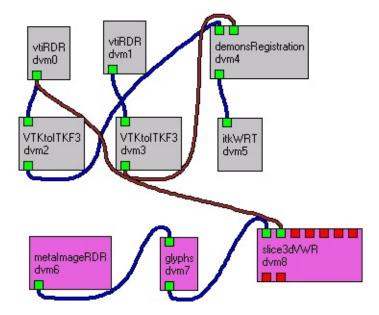
Transform the dataset

Connect the output of the "landmarkTransform" to the second input of a "transformVolumeData" module (category "Filters"). Connect the dataset that you are registering onto the reference dataset to the first input of the "transformVolumeData" module. The output of this module will be the transformed volume dataset. You can add it to one of the reference dataset's slice3dVWR's inputs for an overlay. Select the "Primary LUT fusion" overlay mode in the slice3dVWR control panel. The Alpha-parameter can be adjusted to "fade" from the reference dataset to the registered dataset and back.

Don't forget to save your transformed dataset: Connect the output of the "transformVolume" module to the input of a "vtiWRT" module for example. Double click on the "vtkWRT" module and then click on the browse button to select a filename. For all writer modules, you HAVE to click on the "Execute" button for it to save the file.

In order to determine the optical flow between two rigidly registered datasets, we make use of the demonsRegistration non-rigid registration module. Do this after the rpgid landmark-based pre-registration and after having extracted a common sub-volume from all datasets.

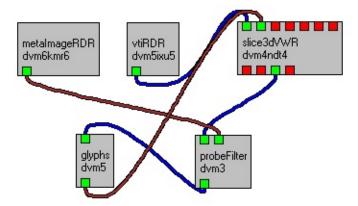
This is an ITK module, so you have to use the ITK-enabled version of DeVIDE. For each pair of successive volumes, determine the optical flow by making use of a network like the one shown in the figure below:



Text to be completed...

If a vector dataset is available, such as the one generated by the optical flow procedure described above, the deformation vector field can be visualised by making use of the Filters|glyphs module. However, this module visualises the complete vector field.

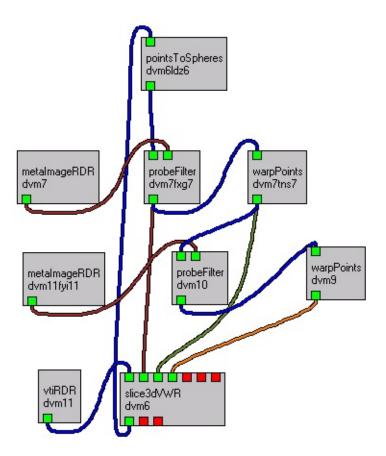
One can also visualise only the vectors on the current slices in the slice3dVWR. To do this, build a network as shown below:



The slice3dVWR outputs a poly data representing the geometry of all current planes. In this case, we use it as a probe input so that we can visualise arrow glyphs located on the plane.

Remember that the opacity of slices can be adjusted (Slices menu: "Set Opacity") so that visualising a background slice along with the deformation vectors on it is easier.

The figure below shows how to perform interactive probing and warping (advection) with DeVIDE:



The volume data has been loaded with vtkRDR dvm11 and serves as context in the slice3dVWR. In this case, we are using two vector fields, loaded with respectively metaImageRDR dvm7 and dvm11fyi11. Points that have been selected and stored with the slice3dVWR (these can be changed at any time, the network updates interactively) are changed into spherical shells of points by the pointsToSpheres module dvm6ldz6. The output of this module can be connected to the slice3dVWR to view the initial sphere.

The first probeFilter dvm7fxg7 maps and interpolates the vectors from the first vector field onto the points of the spherical cloud. Connect the output of the probeFilter to the warpPoints module. After connecting the warpPoints input, make sure to "Execute" and "Apply" the warpPoints module once. The correct "Vectors selection" can then be made from its View/Config window. "Default

Active Vectors" is very often NOT what you want. Experiment with this. (Thijs en Ronald, in julie geval moet dit "ImageFile" zijn!). After having made this selection, "Apply" or "Execute" the module again. Now you can connect its output to the slice3dVWR. The points are advected by the vectors that you have associated to them.

This output is also used in the next advection step, using the next vector field. Once again, make sure to select the correct vectors in the warpPoints module View/Config window. You could also simply copy and paste the first probeFilter and warpPoints module (AFTER you've performed the vector selection step), in which case the selection will already be correct.

Add as many of these steps as are necessary to complete the full advection with all your available vector fields. At each step, check that your sphere is being correctly advected before you continue.

Introduction Previous Top

This chapter will explain the various ways in which you can extend DeVIDE's functionality. This mostly happens via the creation of new DeVIDE modules. There are various types of modules that one can create, this chapter documents most of 'em.

Other possibilities for extending DeVIDE include writing Snippets (a kind of poor man's plugins) or simply making use of the <u>CodeRunner</u> module. The latter is recommended if you want to test your idea as quickly as possible and can't be bothered with reading manuals. Making a new module, although taking slightly more time, has its advantages, such as more flexibility, easier distribution, and creating DeVIDE MiniApps!