Programs for Needle Geometries

Rob MacLeod

June 23, 2007

1 Introduction

The programs described in this document all serve one purpose—to make management of needle geometry as easy as possible. They take care of all the steps required to convert digitized rod entry points and rod tips into geometry files, with a few additional "goodies" for fun.

The steps required to manage needles are as follows:

1. Either a) remove the needles from the heart and replace each with a rod, making sure to keep track of the needle number, or b) leave needles in place.
2. If a), digitize the locations of all rod tips and the rod entry points; if b) digitize the locations of the ends of the needles, at the point where the wires connect, and at the entry point of the needle.
3. Convert the resulting file to a landmarks file with *digtoxyz* (See Section 2). Optionally, view the landmarks file in the context of the surface geometry, if it is available, with *map3d*.
4. Convert the resulting landmarks file to a geometry file using *needlegeom* (see Section 3) and view the results again with *map3d*.
5. Digitize the innermost surface of the geometry using *map3d*.
6. Propagate this triangularization to all other surfaces in the geometry using *needletri* (see Section 4)
7. View the result with *map3d*
8. If you wish to have subsets of needles from the full geometry, use the program *makeplanes* (see Section 5).

The programs are not perfect, nor is the documentation, but we can make it that way with your help. Let Rob know what is not clear or what fails to perform as required and we will aim to please!

2 Processing digitizer files: *digtoxyz*

The format of data that comes from the digitizer is not directly usable by any of the standard CVRTI programs and must be converted before anything else happens with is. The program *digtoxyz* is one program for this conversion. It takes digitized datasets in a variety of formats and tries to identify what the contents are, then converts and saves them in the appropriate manner. The reason for the complexity lies in the diversity of things we digitize, from sock electrodes points to the tips and rods, and the need to treat each one a little differently. A further complexity lies in the two different devices we use to digitize; on good days this program handles both with grace and beauty.

Below we list each of the different types of geometrical objects or information that *digtoxyz* knows about and what type of input file format is necessary for consistent interpretation. I have removed all references to what used to be called the "old digitizer", this was the one built at the CVRTI that fixes the object on a rotating jig and moves the caliper to touch the points. We no longer use nor support this device. In contrast, the "Microscribe digitizer" is the device we have used for 10 years now. We create files from this device using Phil's program on the Macintosh. The standard extension for these files is ".ms".

2.1 Digitizing Tips

There is no magic nor strict paradigm for digitizing with the Microscribe; on the other hand, a few lessons from experience can be useful:

1. Always try and perform the entire digitizing pass with the same settings and zero marks on the digitizer. Resetting the digitizer also re-zeros the origin and so locations digitized with one reference angle cannot be merged with those digitized with a different one. If this happens accidently, then try and at least re-digitize something in the new reference frame that has already been sampled before under the original reference and **make sure to tell someone, ideally Rob, that this has occurred.**

2. Try and save output files from the digitizer with the extension `.ms`. This way, there is less ambiguity about the contents and how we need to read them.

3. The largest source of true error in digitizing seems to be the movement of the heart during the digitization process; the Microscribe should improve this because the object can be kept stationary while the sampling probe moves around it. However, reducing movement remains the most straightforward means of improving accuracy.

4. If in doubt, collect the data, convert it immediately, and then view the results with *map3d*3d. If still in doubt, digitize it a second time. If still in doubt, ask for help! It's OK to be in doubt, but silly not to do something about it.

2.1.1 Order of digitizing

The program *digtoxyz* does not explicitly care about the order of items digitized, nor which itens are included in a single .ms file. However, a little planning in this regard can save time in the subsequenct processing.

Sock Points: In digitizing a set of points from, for example, an epicardial electrode sock, the order of entry of the points in the .ms file determines the order the point will have in the .pts file that *digtoxyz* produces. So **order is important here!**.

Coronaries: At present these are also read and converted into landmarks in the order in which they are stored in the .ms file. The numbering scheme in the input file establishes the layout of the landmarks file so order is only semi-critical here.

Recording needles: For digitizing recording needles (via the rods), *digtoxyz* keeps track of the rod/needle number used in the `.ms` file and saves this information in the `.lmarks` file that results. Thus it is feasible to digitize in an order that is spatially coherent or just convenient and then have *digtoxyz* sort things out. In the subsequent needle geometry processing steps described in this manual, the needle info file should also be organized according to the same needle numbering scheme as the `.ms` file, but need not have any particular order. The software takes care of matching according to needle/rod number as they are labelled in the `.ms` file.

Crucial in the entry of needle/rod points is the order in which the points of each needle/rod get digitized. The convention as we have agreed to it so far is **outer end = rod tip, then inner end = rod entry point**.

2.2 Running *digtoxyz*

To run the program *digtoxyz* requires the following command line:

```
digtoxyz [-m -c] infilename outfilename
```

where

-m = metric units on old digitizer

-c = confused units on old digitizer (radius in inches, Z in decimeters)

infilename = name of the input digitizer file.

outfilename = name of the output file, whose type depend on the contents of the input file; there may even be several output files that result from a single input file.

Note: New digitizer includes the units in the file while the old digitizer defaults to inches for units.

2.3 Sock electrodes

Sock electrode locations are read as a series of points, and then later triangulated and formed into geometric meshes that form the basis for visualization, computer models and other types of data analysis. Hence the role of *digtoxyz* is fairly simple—convert each point from digitizer format to a geometry file (either .geom or .pts/.fac files).

2.3.1 Digitizer data format

Header Lines (4 in all)			
FILE NAME: *filename* LAST MODIFIED: *Date and Time* *Free line of text* "Measurements are in" *millimeters or inches*) **********			
File Body			
"Point" #	X-value	Y-value	Z-value
Example			
zero	0.000	0.000	0.000
sock 1	28.508	-173.895	221.950
sock 2	20.061	-178.593	225.062
sock 3	11.862	-179.161	237.611
sock 4	5.729	-175.170	251.998

2.3.2 Output files

The result of running *digtoxyz* on epicardial points is a file that is part of the CVRTI geometry files. Its extensions is .pts and it contains a list of all the points, converted into Cartesian coordinates and units of millimeters.

2.4 Coronaries

These are the files that result from digitizing the paths of the coronary arteries, which needs to be done according to the following tips:

- Always start at the "top" of each major artery and work down to the tip or end of the vessel. Order the segments according to the natural heirarchy of the coronary tree, taking left main first, then LAD, then its branches, *etc.,*.
- After the main path is digitized, return to the branch nearest the top of the vessel and trace it down to its end; always try and digitize a point at the fork in the tree before following the new branch to its end–this point will sometimes be "doubled" is that it could belong to the "parent" segment and also to the "offspring" branch but this is fine.
- Repeat the last two steps recursively for every visible branch in the tree. When one major artery (*eg.,* left anterior descending) is complete, move to the next major artery and repeat.
- Note that you can easily edit the resulting coronaries in the old map3d, now called map3dgl. The new version will have this capability sometime too. To do this:
  1. select "Pick landmark point" in the "Select Pick Action" menu
  2. Then when you select a point from a landmark, the three translation dials will move the point in 3D space
  3. The scaling dial will adjst the radius value of the point.
  4. The rotation dials still rotate the whole object so you can move it around to get the best perspective and then control the movement of the individual landmark points.

2.4.1 Digitizer data format

There are two slightly different file layouts that *digtoxyz* will recognize as being from a coronary artery tree. One contains extra information of a radius for each point in the tree. This eventually determines the thickness of the tubes drawn in *map3d* and if fixed at the time of digitization, then it supplies one set of values that *digtoxyz* otherwise requires from the user when it runs.

The first format includes only the coordinates of each artery point.

Header Lines (4 in all)			
FILE NAME: *filename* LAST MODIFIED: *Date and Time*			
Free line of text			
"Measurements are in" *millimeters or inches*)			
**********			
File Body			
"Point" #	X-value	Y-value	Z-value
Example			
Artery 1-1	28.508	-173.895	221.950
Artery 1-2	-.9385	4.8480	225.062
Artery 1-3	-1.0225	4.5970	237.611
Artery 1-4	-1.1590	4.4565	251.998
Artery 1-5	-1.2280	4.2110	221.950
Artery 1-6	-1.2645	3.9330	225.062
Artery 1-7	-1.2510	3.6935	237.611
Artery 1-8	-1.2050	3.5000	251.998
Artery 2-1	-.9345	4.8945	221.950
Artery 2-2	-.8620	4.5170	225.062
Artery 2-3	-.7850	4.2745	237.611
Artery 2-4	-.6610	3.9005	251.998
Artery 2-5	-.5445	3.6085	221.950

The second format includes a radius value for each point, something one must typically add manually.

Header Lines (4 in all)				
FILE NAME: *filename* LAST MODIFIED: *Date and Time*				
Free line of text				
"Measurements are in" *millimeters or inches*)				
**********				
File Body				
"Point" #	X-value	Y-value	Z-value	Radius
Example				
Artery 1-1	28.508	-173.895	221.950	.25
Artery 1-2	-.9385	4.8480	225.062	.15
Artery 1-3	-1.0225	4.5970	237.611	.14
Artery 1-4	-1.1590	4.4565	251.998	.14
Artery 1-5	-1.2280	4.2110	221.950	.13
Artery 1-6	-1.2645	3.9330	225.062	.1
Artery 1-7	-1.2510	3.6935	237.611	.05
Artery 1-8	-1.2050	3.5000	251.998	.01
Artery 2-1	-.9345	4.8945	221.950	.15
Artery 2-2	-.8620	4.5170	225.062	.12
Artery 2-3	-.7850	4.2745	237.611	.1
Artery 2-4	-.6610	3.9005	251.998	.05
Artery 2-5	-.5445	3.6085	221.950	.01

2.4.2 Output files

The output file for coronaries has to be more complicated than the simple list of points that results from the epicardial sock points. The file format used for this information is called a landmark file and the details of its purpose and format are in the manual for *map3d*. Briefly, this is another simple text file organize by landmark type and segments. There are several different landmark types, one of which is a coronary segment, that consists of a list of all the digitized points, ordered according to the numbers supplied in the digitizer file.

2.5 Recording Needles (via rods)

Needles are usually digitized by replacing each needle with a rod of known length and then digitizing the tips and entry points of each needle. From this information and knowledge of the needle electrode size and number of electrodes we can construct the locations of all the needle electrodes. Recently, we have used two points on the needle to determine direction, removing the need for rods to replace each needle. For historic and compatibilty reasons, however, the key word in the .ms file for both rods and needles is "rod". The order assumed for the points on each rod or needle is **outer end = rod tip, then inner end = rod entry point**.

The first step is to convert the entry points and tips into a landmarks file that becomes the input to both *map3d* and subsequent processing programs described in Section 3 of this document..

2.5.1 Digitizer data format

Header Lines (4 in all)			
FILE NAME: *filename* LAST MODIFIED: *Date and Time*			
Free line of text			
Measurements are in millimeters (*or inches*)			
**********			
File Body			
"Rod" #-1/2	X-value	Y-value	Z-value
Example			
Rod 1-1	28.508	-173.895	221.950
rod 1-2	20.061	-178.593	225.062
rod 2-1	11.862	-179.161	237.611
Rod 2-2	5.729	-175.170	251.998

Notes:

- Case of the "rod" entry does not matter.
- The order of the two points in each rod should be **tip then entry point**, but this can be easily corrected at a later stage of the processing.
- NOTE: there must be a space between the "Rod" (or "rod") and the number sequence!!!
- the program will check the order of rods as they are entered in the file and if it is not the same as the labels (*eg.,* rod #1 is not the first, rod #2 is not the second, *etc.,*), then there is the option of letting the program fix the order later (a good choice).

2.5.2 Output files

The output file from *digtoxyz* is a landmarks file, this time with objects of type rod. This file is already in the form that *map3d* uses so that quality control the digitization and conversion is quite easy. For example, with a points file from the epicardial surface called `24mar96_sock.pts` and a rod landmarks file called `24mar96_rods.lmarks` a sample command line for *map3d* would be:

```
map3d -f 24mar96_sock -lm 24mar96-rods.lmarks
```

The main utility of the rods landmarks file, however, is to convert it into needle electrode positions, for which we have the program *needlegeom* , described in section 3 of this document.

2.6 Stimulus sites, occlusions, and other single locations

Most single locations are organized in a similar fashion in *digtoxyz* , as is the resulting landmarks file, which is the output format for such sites.

2.6.1 Digitizer data format

Header Lines (4 in all)			
FILE NAME: *filename* LAST MODIFIED: *Date and Time*			
Free line of text			
Measurements are in millimeters (*or inches*)			
**********			
File Body			
Point-type #	X-value	Y-value	Z-value
Example			
occl 1	28.508	-173.895	221.950
occl 2	20.061	-178.593	225.062
pace 1	11.862	-179.161	237.611
infusion 1	5.729	-175.170	251.998
lead 1	9.729	-134.170	123.998

Note the example contains names for each of the various single points that we currently recognize. “occl” = occlusion; “pace” = pacing site, “infus” = infusion site; and “lead” = selected lead. These items can appear as part of other digitizer files, *eg.,* at the end of a file containing coronaries.

2.6.2 Output files

The output file is a landmarks file. A single landmarks file can contain any mix of the different objects we have described so far. Hence you may wish to digitize coronaries and single points on one file, then have a separate file for needles, and a third file for sock electrodes.

3 Landmarks to geometry: *needlegeom*

The purpose of *needlegeom* is to convert the contents of a landmarks file containing rod or needle endpoints into a .geom file containing needle electrode locations. This output file then serves as input for *map3d* and any program that reads needle geometry.

3.1 Using *needlegeom*

To perform this task, *needlegeom* needs some information, some of which is part of the command line, with the rest (optionall) coming from supplementary files. The command line arguments are as follows:

```
needlegeom lmarkfile geomfile
     [-i needleinfofile]
```

```
[-n num_elecs_per_needle]
[-e elec_spacing(mm)]
[-o offset] [-c channeloffset]
[-t transmatrixfilename]
```

lmarkfile is the name of the landmarks file that contains the rod information. This file is described in Section 2 of this document. The landmarks file can contain other information but only the rod entries are used.

geomfile is the name of the output file you wish to make—a standard, CVRTI, .geom file

-i needleinfofile is a file we shall name "needle info file" because it contains the links between needle numbers and channel numbers from the experiment (channel numbers), as well as the spacing and number of electrodes in each needles. The format of the file is described in section 3.2 below. This file is optional, but only if the channels for the needles all fall in sequence, with no gaps, and the spacing and number of electrodes is constant for all needles so that we can describe them completely by the following command arguments.

-n num_elecs_per_needle is the number of electrodes per needle that you have used. The program assumes that this number is constant for all needles in an experiment—if this is not the case, it is necessary to specify a needle info file..

-e electrode_spacing is the distance **in millimeters** between adjacent electrodes on the needle. The default value is 1.6 mm so this parameter is only necessary if the spacing differs from 1.6. Here too, we assume that the same value can be used for all electrode pairs—if this is not the case, a needle info file is necessary.

-o offset is the distance in millimeters between the entry point on the rod and the first electrode, which has a default value of 0.5 mm. Only specify this option if the offset value is different from 0.5 mm. If it is not constant for all rodes/needles, then a needle info file is necessary.

-c channeloffset is the offset applied to the first channel number assigned to a needle electrode. This is used only when the needle info file is not included in the command line. We assume in this case that electrodes follow in strict sequence for all needles, with no gaps in the order. So needle 1 would have a channel number of *channeloffset* + 1, then needle 2 would follow immediately behind in the ordering, and so on **for all needles!!!** This is probably the exception rather than the rule because of missing or broken needles but it is convenient and removes the need for an explicit needle info file.

-t transmatrixfilename supplies the name of a file that contains the transformation matrix computed by *align* in a separate operation. See Section 3.3 for details.

3.1.1 Examples

If you have a needle info file for the experiment dated 24mar96, the command would be :

```
needlegeom 24mar96-needles.lmarks 24mar96-needles.geom -i 24mar96.ni
```

We assume that the landmarks file `24mar96-needles.lmarks` already exists and that the output file `24mar96-needles.geom` does not (or you wish to overwrite it).
The file `24mar96.ni` is the needle info file, which we describe next.

If we had a simple case in which all needles in the landmarks file followed in order, and there were 10 electrodes per needle, with otherwise default values the command line would become

```
needlegeom 24mar96-needles.lmarks 24mar96-needles.geom -e 10
```

or if needles began at channel 193

```
needlegeom 24mar96-needles.lmarks 24mar96-needles.geom -e 10 -o 192
```

Note that the value after the `-o` option is an offset applied to all channel numbers so for offset = 192, the first needle channel is 193.

3.2 The needle info file

A needle info file serves a number of purposes in telling *needlegeom* :

1. the number label of each rod or needle,
2. the data channel numbers associated with each needle.
3. the number of electrodes in each needle,
4. the spacing between electrodes along the needle, and
5. the offset distance from the entry point to the first electrode of each needle.

Needle number label: Each needle in the landmarks file can have a label associated with it that tell the program (and the human reader of the file) what "needle number" is associated with the entry in the file. This needle number is usually assigned according to some labeling scheme used during the experiement. As described in Section 2, this number also appears in the original digitizing file and is simply carried through to the landmarks file, and to the needle info file. The advantage of the label is that there is no explicit ordering of entries in either the landmarks or the needleinfo file, yet they will be aligned properly by *needlegeom* at execution time.

Channel numbers: the same as the number of the input channel of the multiplexer. We assume that with each needle, the order of electrodes maps directly to an ascending series of channel numbers with the innermost electrode the lowest channel. So a single needle might run from channels 193–202, going from the tip of the needle back toward the wire.

Needles per electrode: In principle, this parameter could take on any value but in practrice, the current needles have either 3, 10, or 12 electrodes and *needlegeom* checks againstthese values for "sanity" at run time.

Needle spacing: This can be a single value or a set of three values, depending on whether the needle contains 10 or 12 electrodes. If we have 12 electrodes, two of them lie outside the heart and have a different spacing. The configurations are illustrated in Figure 1 and the distances in the figure have to be stored in the needle info file. For inside electrodes, we assume that the value is constant over the whole needle, the next two values indicate the spacing to the first outside electrode and between the first and second outside electrodes, respectively. For all spacing and offset distances, we assume units of millimeters.

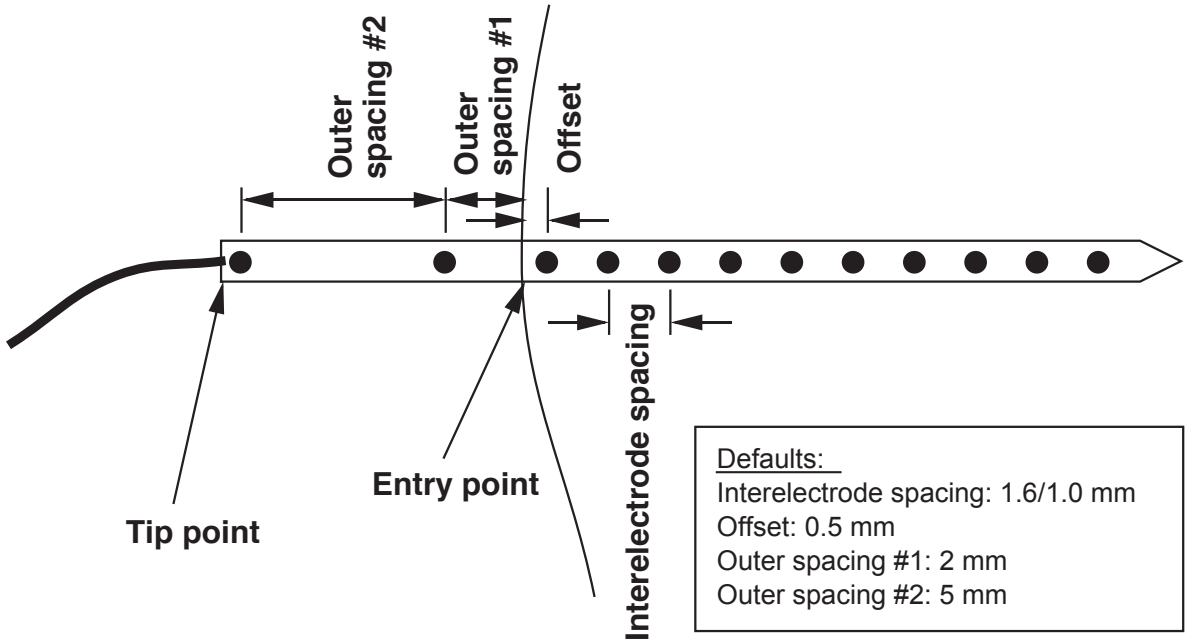

Figure 1: Dimensions of the cardiac needle electrodes. Currently there are three physical configurations: a) 12-pole, spacing 1.0 mm (RV), b) 12-pole, spacing 1.6 mm (LV), and c) 3-pole, spacing 1.6 mm (most distal 3 of the 10 poles.) The 12-pole needles can, however, be connected so that only the 10 intramyocardial electrodes are available and included in the geometry.

Offset: This is the distance between the entry point and the first inside electrode of the needle.

3.2.1 Needle info file format

The format of the needleinfo file is an ASCII, or plain text file, organized as follows:

Format of needle info file	
Comment lines:	
# A comment	
Header Line:	
numneedles	*Number of needles in the file*
Subsequent lines	
First field	*Needle label number*
Second field	*Channel number of distal tip electrode*
Third field	*Number of electrodes per needle*
Fourth field	*Spacing distances, in mm: interelectrode [outer_1 outer_2]*
Fifth field	*Offset distance in mm to first electrode*

Sample Needle Info File

# A sample needle info file for you					
# It contains lots of variety					
14	*14 needles described in the file*				
1	321	10	1.2	0.5	*First needle begins at channel 321, so runs from 321–330*
2	331	10	1.2	.5	*Second needle begins at channel 331, so runs from 331–340*
3	341	10	1.2	0.5	*There are 10 electrode per needle and 1.2 mm spacing*
4	351	3	1.6	0.5	*Only three electrodes in this needle*
5	354	10	1.2	0.5	*Note the jump in channel # of only three from the last needle*
6	364	10	1.2	0.5	
8	384	10	1.2	0.5	*Jump in needle label; # 7 fell out*
9	394	10	1.2	0.5	
10	404	12	1.6 2.0 5.0	0.5	*12-pole needle, spacing changes to 1.6 2.0 amd 5.0 mm*
11	416	12	1.6 2.0 5.0		
12	428	12	1.6 2.0 5.0	0.5	
13	438	12	1.6 2.0 5.0	0.5	*Channels jumps by only 10; last 2 electrodes of previous needle not wired*
15	448	10	1.6	0.5	*Skipped another needle label*
16	458	10	1.6	0.5	14 needles but last one has label 16

To create these files, use any editor (if you use Microsoft Word, be sure to save the file as text only, without line-breaks) and then transfer the file to the SGI directory where the landmarks and geometry files live. The standard extension for the needleinfo file is `.ni` or `.needleinfo`. [1]

3.3 Transformation matrix

The program to place epicardial sock geometry into the reference frame of the torso tank is called *align*. More generally, *align* will align any two sets of points according to some subset of points from both files that should be approximately the same. It generates a transformation matrix that any program can multiply with the nodes of the geometry to apply the same transformation to any other points. When we run *align* we apply the matrix automatically to the nodes of the geometry,

[1] Note that someday this same needle info file will be usable by the *needlegrid* program to make a grid for *smap* and everett.

and any landmarks in a file optionally included on the command line. Usually, we don't include the rod points in this landmarks file (although this would work too) yet we need to apply the same transformation to the needle points that we apply to the sock (and coronaries, stimulus sites, or any other landmarks that we digitize at the same time as the sock). Thus, the transformation matrix.

The format of the transformation is very simple, and the *align* program actually generates the text for this file, and writes it at the time of execution to a log file for your convenience. Just make sure the file exists, and the include it in the argument list for *needlegeom* .

4 Triangulation of needle geometry: *needletri*

4.1 What is *needletri* ?

The program *needletri* has a simple purpose—to take the triangles already entered for surfaces of a needle geometry and propagate them through a subset of the other surfaces in that geometry.

It is effectively the final stage in going from digitized rods to a needle geometry and other sections of this document describe the other steps.

4.2 Using *needletri*

The command line for *needletri* is as follows:

```
needletri infilename outfilename
```

where `infilename` is the name of the input geometry file and `outfilename` the name of the desired output filename (which can be the same as the input).

All that is necessary is to have at least one surface in the input file triangulated and the program applies this same triangularization to any other surface in the file. The user controls which surfaces get propagated to which others and the program does some sanity checks for the number of points in each surface (*i.e.,* you cannot take a triangulated surface with 10 points and propagate it to a surface with 12 points). Another assumption is that we have a regular array of needles with little or no crossing of needles within the tissue. But in the worst case, some manual editing of the resulting file will be necessary with *map3d*.

5 Extracting needle planes: *makeplanes*

The purpose of the *makeplanes* program is to extract sets of needles from a multi-needle geometry file and put them together in a separate file for display of data.

5.1 Using *makeplanes*

```
makeplanes in.geom out.geom config.ni [-o] [-c]
```

where

```
in.geom is the input geometry file
out.geom is the desired output geometry file
config.ni is the needle info file
-o   = assign channels in .ni file to outermost electrode
-c   = close the ends of the planes
```

In more detail

in.geom is a regular .geom file in the CVRTI format. It should contain geometry for some number of needles from which you wish to extract a subset.

out.geom is the output file that *makeplanes* will create for you (when all goes well).

config.ni is the needleinfo file, described in Section 3.2 of this document.

-o is an optional argument that tells *makeplanes* to assume that the channel number in the needleinfo file refers to the outermost electrode and not the innermost, as is the convention.

-c an optional argument that causes *makeplanes*to lace the first and the last needles together.

Note that the input file will have multiple surfaces, one for each of the electrodes in the needles. The output file, on contrast, has just **one** surface and the triangulation of this file is across the needles, in a sense orthogonal to the triangulation of the input file.

5.1.1 Examples

```
makeplanes bt11may94.geom  two24-1092.geom two24-1092.ni -o
```

The famous case which initiated the -o option. This one takes needles from the file bt11may94.geom according to the contents of two24-1092.ni and makes a new file called two24-1092.geom. The -o option ensures that the electrode order along the needle is reversed, *i.e.,* that the channel numbers in two24-1092.ni refer to the outermost electrode.

The file two24-1092.ni looks like this

```
9
1 24 12 1.6 2.0 4.0 0.5
2 192 12 1.6 2.0 4.0 0.5
3 336 12 1.6 2.0 4.0 0.5
4 408 12 1.6 2.0 4.0 0.5
5 600 12 1.6 2.0 4.0 0.5
6 504 12 1.6 2.0 4.0 0.5
7 888 12 1.6 2.0 4.0 0.5
8 1032 12 1.6 2.0 4.0 0.5
9 1092 12 1.6 2.0 4.0 0.5
```

So 24 refers to channel 24, which is on the outer end of the needle (13 is on the inner end).