Kitware

BoneTextureExtension

User Tutorial

Jean-Baptiste Vimort et al.

UNIVERSITY OF MICHIGAN

THE UNIVERSITY of NORTH CAROLINA at CHAPEL HILL

- Presentation
  - Quick view
  - Installation
  - The modules in Slicer
- The interactive modules
  - Bone Texture
  - Bone Texture Serializer
- The Computational modules
  - Compute GLCM Features
  - Compute GLCM Feature Maps
  - Compute GLRLM Features
  - Compute GLRLM Feature Maps
  - Separate Vector Image
- Parameters influence
  - Experimental conditions
  - Default parameters
  - Mask
  - Number of bins
  - Voxel intensity Range
  - Distance Range
  - Neighborhood radius
- Acknowledgement Resources Contact

Kitware

Presentation

- Quick view
- Installation
- The modules in Slicer

Kitware

Quick view

The goal of this 3DSlicer extension is to allow textural analysis of 3D volumes.

For that two well-known texture analysis methods are used:

- the study of Grey Level Co-occurrence Matrix (GLCM)
- the study of Grey Level Run Length Matrix (GLRLM)

Two different types of outputs can be computed:

- *textural features:* this type of feature set will be computed over the whole volume and thereby characterize the texture of the entire 3D volume
- *textural maps:* for each voxel of the input volume, a set of textural features will be computed for the neighborhood of this voxel. The output will be a 3D texture map where each voxel locally describe the texture of the input volume

Quick view

BoneTextureExtension is composed 7 modules:

- *Two main modules:* those modules are made to help the user by allowing to run all the algorithms from the same modules. They allow to easily modify the parameters, visualize the results, and recursively run the algorithms on an input set

- *Five secondary modules:* those modules each run a single algorithm, they are useful to have better computation time (they are used by the two main modules). If needed those modules can also be used outside of Slicer thanks to command lines in the terminal.

Quick view

The goal of this 3DSlicer extension is to allow textural analysis of 3D volumes.

For that two well-known texture analysis methods are used:

- the study of Grey Level Co-occurrence Matrix (GLCM)
- the study of Grey Level Run Length Matrix (GLRLM)

Two different types of outputs can be computed:

- *textural features:* this type of feature set will be computed over the whole volume and thereby characterize the texture of the entire 3D volume
- *textural maps:* for each voxel of the input volume, a set of textural features will be computed for the neighborhood of this voxel. The output will be a 3D texture map where each voxel locally describe the texture of the input volume

Quick view

The two main modules of BoneTextureExtension are:

- *BoneTexture:*
  - four textural analysis algorithms available
  - run on a single case
  - immediate visualization of the results in Slicer
  - easy to modify the input parameters and re-run

- *BoneTextureSerializer:*
  - four textural analysis algorithms available
  - run on several cases
  - save every results (no immediate visualization in Slicer)
  - easy specification of the input results

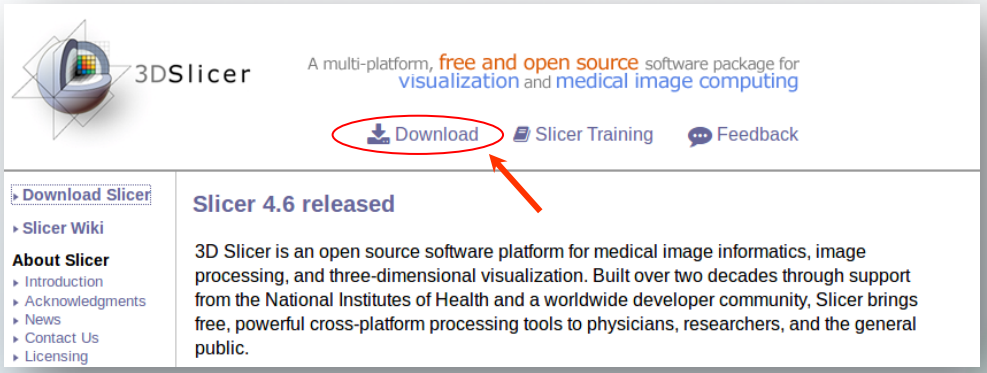
Quick view

The five secondary modules are:

- *ComputeGLCMFeatures:* Compute the textural features over the whole scan using the Grey Level Co-occurrence Matrix
- *ComputeGLCMFeatureMaps:* Compute the textural features maps using the Grey Level Co-occurrence Matrix
- *ComputeGLRLMFeatures:* Compute the textural features over the whole scan using the Grey Level Run Length Matrix
- *ComputeGLRLMFeatureMaps:* Compute the textural features maps using the Grey Level Run Length Matrix
- *SeparateVectorImage:* Separate the feature maps contained in a singl dwi file into several separated modules

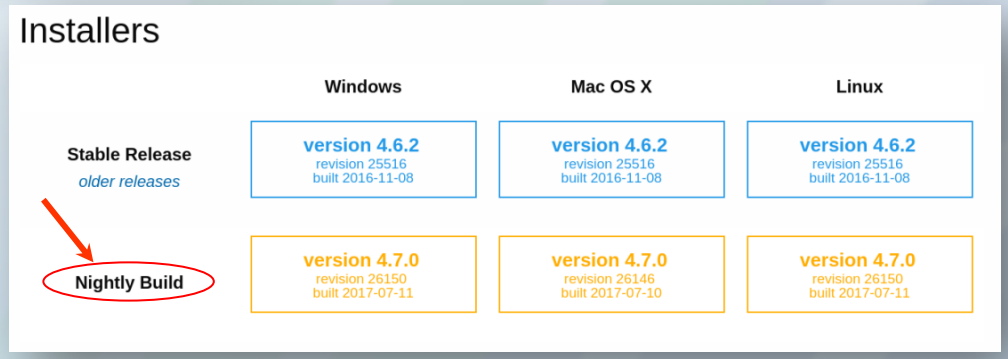
Installation

BoneTextureExtension is a plugin extension only available on the nightly version of Slicer at this date (07/10/2017), if you don't already have a nightly version of Slicer on your machine, you should download one on the [3D Slicer website] and install it:

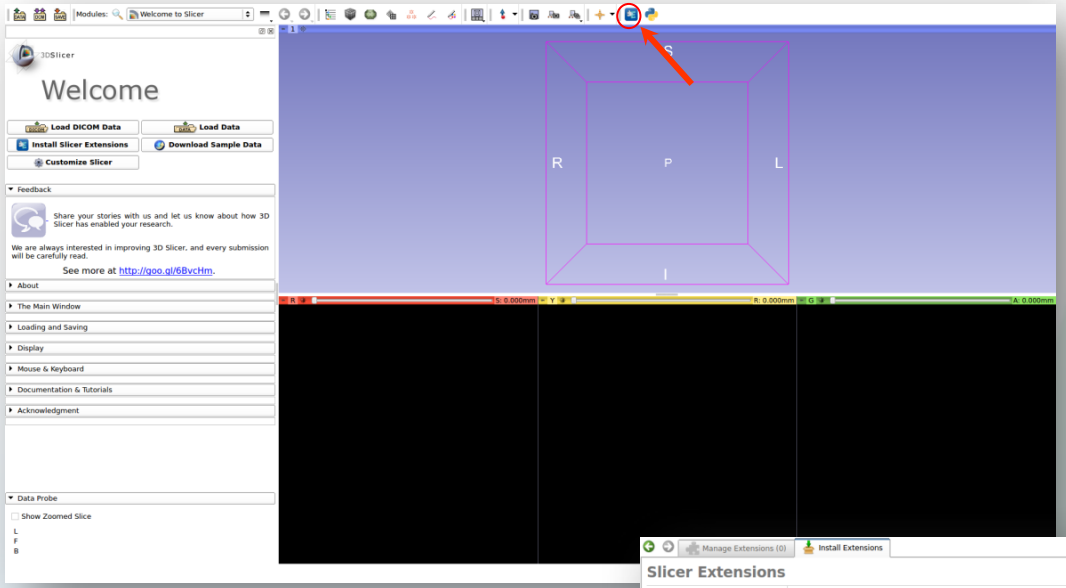


- Go to the Slicer download page

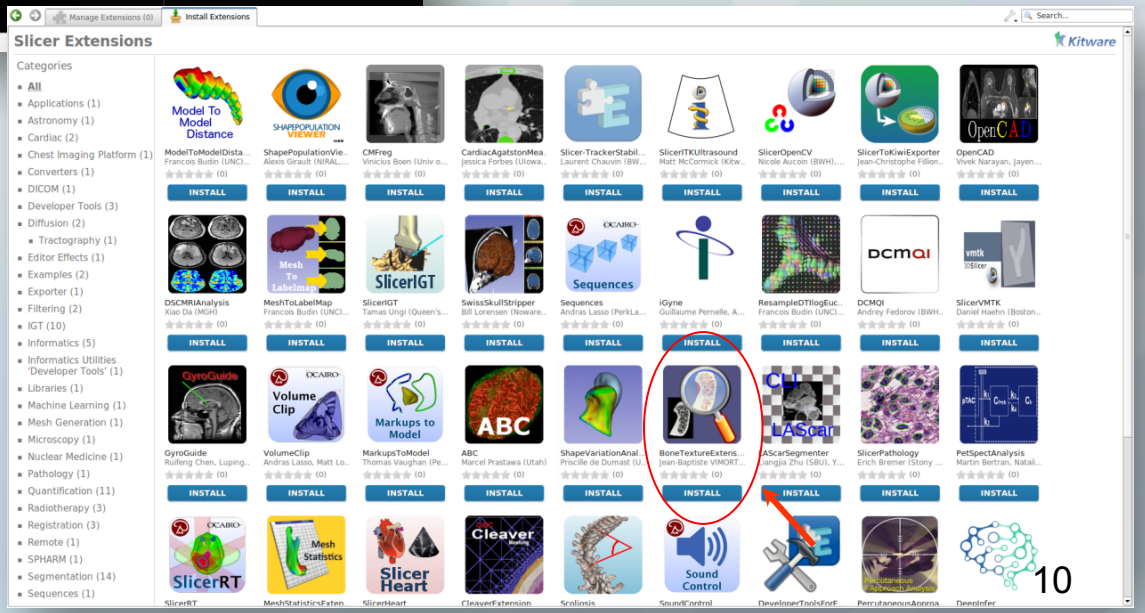
- Download the nightly version of 3DSlicer corresponding to your operating system and install it



Installation



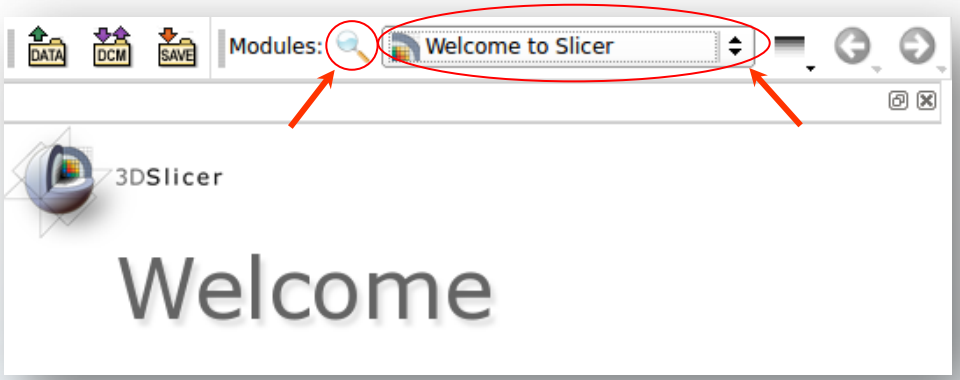
- Once You launched 3DSlicer, open the extension manager



- In the Extension Manager, look for BoneTextureExtension and install it by clicking on the “install” button under it, then restart Slicer

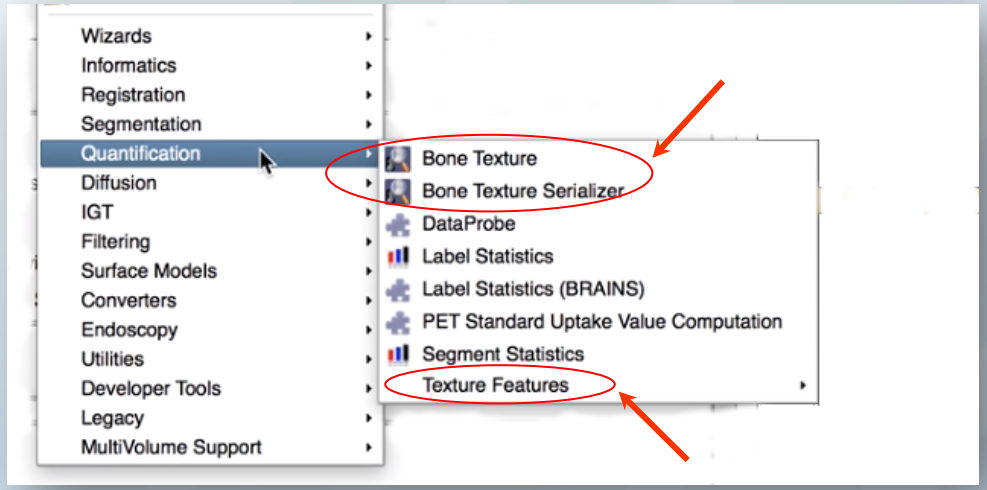
The modules in Slicer

In order to find the modules in Slicer you can either:



- Click on the loop and look for the module by writing its name
- Use the drop-down menu listing all the installed module

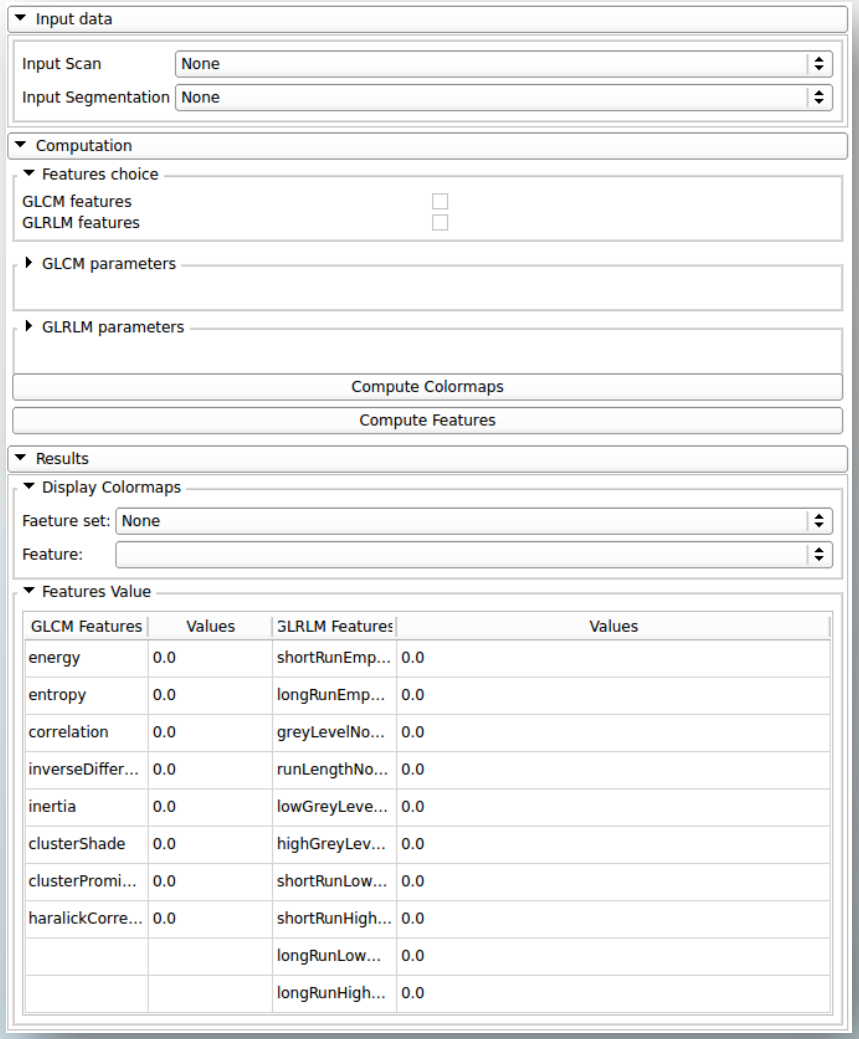
- In the drop-down menu, the two main modules are located in the “quantification” tab, the secondary modules are located in a sub-tab called “Texture Feature” in the “quantification” tab



The interactive modules
(scripted modules)

- Bone Texture
- Bone Texture Serializer

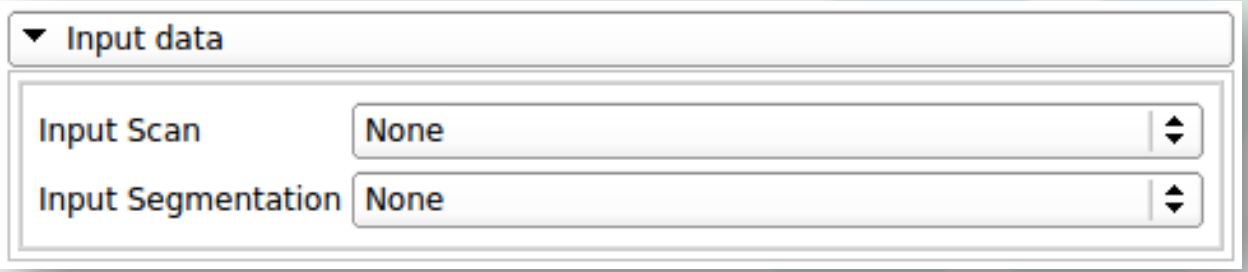
Bone Texture



The BoneTexture module is composed of three main parts:

- *The input data tab:* used to specify the input volume and the input segmentation (if it exist)
- *The computation tab:* used to specify the type of results wanted as well as the parameters for each algorithms
- *The results tab:* used to observe the results

Bone Texture

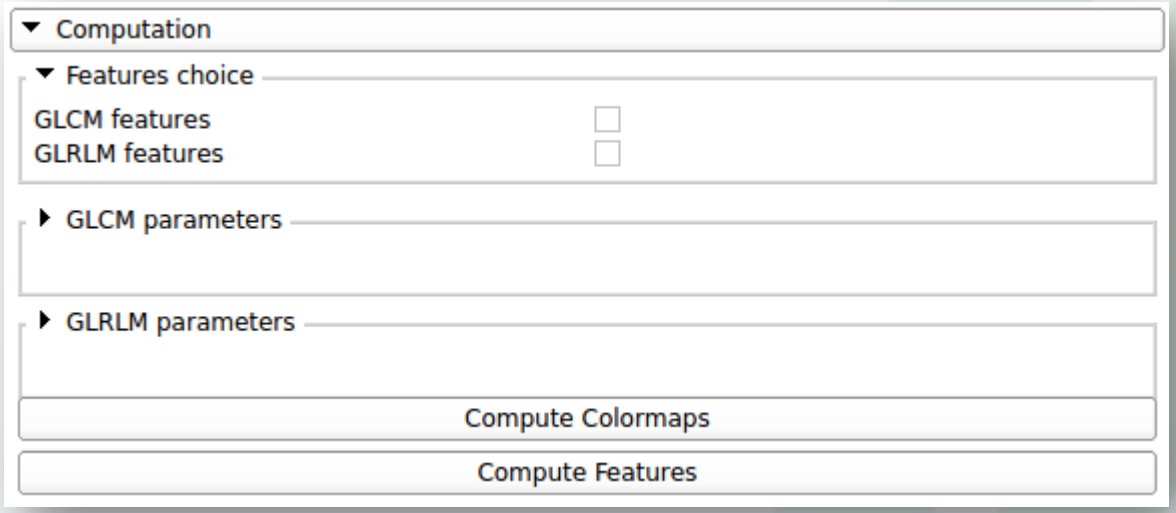


The input section allows the user to specify:

- *An input scan:* it's interesting to notice that the computation time will improve if the useful data is taking as much space as possible in the input scan (a preliminary cropping step might be considered in order to erase the useless data on the boundaries)

- *An input mask:* the mask is optional, therefore it will reduce the computation time of the algorithms

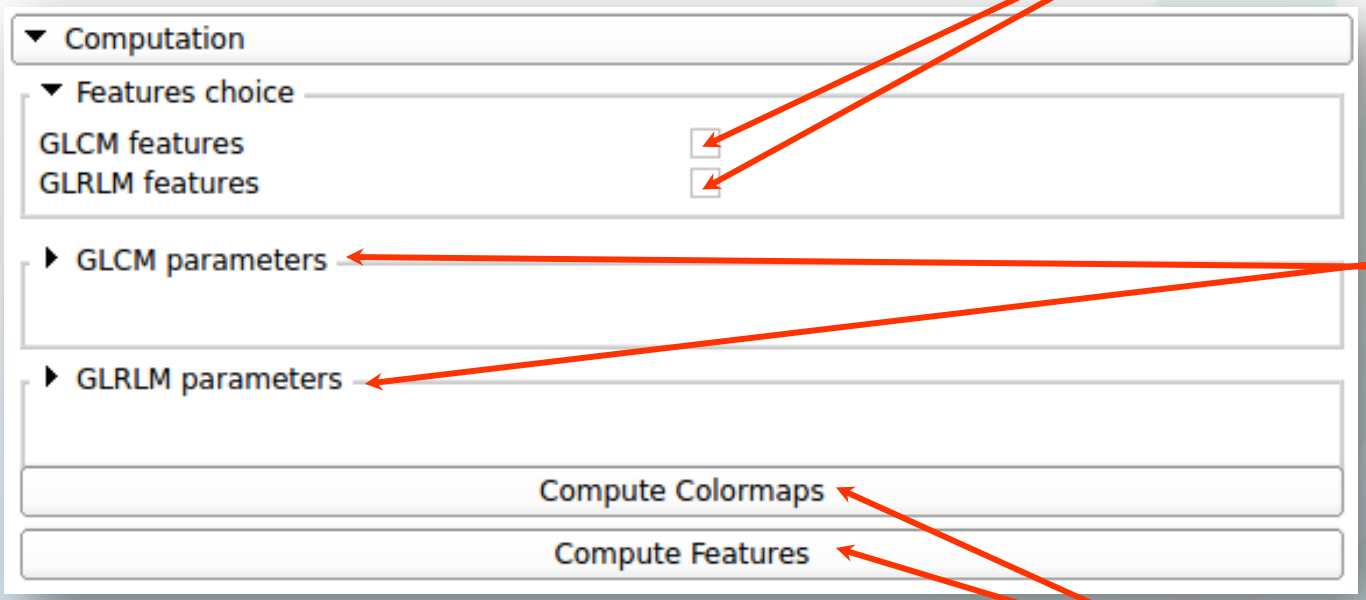
Bone Texture



This section allows the user to choose:

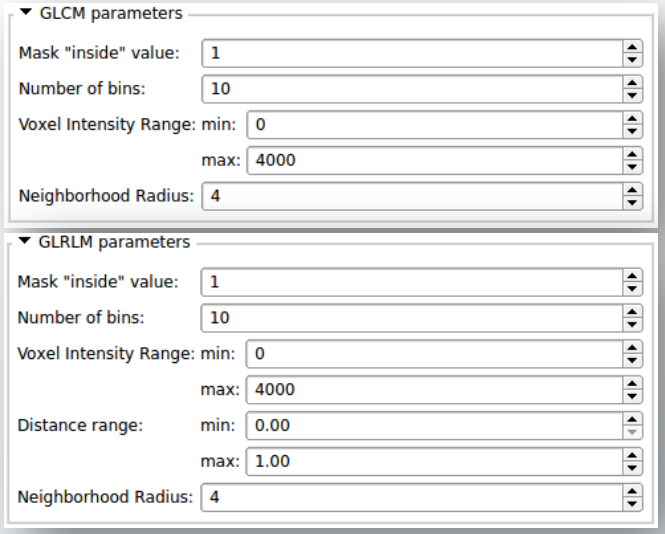
- The type of features wanted: Run length features and/or co-occurrence features
- All the parameters necessary to run each algorithms
- The type of outputs wanted: texture features and/or texture maps

Bone Texture



- Check the type of feature desired
- Specify the input parameters for each algorithm
- Click on the button to obtain the corespondent features

Bone Texture



There is more precision about the influence of each parameter later in the presentation

The GLCM and GLRLM sub-tabs allow to choose for each type of algorithm: (more precision about the influence of each parameter later in the presentation)

- *Inside Mask Value:* The pixel value that defines the ”inside” of the mask
- *Number of Intensity bins:* The number of intensity bins in the Grey Level Matrices
- *Pixel Intensity Min:* Minimum of the pixel intensity range over which the features will be calculated
- *Pixel Intensity Max:* Maximum of the pixel intensity range over which the features will be calculated
- *Distance Min:* Minimum of the distance range over which the features will be calculated
- *Distance Max:* Maximum of the distance range over which the features will be calculated

Bone Texture

The results section allows the user to observe:

- *The texture features:* they will be displayed thanks to a table.
- *The texture maps:* the user will be able to choose the dwi file (file generated by the algorithms) containing the texture maps and which feature he wants to display in the visualization window of Slicer

Bone Texture

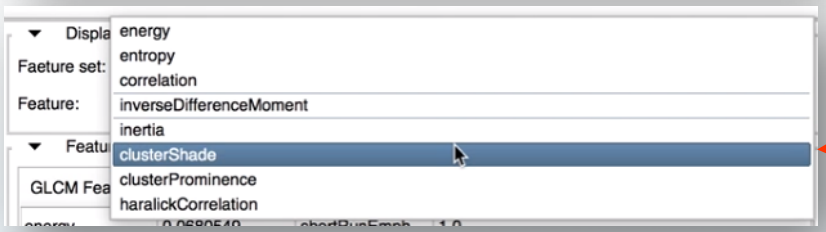
GLCM Features	Values	GLRLM Features	Values
energy	0.068...	shortRunEmphasis	1.0
entropy	4.322...	longRunEmphasis	1.0
correlation	0.203...	greyLevelNonuniformity	14362.6
inverseDifferenceMoment	0.823...	runLengthNonuniformity	105010.0
inertia	0.381...	lowGreyLevelRunEmphasis	0.068747
clusterShade	39.29...	highGreyLevelRunEmphasis	33.1306
clusterProminence	840.5...	shortRunLowGreyLevelEmphasis	0.068747
haralickCorrelation	6464....	shortRunHighGreyLevelEmphasis	33.1306
		longRunLowGreyLevelEmphasis	0.068747
		longRunHighGreyLevelEmphasis	33.1306

Each texture features computed over the whole input volume will be displayed in this table. Only the selected features will be computed and displayed.

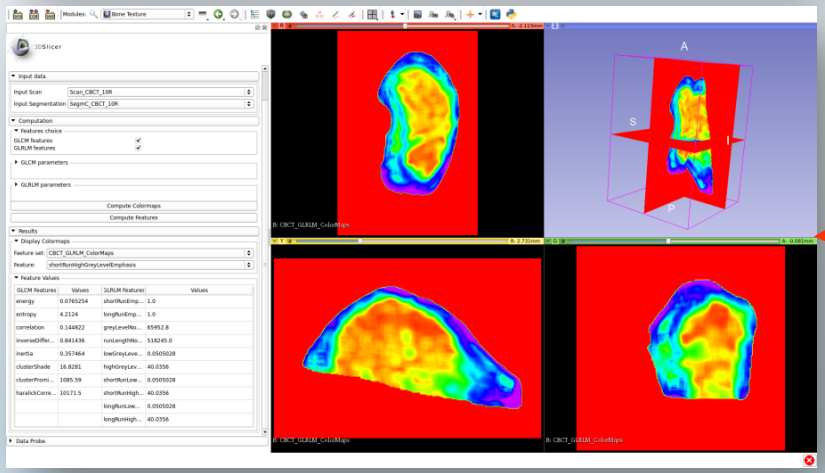
Bone Texture

To observe the feature maps generated:

- Select feature set (saved as dwi image) containing the different feature maps

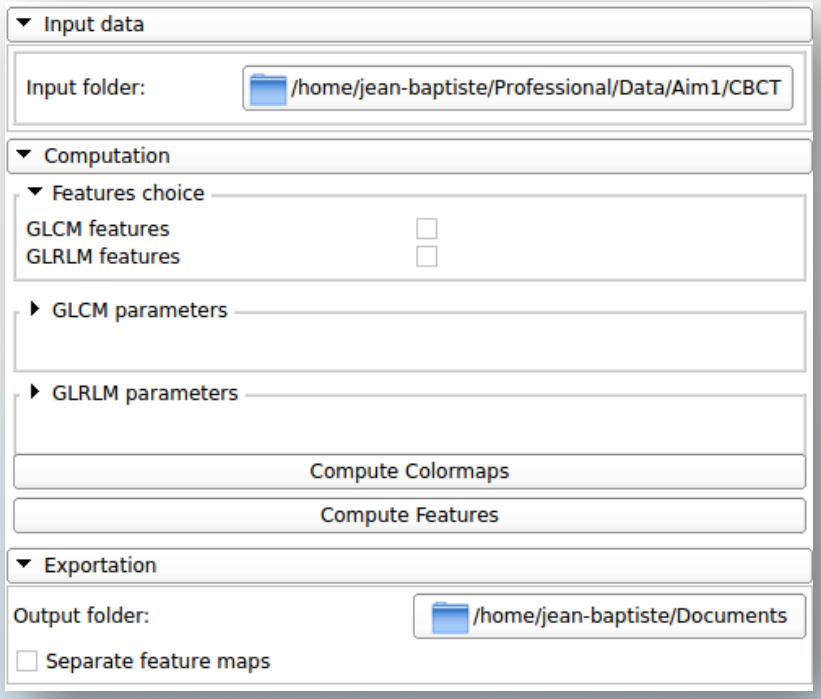


- Choose in this feature set, the feature that you want to display



- The feature map will be displayed in the visualization window of Slicer

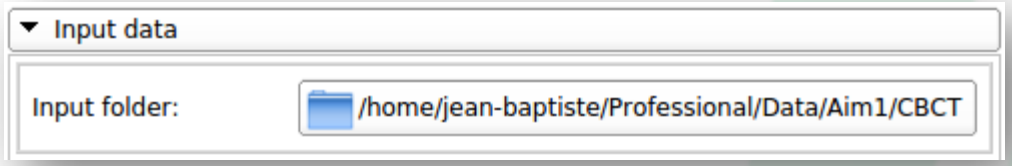
Bone Texture Serializer



The BoneTextureSerializer module is composed of three main parts:

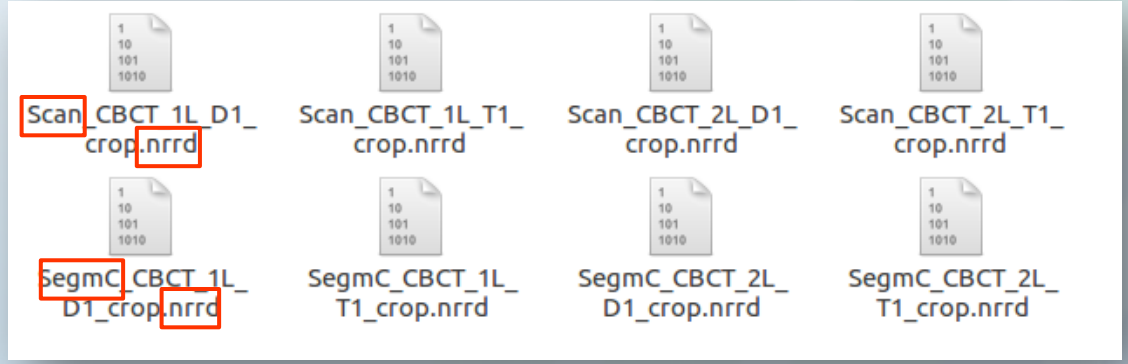
- *The input data tab:* used to specify the folder containing the input volumes and the input segmentations (if it exist)
- *The computation tab:* used to specify the type of results wanted as well as the parameters for each algorithms
- *The exportation tab:* used to specify the output folder where the results will be saved

Bone Texture Serializer



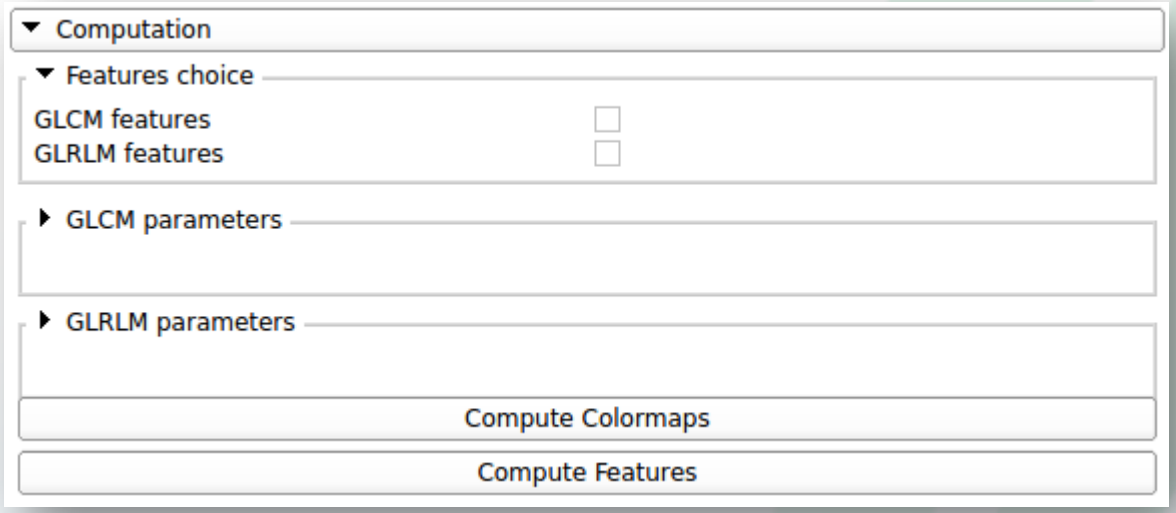
The input section allows the user to specify:

- An input folder that should contains all the input volumes and segmentations; for a better computation time, the input should fit to the description made in the BoneTexture input slide)
- The input scan should be named ScanXXX.nrrd, where XXX is the ID of the case
- If it exists, the input mask should be named SegmCXXX.nrrd, where XXX is the ID corresponding to the the input scan



Kitware

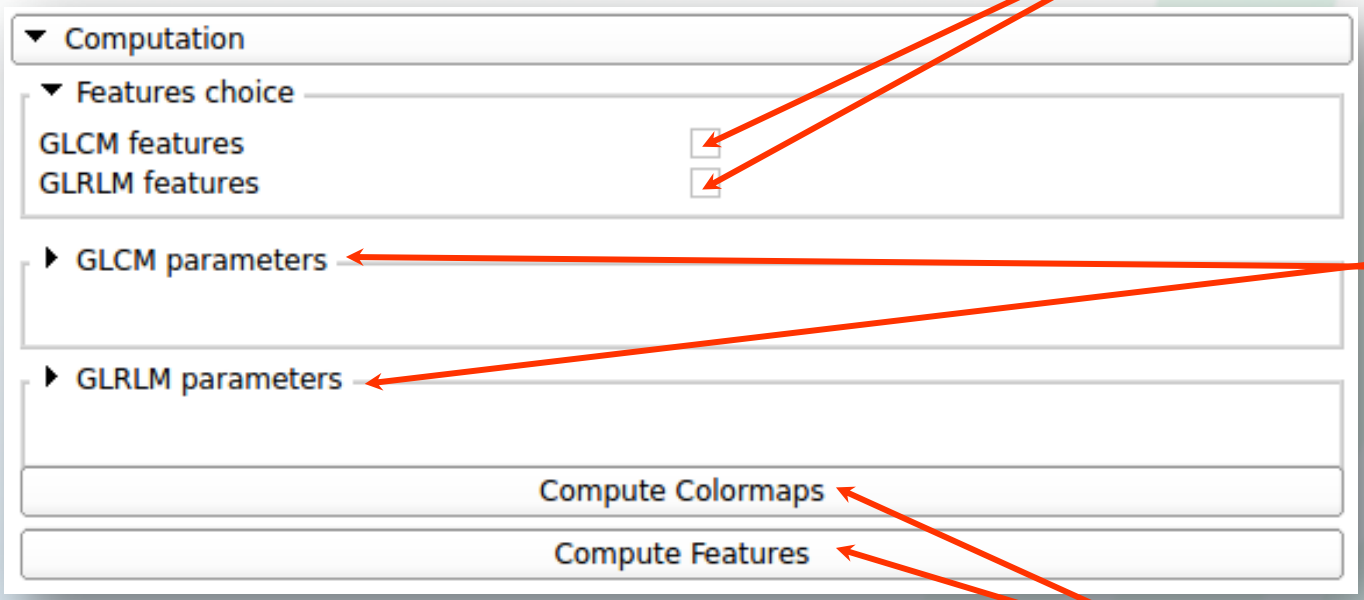
Bone Texture Serializer



This section allows the user to choose:

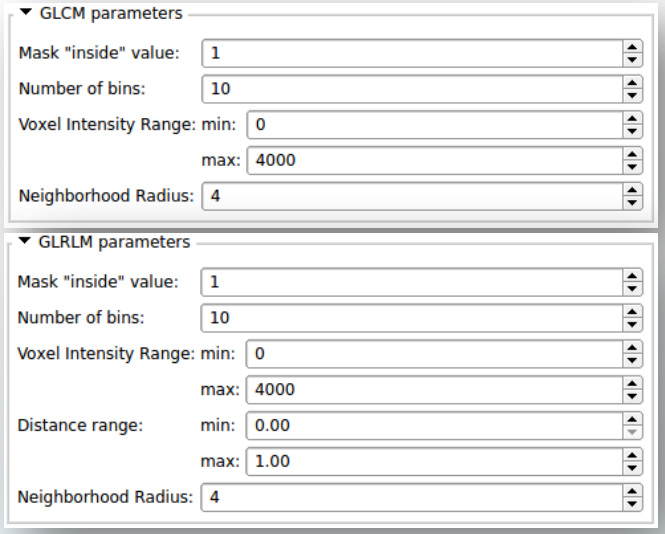
- The type of features wanted: Run length features and/or co-occurrence features
- All the parameters necessary to run each algorithms
- The type of outputs wanted: texture features and/or texture maps

Bone Texture Serializer



- Check the type of feature desired
- Specify the input parameters for each algorithm
- Click on the button to obtain the corespondent features

Bone Texture Serializer

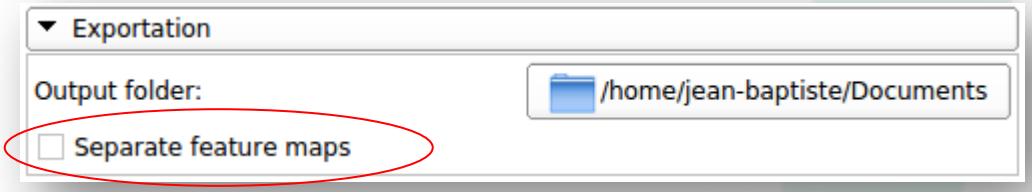


There is more precision about the influence of each parameter later in the presentation

The GLCM and GLRLM sub-tabs allow to choose for each type of algorithm: (more precision about the influence of each parameter later in the presentation)

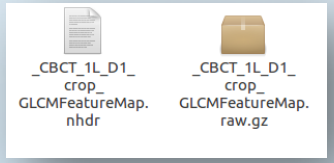
- *Inside Mask Value:* The pixel value that defines the ”inside” of the mask
- *Number of Intensity bins:* The number of intensity bins in the Grey Level Matrices
- *Pixel Intensity Min:* Minimum of the pixel intensity range over which the features will be calculated
- *Pixel Intensity Max:* Maximum of the pixel intensity range over which the features will be calculated
- *Distance Min:* Minimum of the distance range over which the features will be calculated
- *Distance Max:* Maximum of the distance range over which the features will be calculated

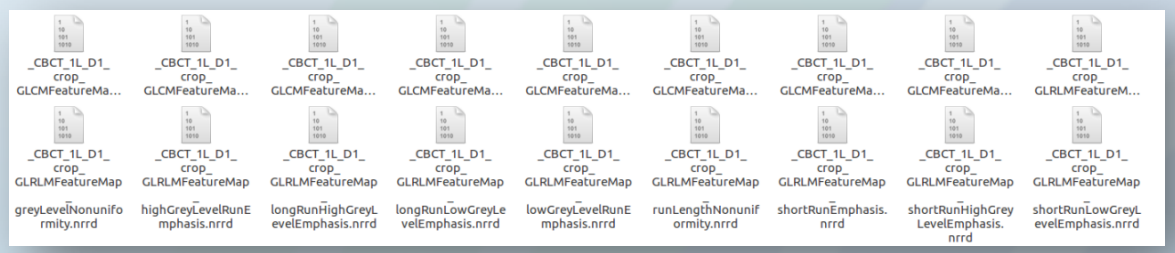
Bone Texture Serializer



The exportation section allows the user to specify the output folder where all the results will be saved:

- The texture features for the whole image will be saved in a csv file
- The texture maps can either be saved:
  - all in a same file (a diffusion weighted image containing all the feature maps of a case)
  - In separated files (each feature map will be saved as a different volume)



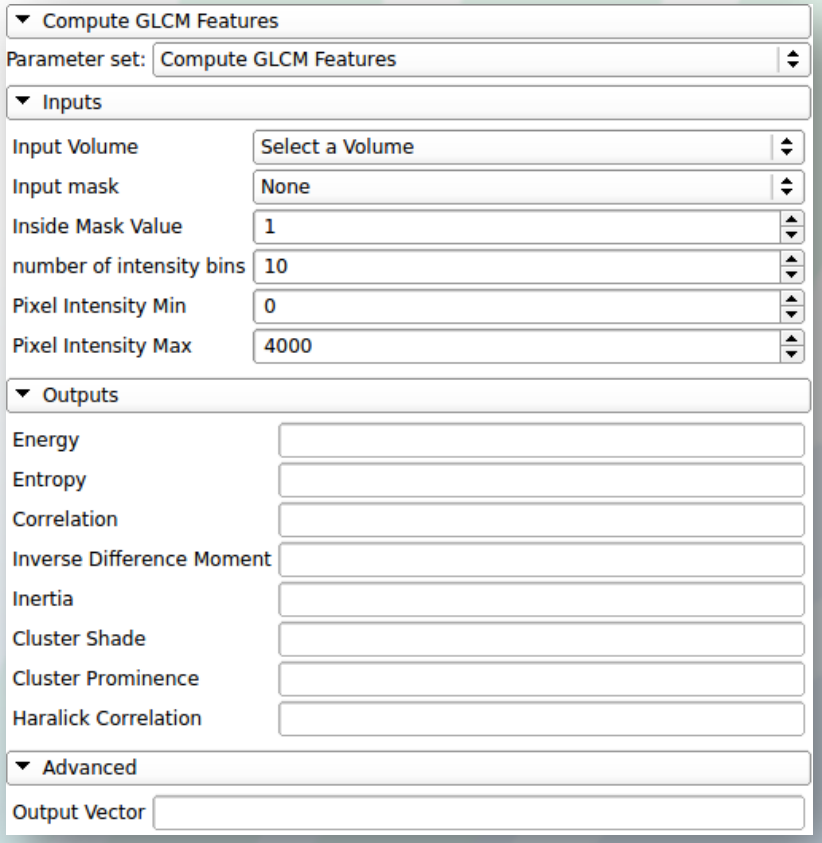


The computational modules
(CLI modules)

- Compute GLCM Features
- Compute GLCM Feature Maps
- Compute GLRLM Features
- Compute GLRLM Feature Maps
- Separate Vector Image

Compute GLCM Features

This module will compute the textural features over the whole scan using the Grey Level Co-occurrence Matrix:

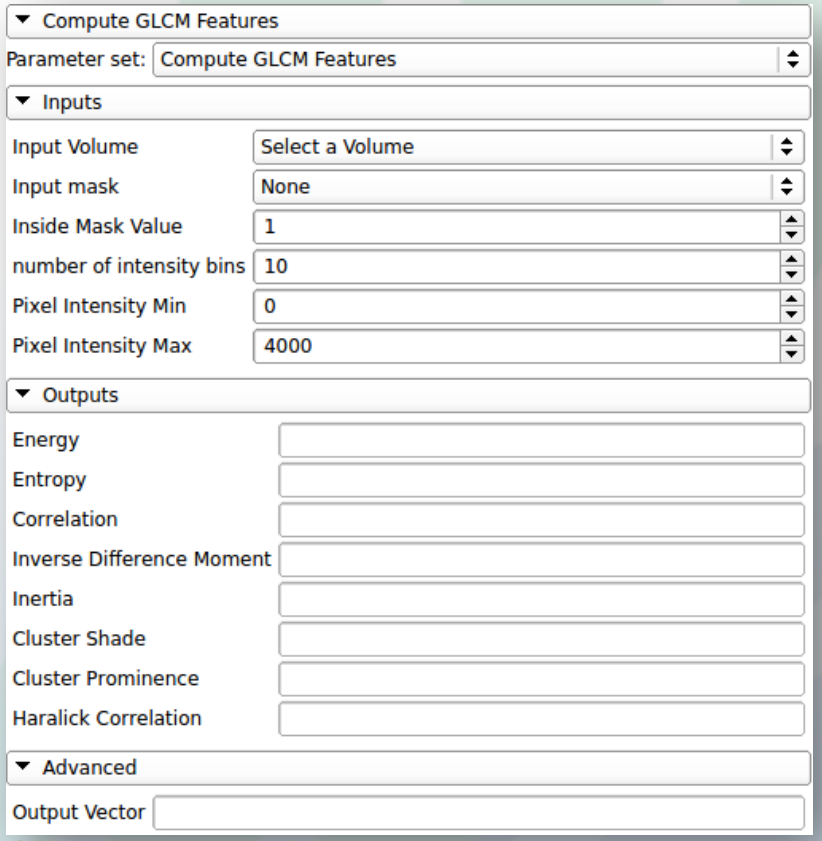


Inputs:

- Input volume [index: 0] : Input Volume
- Input mask [-s --inputMask] (None) : A mask defining the region over which texture features will be calculated
- Inside Mask Value [-i --inputMask] (1) : The pixel value that defines the ”inside” of the mask
- Number of Intensity bins [-b --binNumber] (10) : The number of intensity bins
- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minimum of the pixel intensity range over which the features will be calculated
- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minimum of the pixel intensity range over which the features will be calculated

Compute GLCM Features

This module will compute the textural features over the whole scan using the Grey Level Co-occurrence Matrix:



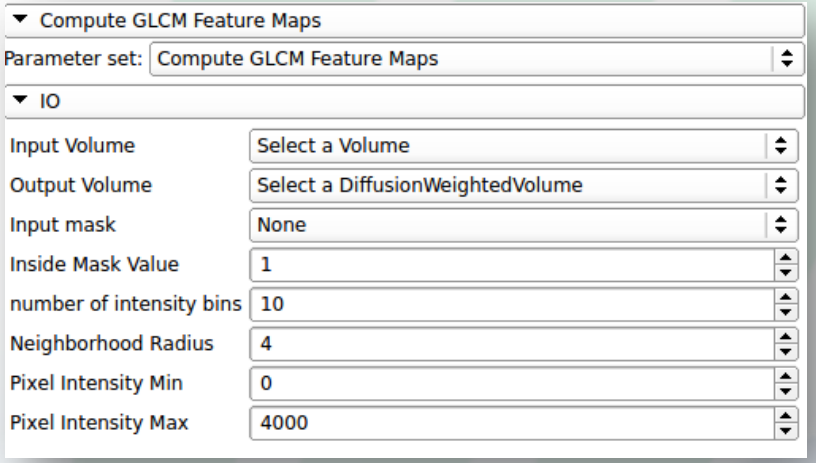
Outputs:

- Energy [output] : Energy feature value
- Entropy [output] : Entropy feature value
- Correlation [output] : Correlation feature value
- Inverse difference moment [output] :Inverse difference moment feature value
- Inertia [output] : Inertia feature value
- Cluster shade [output] : Cluster shade feature value
- Cluster prominence [output] : Cluster prominence feature value
- Haralick correlation [output] : Haralick correlation feature value

Advanced:

- Output Vector [output] : Output vector containing all the feature value stored in the same order than previously

Compute GLCM Feature Maps



This module will compute the textural features maps using the Grey Level Co-occurrence Matrix:

Inputs/Outputs:

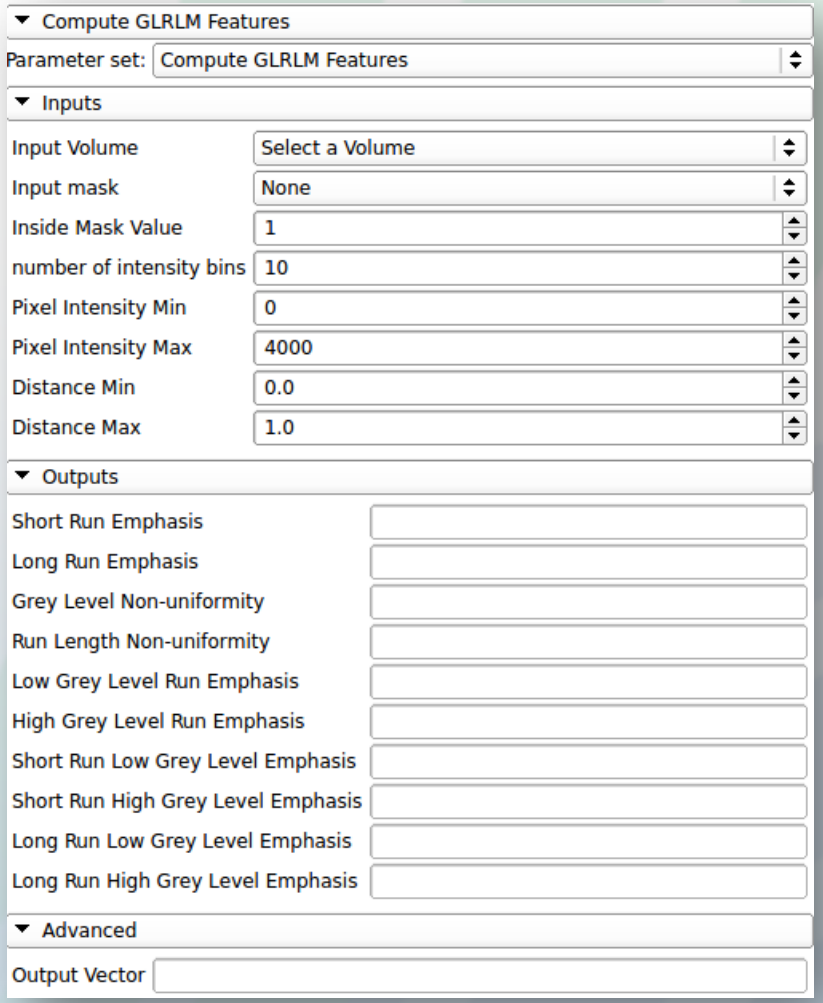
- Input volume [index: 0] : Input Volume
- Output volume [index: 1] : Output diffusion-weighted volume where the 8 feature maps will be stored
- Input mask [-s --inputMask] (None) : A mask defining the region over which texture features will be calculated
- Neighborhood radius [-n --neighborhoodRadius] (4) : The size of the neighborhood radius
- Inside Mask Value [-i --inputMask] (1) : The pixel value that defines the ”inside” of the mask
- Number of Intensity bins [-b --binNumber] (10) : The number of intensity bins
- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minimum of the pixel intensity range over which the features will be calculated
- Pixel Intensity Max [-P --pixelIntensityMax] (4000) : Maximum of the pixel intensity range over which the features will be calculated

Compute GLRLM Features

This module will compute the textural features over the whole scan using the Grey Level Run Length Matrix:

Inputs(1/2):

- Input volume [index: 0] : Input Volume
- Input mask [-s --inputMask] (None) : A mask defining the region over which texture features will be calculated
- Inside Mask Value [-i --inputMask] (1) : The pixel value that defines the ”inside” of the mask
- Number of Intensity bins [-b --binNumber] (10) : The number of intensity bins

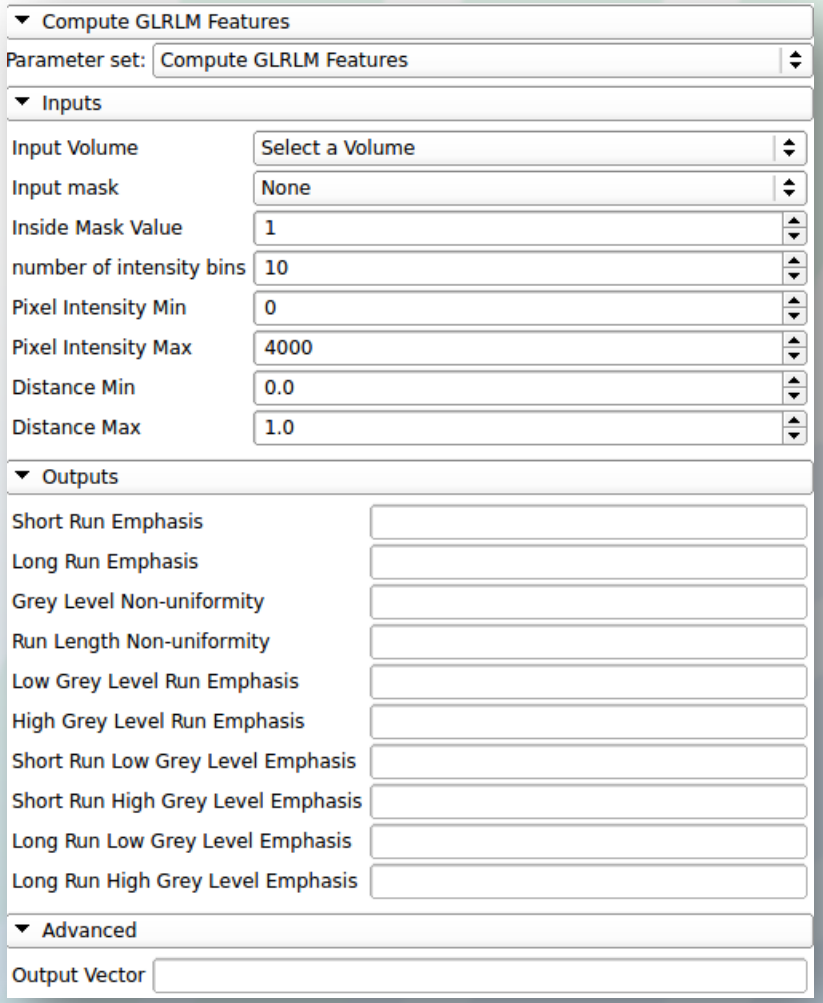


Compute GLRLM Features

This module will compute the textural features over the whole scan using the Grey Level Run Length Matrix:

Inputs(2/2):

- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minnimum of the pixel intensity range over which the features will be calculated
- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minimum of the pixel intensity range over which the features will be calculated
- Distance Min [-d --distanceMin] (0.0) : Minimum of the distance range over which the features will be calculated
- Distance Max [-D --distanceMax] (1.0) : Maximum of the distance range over which the features will be calculated

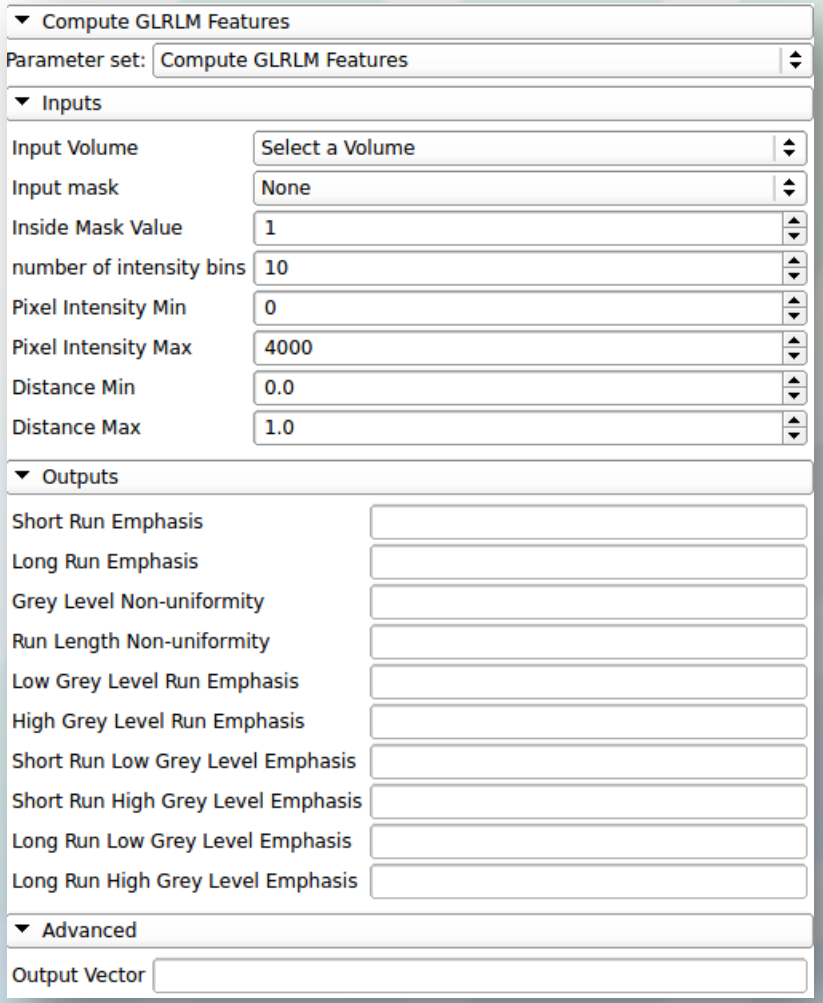


Compute GLRLM Features

This module will compute the textural features over the whole scan using the Grey Level Run Length Matrix:

Outputs(1/2):

- Short Run Emphasis [output] : Short Run Emphasis feature value
- Long Run Emphasis [output] : Long Run Emphasis feature value
- Grey Level Non-uniformity [output] : Grey Level Non-uniformity feature value
- Run Length Non-uniformity [output] : Run Length Non-uniformity feature value
- Low Grey Level Run Emphasis [output] : Low Grey Level Run Emphasis feature value
- High Grey Level Run Emphasis [output] : High Grey Level Run Emphasis feature value

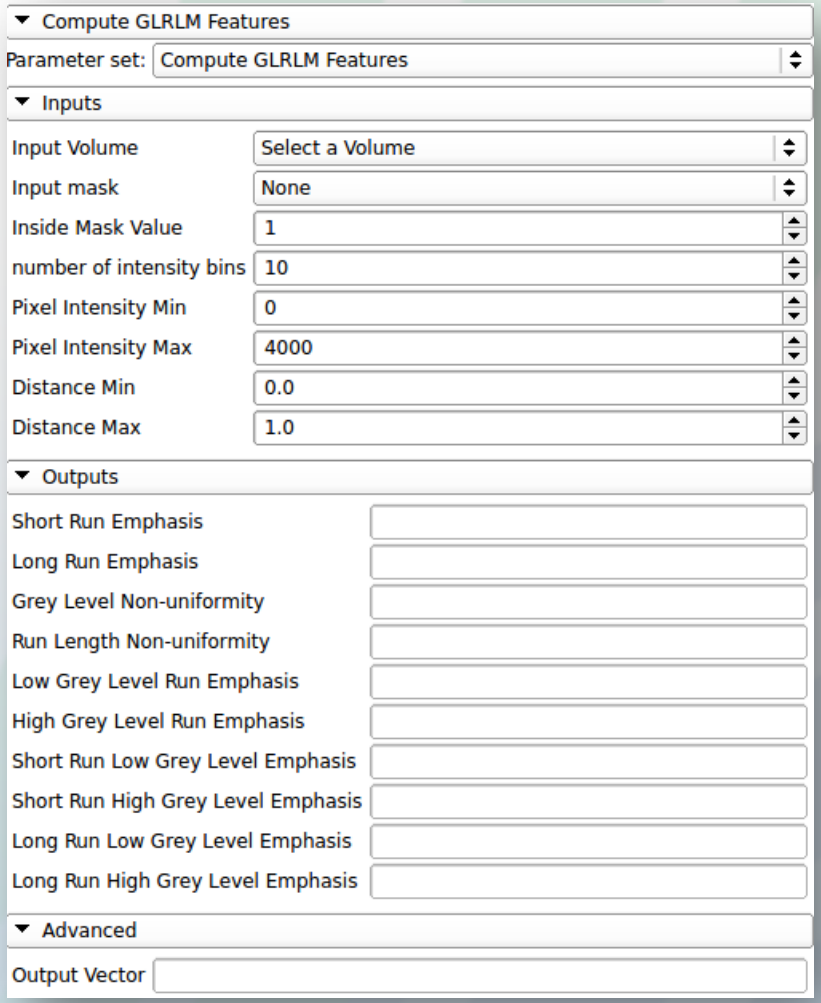


Compute GLRLM Features

This module will compute the textural features over the whole scan using the Grey Level Run Length Matrix:

Outputs(2/2):

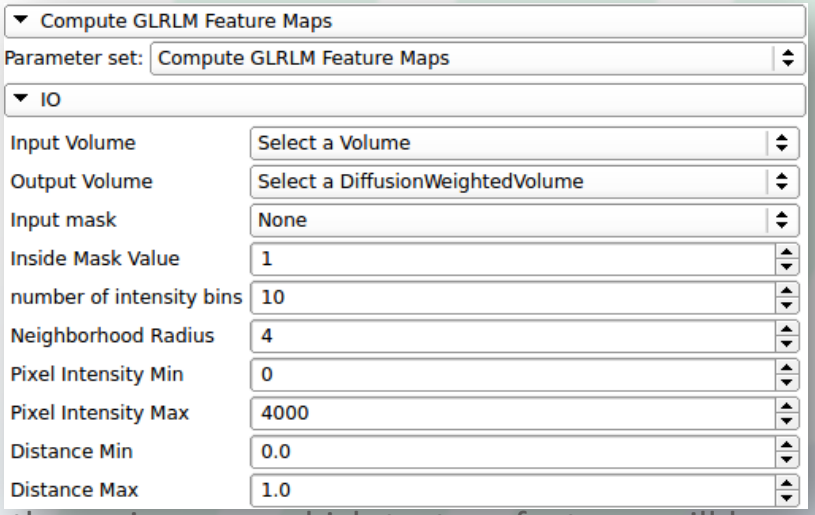
- Short Run Low Grey Level Emphasis [output] : Short Run Low Grey Level Emphasis feature value
- Short Run High Grey Level Emphasis [output] : Short Run High Grey Level Emphasis feature value
- Long Run Low Grey Level Emphasis [output] : Long Run Low Grey Level Emphasis feature value
- Long Run High Grey Level Emphasis [output] : Long Run High Grey Level Emphasis feature value



Advanced:

- Output Vector [output] : Output vector containing all the feature value stored in the same order than previously

Compute GLRLM Feature Maps



This module will compute the textural features maps using the Grey Level Run Length Matrix:

Inputs/outputs:

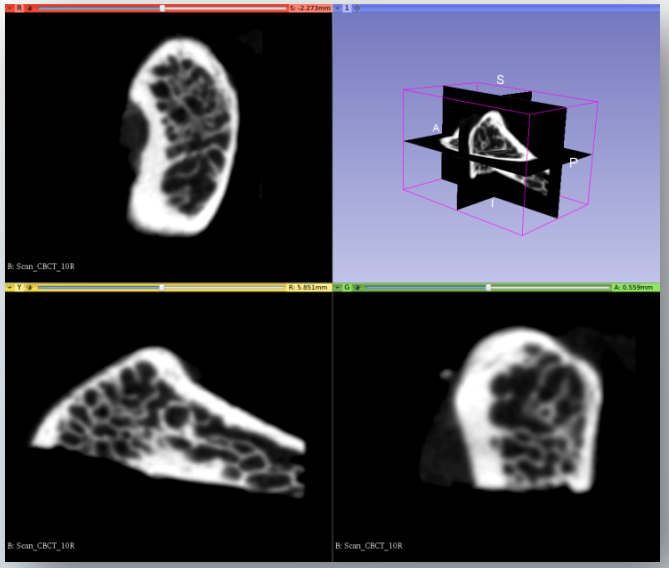
- Input volume [index: 0] : Input Volume
- Output volume [index: 1] : Output diffusion-weighted volume where the 8 feature maps will be stored
- Input mask [-s --inputMask] (None) : A mask defining the region over which texture features will be calculated
- Inside Mask Value [-i --inputMask] (1) : The pixel value that defines the ”inside” of the mask
- Number of Intensity bins [-b --binNumber] (10) : The number of intensity bins
- Neighborhood radius [-n --neighborhoodRadius] (4) : The size of the neighborhood radius
- Pixel Intensity Min [-p --pixelIntensityMin] (0) : Minimum of the pixel intensity range over which the features will be calculated
- Pixel Intensity Max [-P --pixelIntensityMax] (4000) : Maximum of the pixel intensity range over which the features will be calculated
- Distance Min [-d --distanceMin] (0.0) : Minimum of the distance range over which the features will be calculated
- Distance Max [-D --distanceMax] (1.0) : Maximum of the distance range over which the features will be calculated

Parameters influence

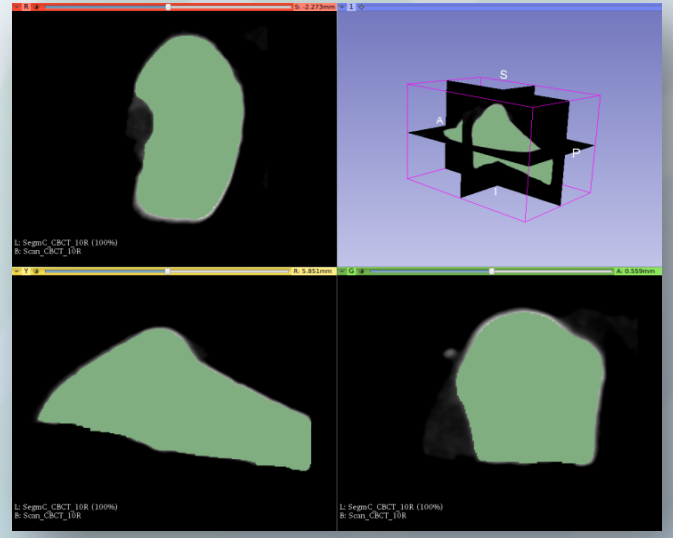
- Experimental conditions
- Default parameters
- Mask
- Number of bins
- Voxel intensity Range
- Distance Range
- Neighborhood radius

Experimental Conditions

In order to observe the influence of each parameters on the output textural maps and the computation time BoneTexture module has been ran on the same input with different parameters set.



Our study case input is a Cone Beam Computed Tomography (CBCT) of a condyle. The input volume size is 197x287x193 (10.912.024 voxels) with an isotropic spacing of 0.08mm and a intensity range between -777 and 4276.



We also used a segmentation of the condyle (mask), this segmentation is composed of 1.622.440 voxels (15% of the scan).

Experimental Conditions

The machine used for the experiments had the following configuration:

The version of Slicer used was the nightly build of July 4th 2017 (built in released mode).

Default parameters

The default parameter set that is used as reference is:

- *Inside Mask Value:* 1
- *Number of Intensity bins:* 10
- *Pixel Intensity Min:* 0
- *Pixel Intensity Max:* 4500
- *Distance Min:* 0.0
- *Distance Max:* 1.25
- *Neighborhood:* 4

As a reference, with the default parameters, the GLCM algorithm takes about 1:11 minute to run, and the GLRLM one 2:49 minutes.

Warning: The discussions that will be made in this section might be specific to the study case and the type of features that need to be detected.

They are mainly here to illustrate the importance of choosing correctly the parameters and try to give a better understanding of the influence of each one of those parameters.

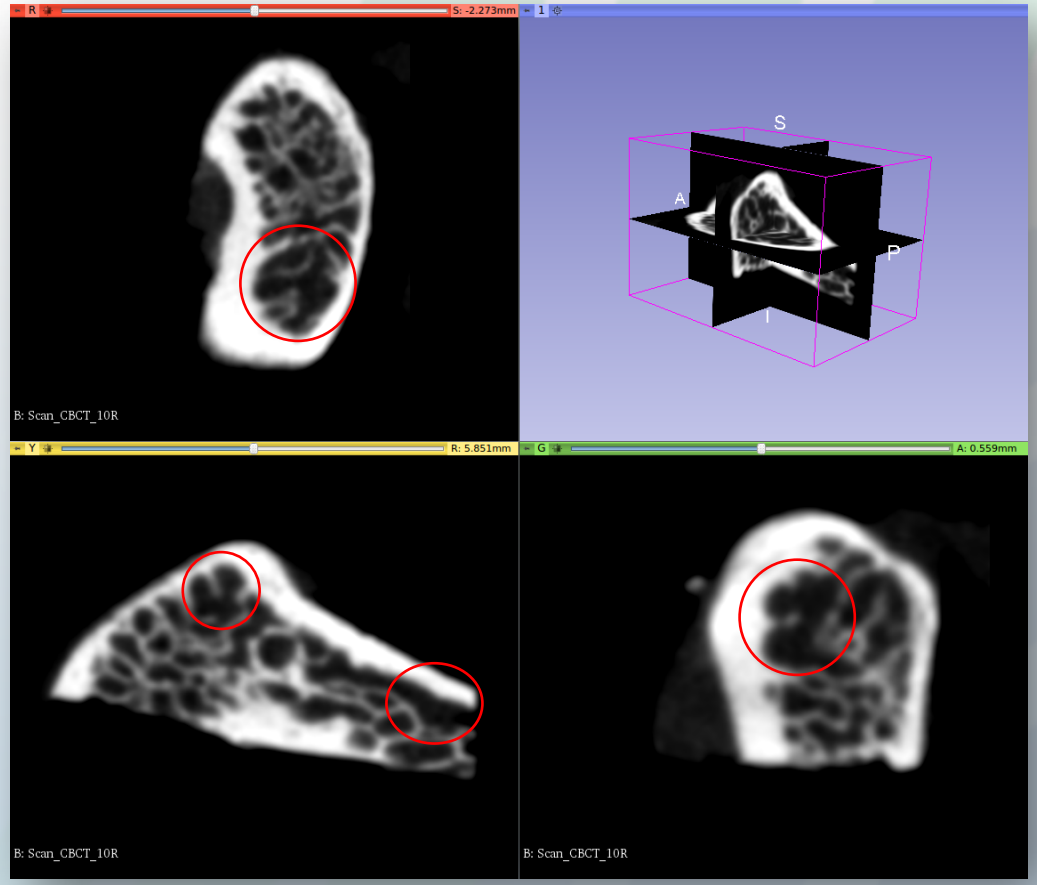
The best parameter set specific to your study case might be different.

Default parameters

Detection of early stage of Osteoarthritis (OA) in Temporomendibular Joints (TMJ):

For our particular study case, we are trying to detect early stage of TMJOA, which is characterized by a lack of trabecula in the bone texture (i.e. holes in the bone texture).

In our input data we can identify four different places where the bone seems to be diseased.

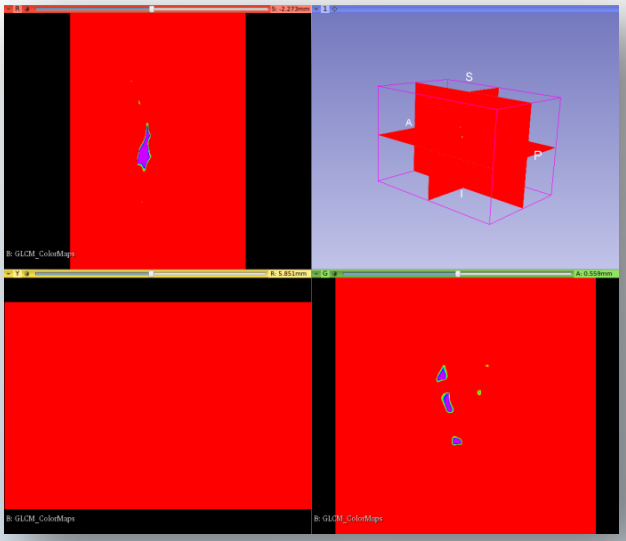


Kitware

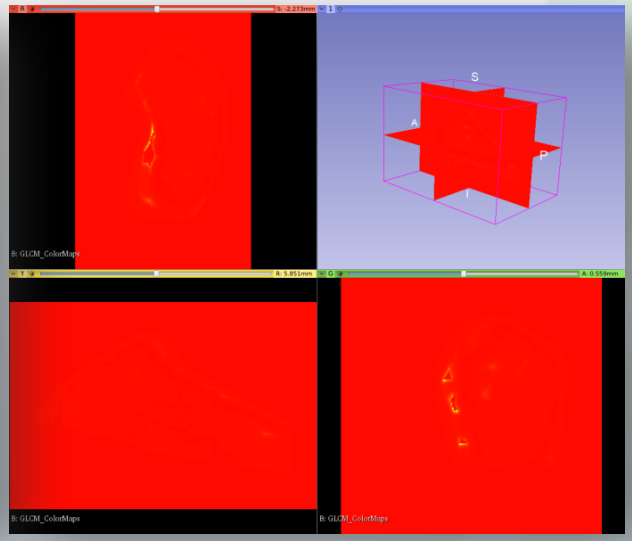
Default parameters

After running both algorithms (GLCM and GLRLM) with the default parameter set, it appears that:

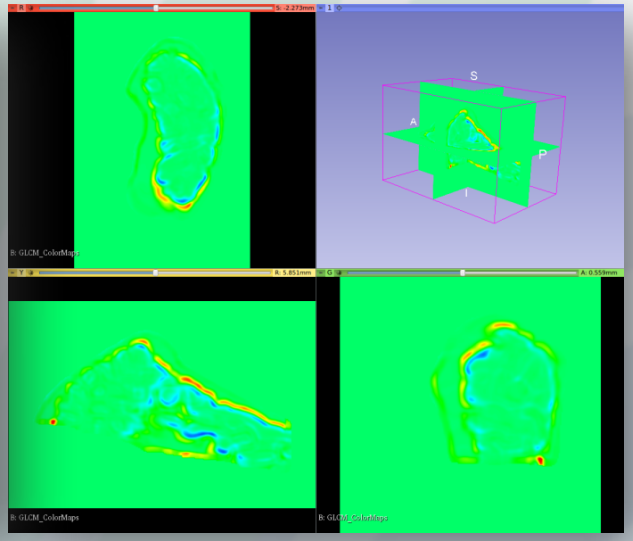
Some texture maps does not give any information



inverse difference moment



correlation

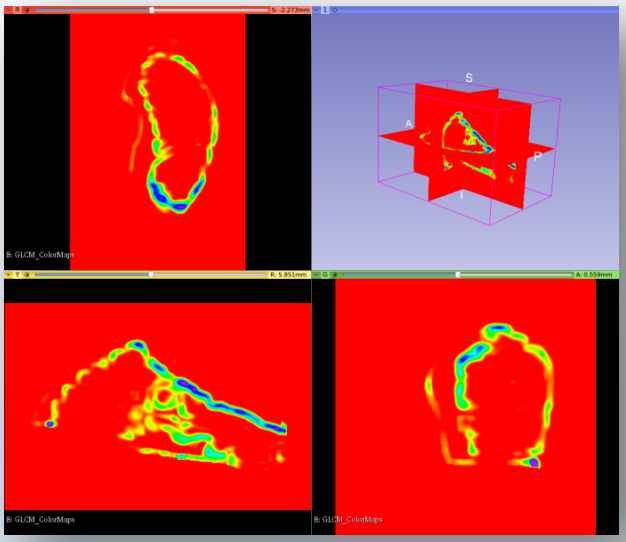


cluster shade

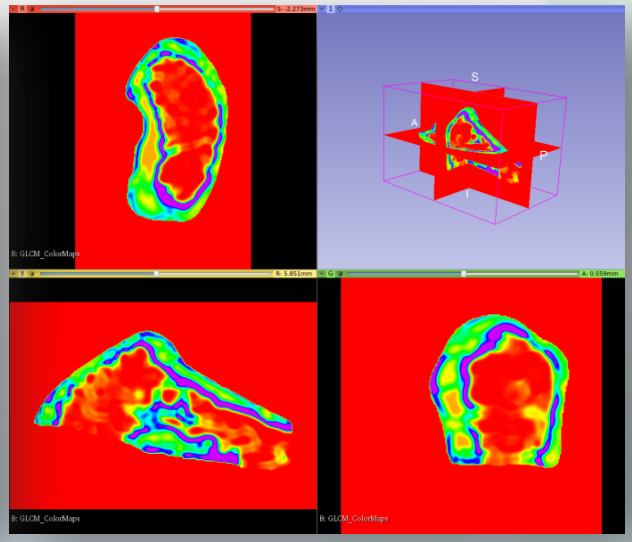
Default parameters

After running both algorithms (GLCM and GLRLM) with the default parameter set, it appears that:

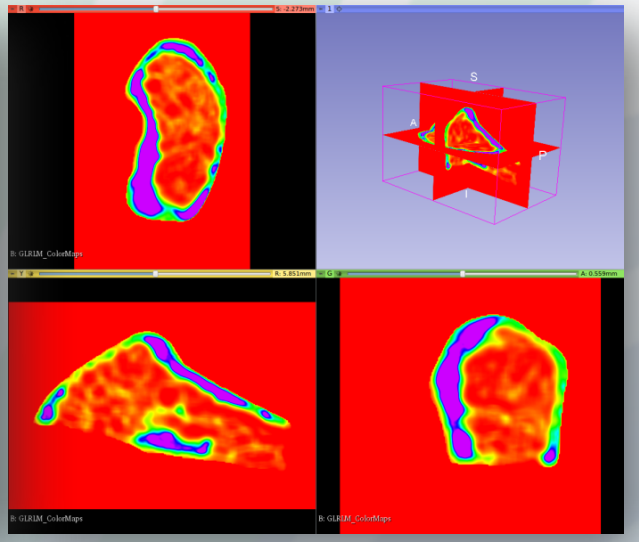
- Some others does give information, but they are not useful in our case



cluster prominence



Haralick correlation

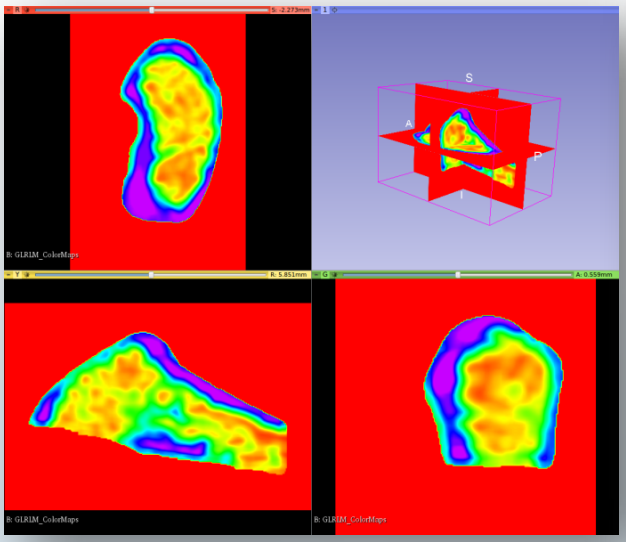


Long run high gray level emphasis

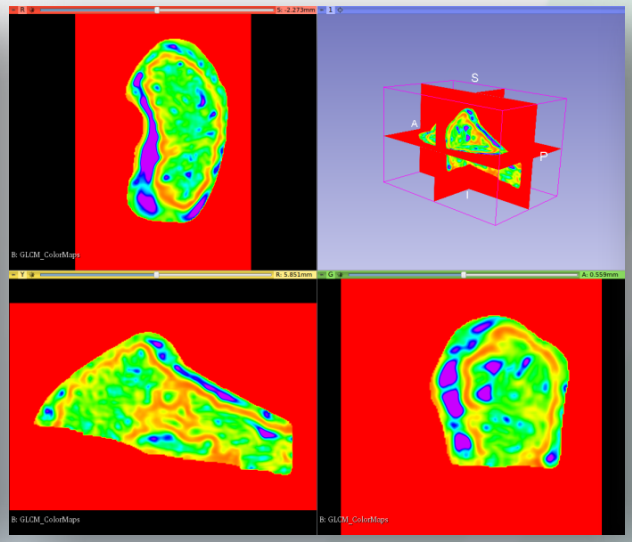
Default parameters

After running both algorithms (GLCM and GLRLM) with the default parameter set, it appears that:

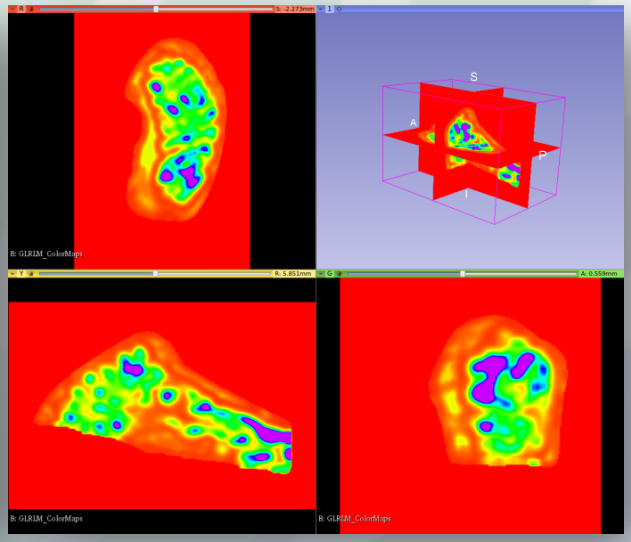
And some give us information we could exploit



high gray level run emphasis



energy



long run low gray level emphasis

Mask

For this experiment we used the default parameter set, but *WITHOUT* any mask:

The computation time is significantly higher: 3:18 for the GLCM algorithm, 6:17 for the GLRLM algorithm

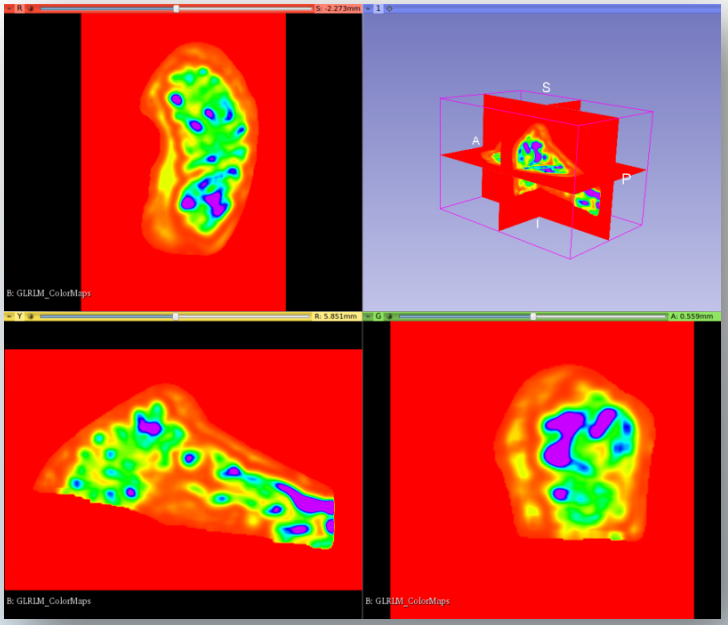
For the voxels that were ''inside the mask'' the output values are the exact same ones

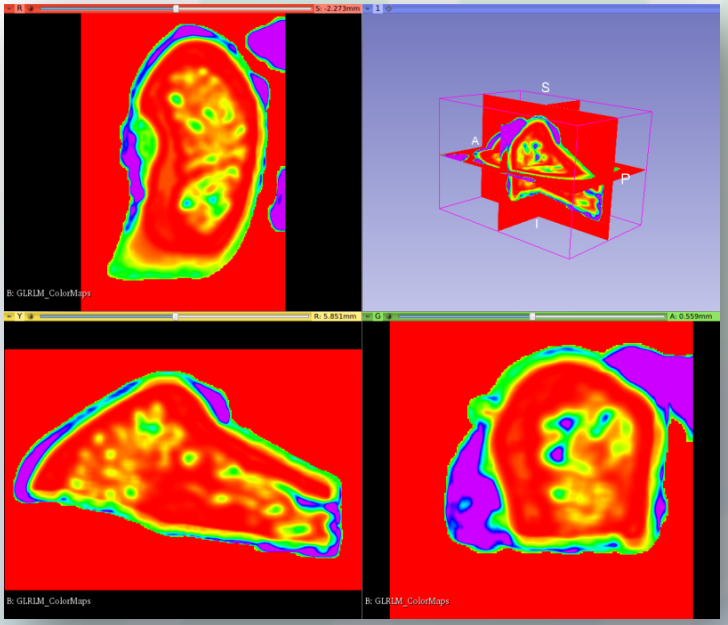
For the voxels that were on the boundaries of the mask the values are slightly different

For the voxels that were outside of the mask, new values have been computed

Mask

If we compare the Long Run Low Grey Level Emphasis feature maps with (left) and without (right) a mask:





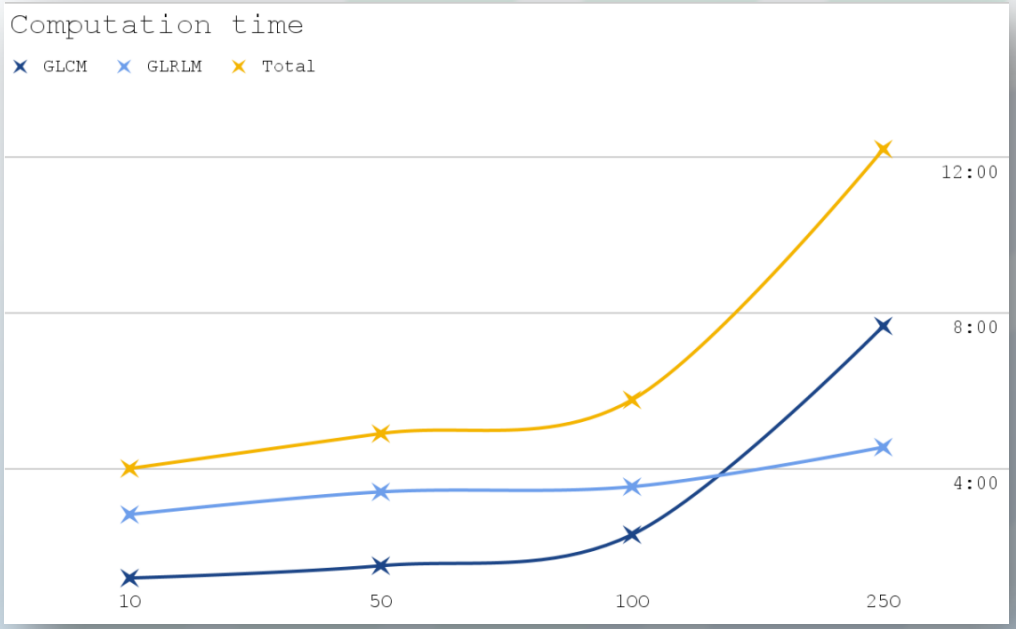
It looks like the feature maps computed without a mask is not giving the same information even for the voxels that were inside the mask. Therefore, they contain the same information: the only difference is the value range of the texture map that is larger due to a new maximum value computed in a part outside of the mask. A larger range leads to less contrast, but the information are still the same.

Number of bins

In addition to the default value of 10 bins, we also used 50, 100 and 250 bins:

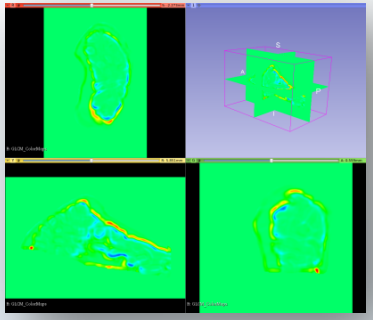
The computation time is increasing with the number of bins

	10	50	100	250
GLCM	1:11	1:30	2:18	7:40
GLRLM	2:49	3:24	3:32	4:33
Total	4:00	4:54	5:46	12:13

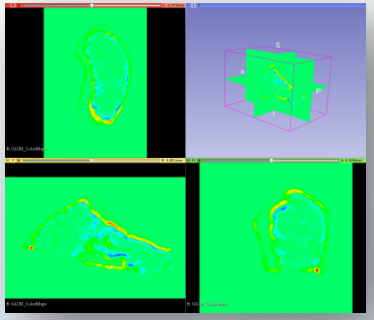


Number of bins

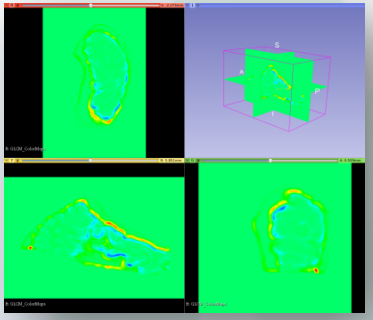
Some texture feature maps doesn't seems to be affected by the number of bins used: cluster prominence, cluster shade (illustration), correlation



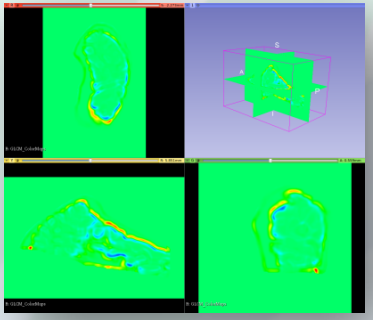
10 bins



50 bins



100 bins

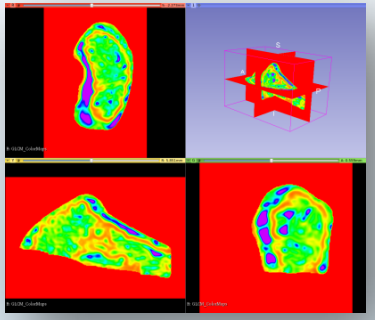


250 bins

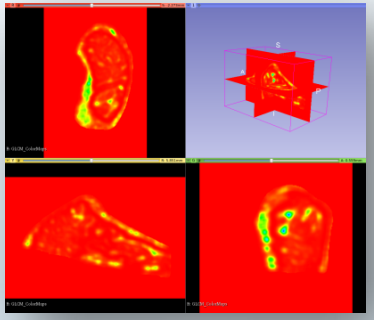
Number of bins

Some other texture feature maps give us different informations when increasing the number of bins:

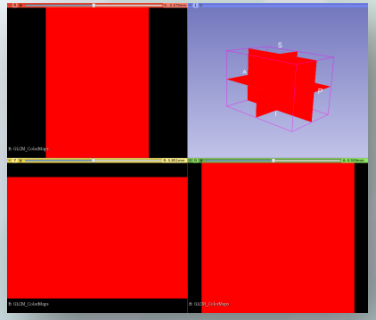
The energy for example seems to give us no information for 50 or 100 bins and a totally different one for 250 bins



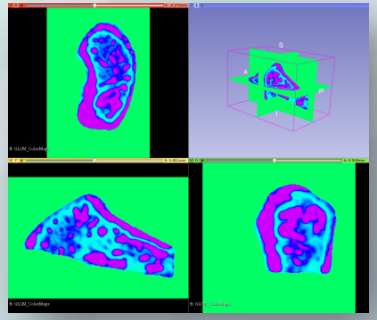
10 bins



50 bins



100 bins

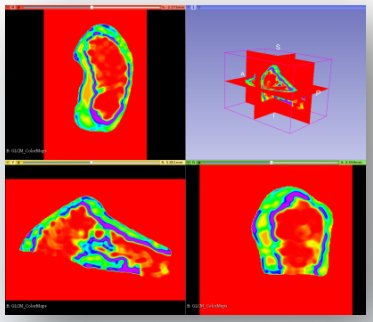


250 bins

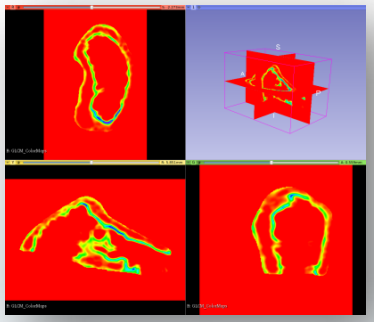
Kitware

Number of bins

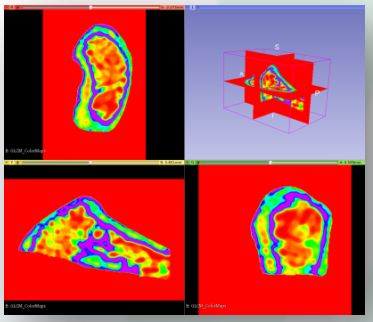
Same with The Harralick Correlation that gives us more information for 100 or 250 bins



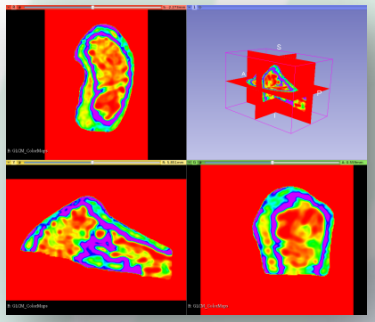
10 bins



50 bins

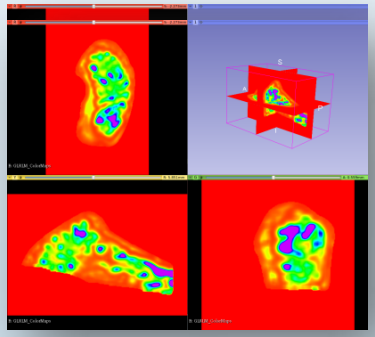


100 bins

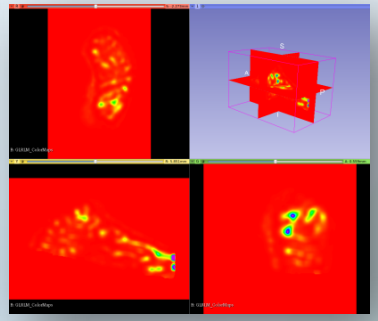


250 bins

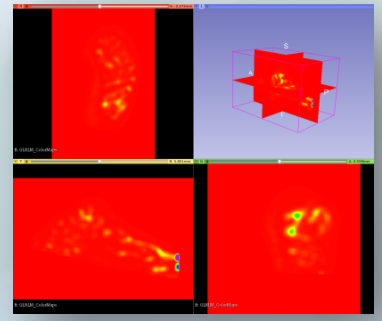
On the other side, some feature maps like the Long Run Low Grey Level Emphasis gives us fewer informations when the number of bins increase



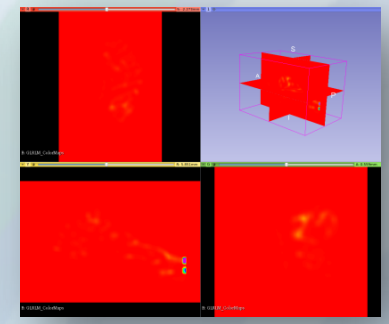
10 bins



50 bins



100 bins



250 bins

Kitware

Voxel Intensity Range

In addition to the default 0-4500 intensity range we also used 0-2000, 0-8000 and 1500-4500 value ranges:

The computation time doesn't seem to be related to the voxel value range size in a linear way. The computation time will mainly depend of how the

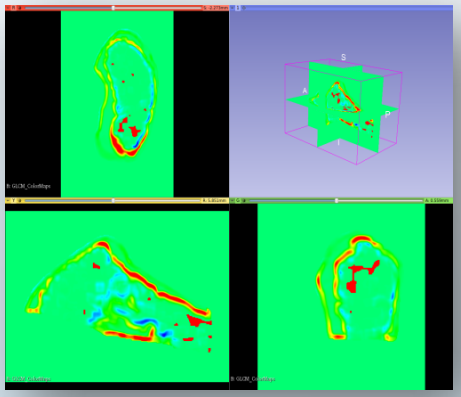
	0 - 2000	1500 - 4500	0 - 4500	0 - 8000
GLCM	0:42	0:49	1:11	1:09
GLRLM	1:37	2:00	2:49	2:28
Total	2:19	2:49	4:00	3:38

Voxel Intensity Range

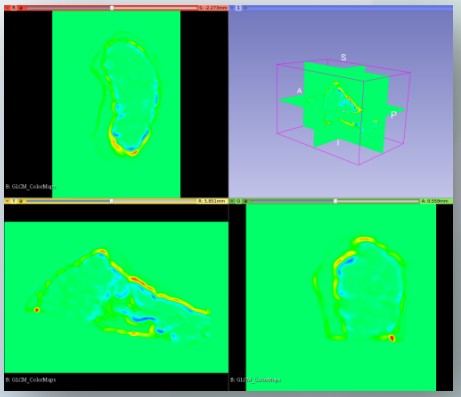
Choosing correctly the value range is very important as all the voxels out of this range will not be taken into account.

In our study case the input image as a value range in between -777 and 4276, therefore all the values contained in the mask are positive (the negative values represent noise in the background of the image), that's why we choose a range of 0 - 4500.

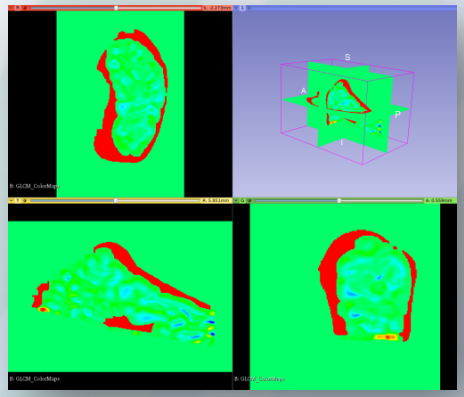
Choosing range like 0 - 2000, or 1500 - 4500 that will exclude a part of the useful data will lead to ''holes'' in the textures maps. They are particularly visible in the Cluster Shade feature maps as they appears in red.



1500-4500



0-4500

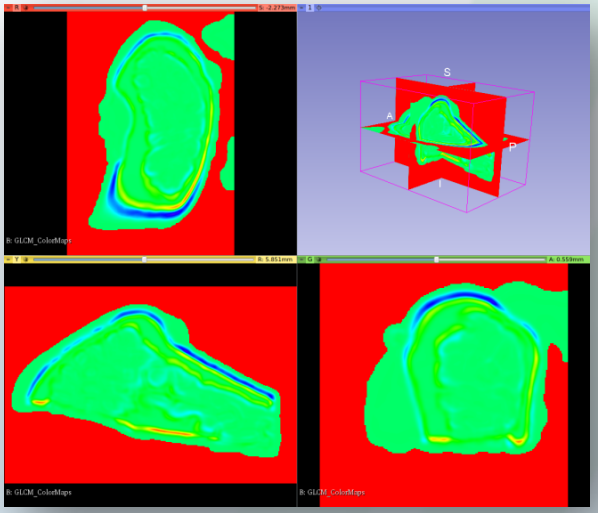


0-2000

Cluster Shade

Voxel Intensity Range

Therefore, excluding some voxel can be useful, particularly if they represent noise in the image. That's actually what happened in the computation without mask, the noisy part of the image hasn't been computed (red part) which improved the computation time.

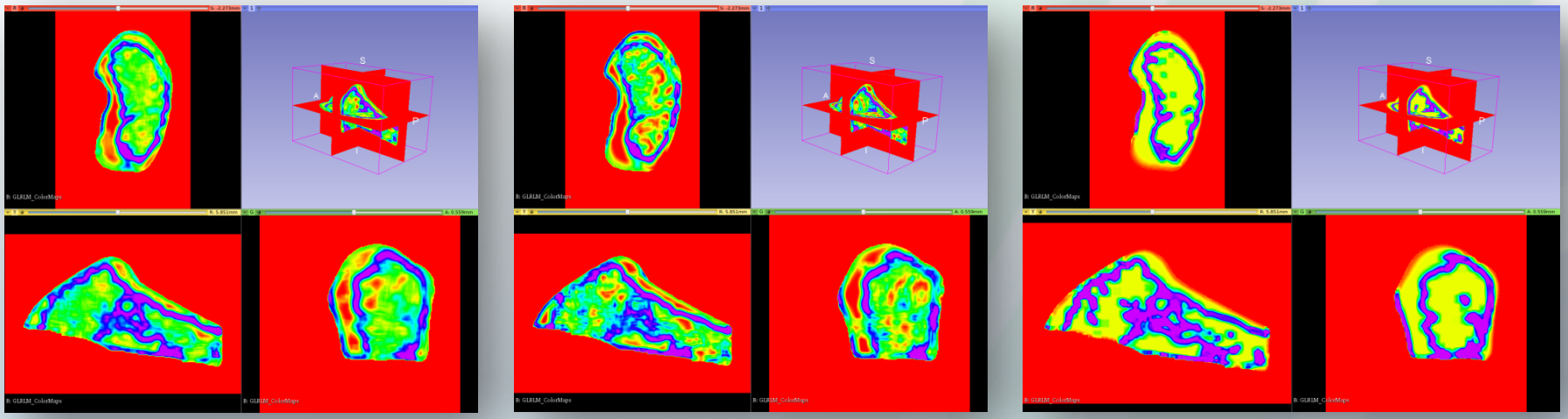


0-4500 without mask
Cluster Shade

Voxel Intensity Range

Increasing the voxel intensity range, even without noise in the image, will modify the output texture maps: in fact it will modify the repartition of the bins and the construction of the matrices.

In our case, the modification is not that significant with the 0 - 8000 range, but the features map are not exploitable anymore with a 0 - 20000 range



0-4500 0-8000 0-20000

Run Length Non Uniformity

Distance Range

In addition to the default 0.0-1.25 distance range we also used 0.0-0.5, 0.0-2.0 and 0.5-1.25 ranges:

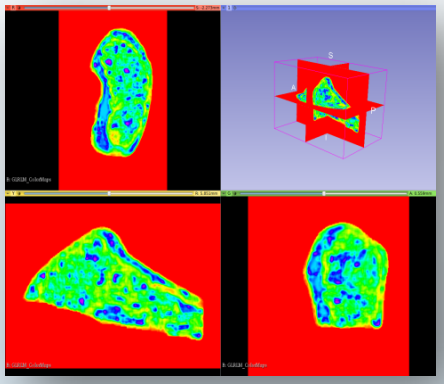
The computation time seems to be totally independant from the distance range

	0.5 - 1.25	0.0 - 0.5	0.0 - 1.25	0.0 - 2.0
GLRLM	2:38	2:35	2:49	2:36

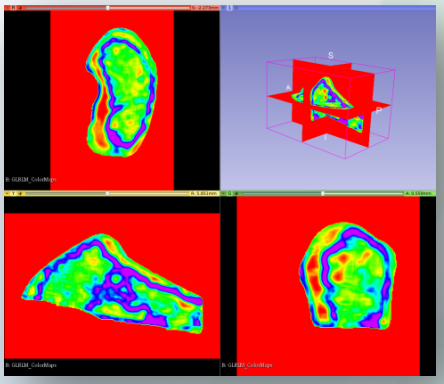
Distance Range

In our study case we chose to take into account the run of all lengths, the choice of 1.25 is then motivated by the fact that we have a neighborhood radius of 4 (which mean squares of 9 voxels for each side) and a spacing of 0.08mm in each direction: the longest run possible will be $\sqrt{(3\times(0.08\times9)^3)}\approx1.05$

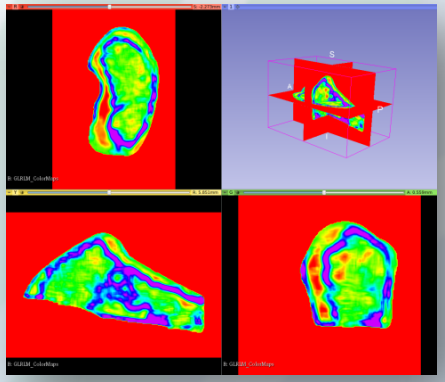
Therefore, excluding runs that are too small or too big might be a good idea depending on what you want to find in your image.



0.25-1.25



0-1.25



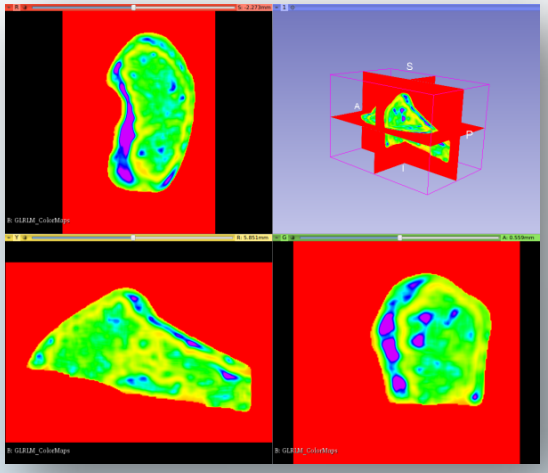
0-0.5

Run Length Non Uniformity

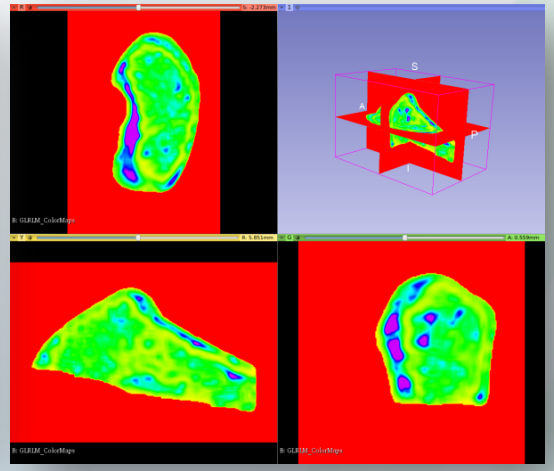
Distance Range

Similarly to the Voxel Intensity range, increasing the distance range, will modify the output texture maps: in fact it will modify the repartition of the bins and the construction of the matrices.

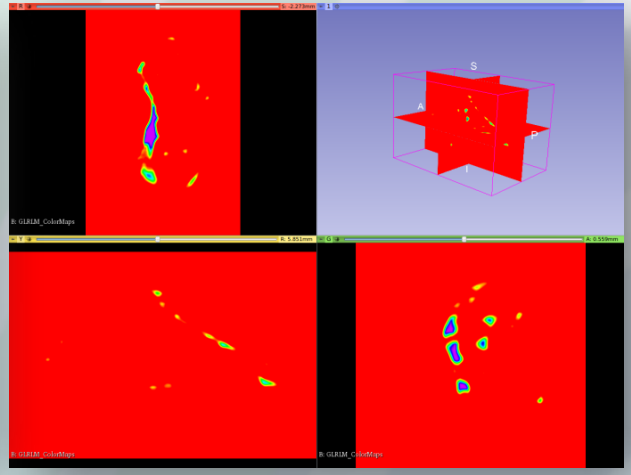
In our case, the modification is not that significant with the 0.0 - 2.0 range, but the features map are not exploitable anymore with a 0 - 5.0 range



0-1.25



0-2.0



0-5.0

Long Run Emphasis

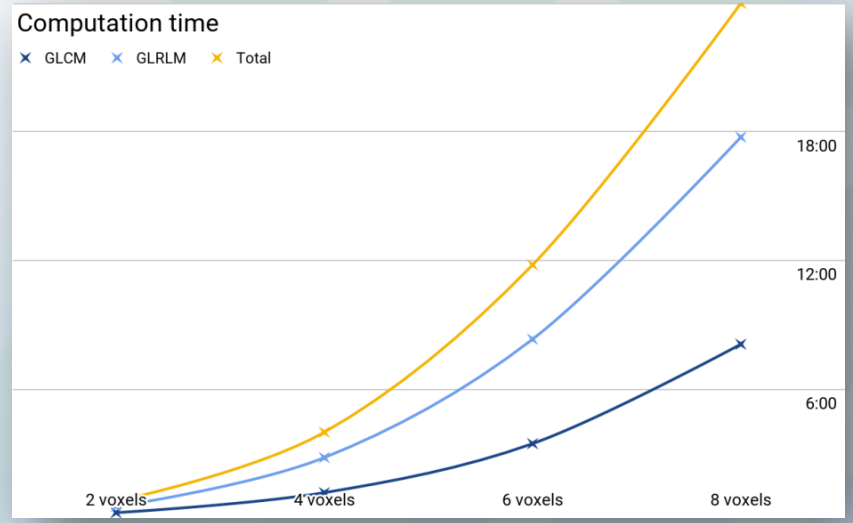
Neighborhood Radius

In addition to the default 4 neighborhood radius we also used 2, 6 and 8 radiuses:

In our case the the distance range have been adapted to the size of the neighborhood in order to take all the runs into acount.

The computation time is increasing with the neighborhood radius (as we have seen before that the distance range does not affect the computation time)

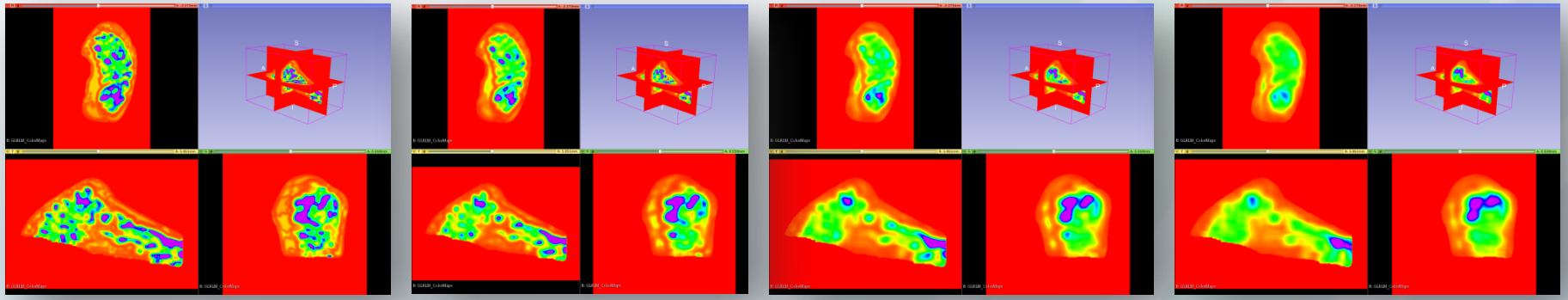
	2	4	6	8
GLCM	0:15	1:11	3:28	8:06
GLRLM	0:32	2:49	8:20	17:45
Total	0:47	4:00	11:48	25:53



Neighborhood Radius

The size of the neighborhood will have a lot of impact on the output texture maps

This parameter should be defined depending on the user's goal: the type of features that need to be observed and particularly their size.



2 4 6 8

Long Run High Grey Level Emphasis

Acknowledgement -Resources - Contact

This work was supported by the National Institute of Health (NIH) National Institute for Dental and Craniofacial Research (NIDCR) R01EB021391 (Textural Biomarkers of Arthritis for the Subchondral Bone in the Temporomandibular Joint)

Github repositories:

- 3DSlicer
- itkTextureFeatures
- BonTextureExtension

Slicer Wiki:

- BonTextureExtension

Slicer Forum:

- Discourse

Article:

- itkTextureFeatures Insight Journal Article

Data:

- Input/Output data used in this presentation

For other remarks or questions please email:

- jb.vimort@kitware.com

Kitware