MedPy

b0.1

Generated by Doxygen 1.7.4

Tue Dec 13 2011 19:01:34

Contents

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 2

Class Index

2.1 Class List

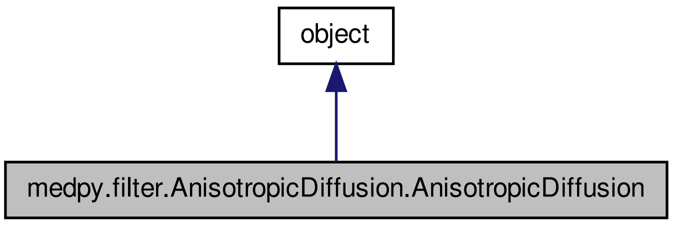
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

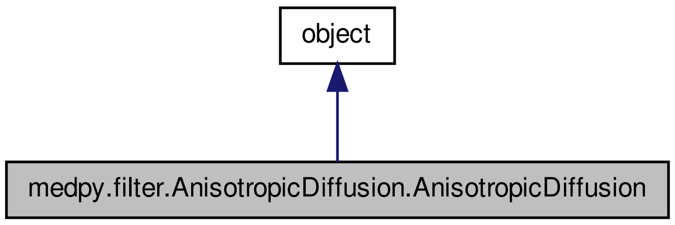
Class Documentation

3.1 medpy.filter.AnisotropicDiffusion.AnisotropicDiffusion Class Reference

Inheritance diagram for medpy.filter.AnisotropicDiffusion.AnisotropicDiffusion:



Collaboration diagram for medpy.filter.AnisotropicDiffusion.AnisotropicDiffusion:



Public Member Functions

- def **__init__**
- def **anisotropic_diffusion**

Private Member Functions

- def **_exp**
- def **_inv**

3.1.1 Detailed Description

Definition at line 24 of file AnisotropicDiffusion.py.

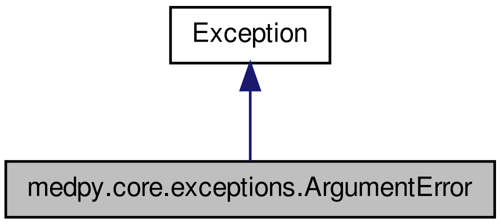
The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/filter/AnisotropicDiffusion.py

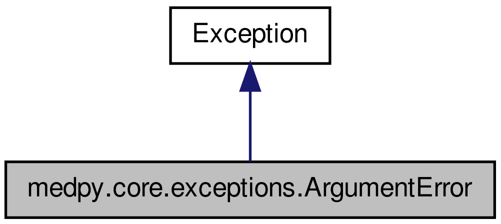
3.2 medpy.core.exceptions.ArgumentError Class Reference

Thrown by an application when an invalid command line argument has been supplied.

Inheritance diagram for medpy.core.exceptions.ArgumentError:



Collaboration diagram for medpy.core.exceptions.ArgumentError:



3.2.1 Detailed Description

Thrown by an application when an invalid command line argument has been supplied.

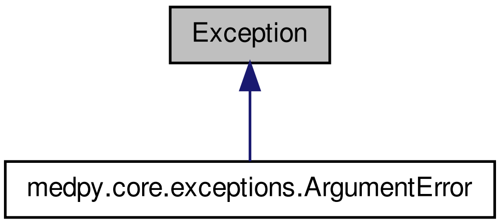
Definition at line 28 of file exceptions.py.

The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/core/exceptions.py

3.3 Exception Class Reference

Inheritance diagram for Exception:



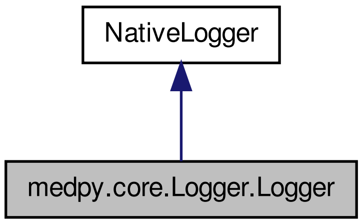
The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/core/exceptions.py

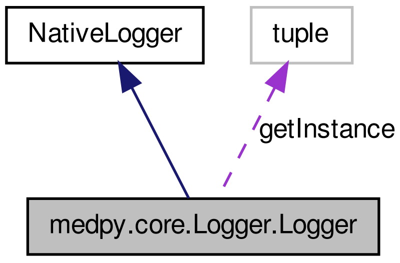
3.4 medpy.core.Logger.Logger Class Reference

This singleton class represents an object that can be used by all applications and classes.

Inheritance diagram for medpy.core.Logger.Logger:



Collaboration diagram for medpy.core.Logger.Logger:



Classes

- class LoggerHelper

  A helper class which performs the actual initialization.

Public Member Functions

- def **__init__**
- def setHandler

  Replaces the current handler with a new one.
- def setLevel

  Overrides the parent method to adapt the formatting string to the level.

Static Public Attributes

- tuple **getInstance** = LoggerHelper()

Static Private Attributes

- **_instance** = None
- **_handler** = None

3.4.1 Detailed Description

This singleton class represents an object that can be used by all applications and classes.

Definition at line 34 of file Logger.py.

3.4.2 Member Function Documentation

3.4.2.1 def medpy.core.Logger.Logger.setHandler (*self, hdlr*)

Replaces the current handler with a new one.

If none should be replaces, but just one added, use the parent classes addHandler() method.

Definition at line 76 of file Logger.py.

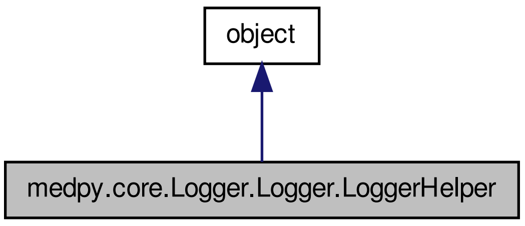
The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/core/Logger.py

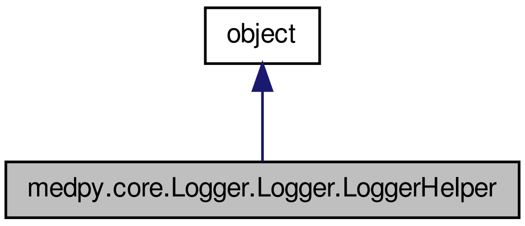
3.5 medpy.core.Logger.Logger.LoggerHelper Class Reference

A helper class which performs the actual initialization.

Inheritance diagram for medpy.core.Logger.Logger.LoggerHelper:



Collaboration diagram for medpy.core.Logger.Logger.LoggerHelper:



Public Member Functions

- def **__call__**

3.5.1 Detailed Description

A helper class which performs the actual initialization.

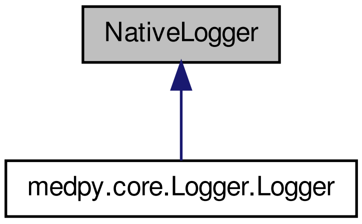
Definition at line 40 of file Logger.py.

The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/core/Logger.py

3.6 NativeLogger Class Reference

Inheritance diagram for NativeLogger:

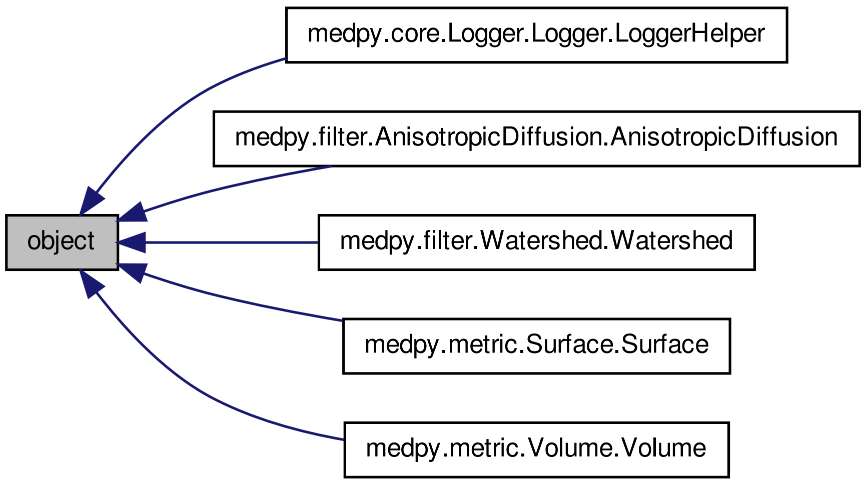


The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/core/Logger.py

3.7 object Class Reference

Inheritance diagram for object:


medpy.core.Logger.Logger.LoggerHelper
medpy.filter.AnisotropicDiffusion.AnisotropicDiffusion
object
medpy.filter.Watershed.Watershed
medpy.metric.Surface.Surface
medpy.metric.Volume.Volume

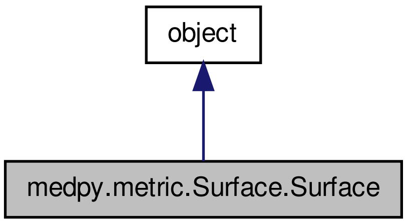
The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/filter/AnisotropicDiffusion.py

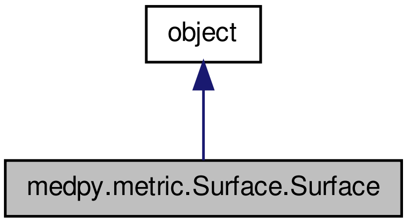
3.8 medpy.metric.Surface.Surface Class Reference

Computes different surface metrics between two 3D-images contain each an object.

Inheritance diagram for medpy.metric.Surface.Surface:



Collaboration diagram for medpy.metric.Surface.Surface:



Public Member Functions

- def __init__

  Initialize the class with two binary images, each containing a single object.
- def GetMaximumSymmetricSurfaceDistance

  Computes the maximum symmetric surface distance, also known as Hausdorff distance, between the two objects surfaces.
- def GetRootMeanSquareSymmetricSurfaceDistance

  Computes the root mean square symmetric surface distance between the two objects surfaces.
- def GetAverageSymmetricSurfaceDistance

  Computes the average symmetric surface distance between the two objects surfaces.
- def GetMaskReferenceNn

- def GetReferenceMaskNn
- def GetMaskEdgePoints
- def GetReferenceEdgePoints

- def ComputeContour

  Uses a 18-neighbourhood filter to create an edge image of the input object.

Static Private Attributes

- **_mask_edge_points** = None
- **_reference_edge_points** = None
- **_mask_reference_nn** = None
- **_reference_mask_nn** = None
- **_distance_matrix** = None

3.8.1 Detailed Description

Computes different surface metrics between two 3D-images contain each an object.

The surface of the objects is computed using a 18-neighbourhood edge detection. The distance metrics are computed over all points of the surfaces using the nearest neighbour approach. Beside this provides a number of statistics of the two images.

During the initialization the edge detection is run for both images, taking up to 5 min (on 512^3 images). The first call to one of the metric measures triggers the computation of the nearest neighbours, taking up to 7 minutes (based on 250.000 edge point for each of the objects, which corresponds to a typical liver mask). All subsequent calls to one of the metrics measures can be expected be in the sub-millisecond area.

Metrics defined in: Heimann, T.; van Ginneken, B.; Styner, M.A.; Arzhaeva, Y.; Aurich, V.; Bauer, C.; Beck, A.; Becker, C.; Beichel, R.; Bekes, G.; Bello, F.; Binnig, G.; Bischof, H.; Bornik, A.; Cashman, P.; Ying Chi; Cordova, A.; Dawant, B.M.; Fidrich, M.; Furst, J.D.; Furukawa, D.; Grenacher, L.; Hornegger, J.; Kainmuller, D.; Kitney, R.I.; Kobatake, H.; Lamecker, H.; Lange, T.; Jeongjin Lee; Lennon, B.; Rui Li; Senhu Li; Meinzer, H.-P.; Nemeth, G.; Raicu, D.S.; Rau, A.-M.; van Rikxoort, E.M.; Rousson, M.; Rusko, L.; Saddi, K.A.; Schmidt, G.; Seghers, D.; Shimizu, A.; Slagmolen, P.; Sorantin, E.; Soza, G.; Susomboon, R.; Waite, J.M.; Wimmer, A.; Wolf, I.; , "Comparison and Evaluation of Methods for Liver Segmentation From CT Datasets," Medical Imaging, IEEE Transactions on , vol.28, no.8, pp.1251-1265, Aug. 2009 doi: 10.1109/TMI.2009.2013851

```
The edge points of the mask object.
```

Definition at line 44 of file Surface.py.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 def medpy.metric.Surface.Surface.__init__ (*self*, *mask*, *reference*, *physical_voxel_spacing* = `[1`, *mask_offset* = `[0`, *reference_offset* = `[0`)

Initialize the class with two binary images, each containing a single object.

Assumes the input to be a representation of a 3D image, that fits one of the following formats: 1. all 0 values denoting background, all others the foreground/object 2. all False values denoting the background, all others the foreground/object The first image passed is referred to as 'mask', the second as 'reference'. This is only important for some metrics that are not symmetric (and therefore not really metrics).

Parameters

mask,:	binary mask as an scipy array (3D image)
reference,:	binary reference as an scipy array (3D image)
physical_-voxel_-spacing,:	The physical voxel spacing of the two images (must be the same for both)
mask_-offset,:	offset of the mask array to 0,0,0-origin
reference_-offset,:	offset of the reference array to 0,0,0-origin

Definition at line 74 of file Surface.py.

3.8.3 Member Function Documentation

3.8.3.1 def medpy.metric.Surface.Surface.ComputeContour (*array*)

Uses a 18-neighbourhood filter to create an edge image of the input object.

Assumes the input to be a representation of a 3D image, that fits one of the following formats: 1. all 0 values denoting background, all others the foreground/object 2. all False values denoting the background, all others the foreground/object The area outside the array is assumed to contain background voxels. The method does not ensure that the object voxels are actually connected, this is silently assumed.

Parameters

array,:	a numpy array with only 0/N0} or False/True values.

Returns

: a boolean numpy array with the input objects edges

Definition at line 295 of file Surface.py.

3.8.3.2 def medpy.metric.Surface.Surface.GetAverageSymmetricSurfaceDistance (*self*)

Computes the average symmetric surface distance between the two objects surfaces.

Returns

: average symmetric surface distance in millimeters

For a perfect segmentation this distance is 0.

Metric definition: Let $S(A)$ denote the set of surface voxels of A. The shortest distance of an arbitrary voxel v to $S(A)$ is defined as:

$$d(v, S(A)) = \min_{s_A \in S(A)} ||v - s_A||$$

where $||.||$ denotes the Euclidean distance. The average symmetric surface distance is then given by:

$$ASD(A,B) = \frac{1}{|S(A)| + |S(B)|}\left(\sum_{s_A \in S(A)} d(s_A, S(B)) + \sum_{s_B \in S(B)} d(s_B, S(A))\right)$$

Definition at line 218 of file Surface.py.

3.8.3.3 def medpy.metric.Surface.Surface.GetMaskEdgePoints (*self*)

Returns

: The edge points of the mask object.

Definition at line 270 of file Surface.py.

3.8.3.4 def medpy.metric.Surface.Surface.GetMaskReferenceNn (*self*)

Returns

: The distances of the nearest neighbours of all mask edge points to all reference edge points.

Definition at line 239 of file Surface.py.

3.8.3.5 def medpy.metric.Surface.Surface.GetMaximumSymmetricSurfaceDistance (*self*)

Computes the maximum symmetric surface distance, also known as Hausdorff distance, between the two objects surfaces.

Returns

: the maximum symmetric surface distance in millimeters

For a perfect segmentation this distance is 0. This metric is sensitive to outliers and returns the true maximum error.

Metric definition: Let $S(A)$ denote the set of surface voxels of A. The shortest distance of an arbitrary voxel v to $S(A)$ is defined as:

$$d(v,S(A)) = \min_{s_A \in S(A)} ||v - s_A||$$

where $||.||$ denotes the Euclidean distance. The maximum symmetric surface distance is then given by:

$$MSD(A,B) = \max\left\{ \max_{s_A \in S(A)} d(s_A,S(B)), \max_{s_B \in S(B)} d(s_B,S(A)), \right\}$$

Definition at line 133 of file Surface.py.

3.8.3.6 def medpy.metric.Surface.Surface.GetReferenceEdgePoints (*self*)

Returns

: The edge points of the reference object.

Definition at line 277 of file Surface.py.

3.8.3.7 def medpy.metric.Surface.Surface.GetReferenceMaskNn (*self*)

Returns

: The distances of the nearest neighbours of all reference edge points to all mask edge points.

The underlying algorithm used for the scipy.spatial.KDTree implementation is based on: Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. 1998. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. J. ACM 45, 6 (November 1998), 891-923

Definition at line 257 of file Surface.py.

3.8.3.8 def medpy.metric.Surface.Surface.GetRootMeanSquareSymmetricSurfaceDistance (*self*)

Computes the root mean square symmetric surface distance between the two objects surfaces.

Returns

: root mean square symmetric surface distance in millimeters

For a perfect segmentation this distance is 0. This metric punishes large deviations from the true contour stronger than the average symmetric surface distance.

Metric definition: Let $S(A)$ denote the set of surface voxels of A. The shortest distance of an arbitrary voxel v to $S(A)$ is defined as:

$$d(v, S(A)) = \min_{s_A \in S(A)} ||v - s_A||$$

where $||.||$ denotes the Euclidean distance. The root mean square symmetric surface distance is then given by:

$$RMSD(A,B) = \sqrt{\frac{1}{|S(A)|+|S(B)|}} \times \sqrt{\sum_{s_A \in S(A)} d^2(s_A, S(B)) + \sum_{s_B \in S(B)} d^2(s_B, S(A))}$$

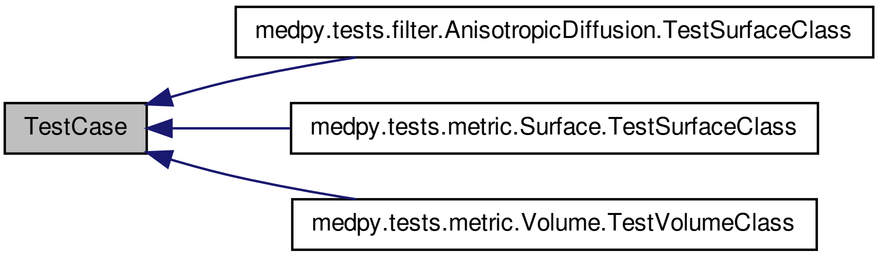
Definition at line 171 of file Surface.py.

The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/metric/Surface.py

3.9 TestCase Class Reference

Inheritance diagram for TestCase:

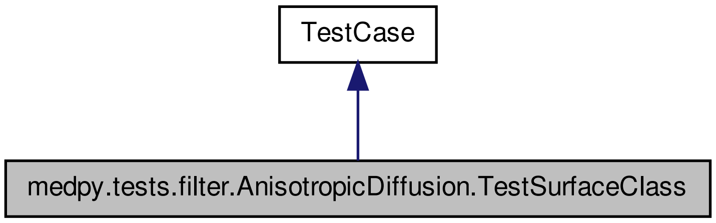

medpy.tests.filter.AnisotropicDiffusion.TestSurfaceClass
TestCase
medpy.tests.metric.Surface.TestSurfaceClass
medpy.tests.metric.Volume.TestVolumeClass

The documentation for this class was generated from the following file:

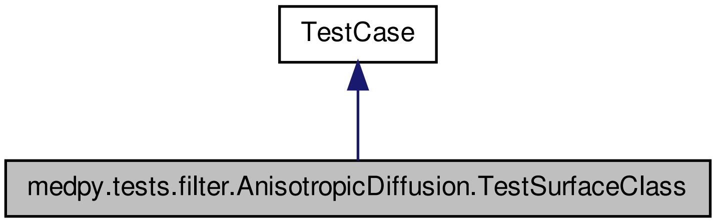
- /home/loli/Programming/Python/MedPy/src/medpy/tests/metric/Surface.py

3.10 medpy.tests.filter.AnisotropicDiffusion.TestSurfaceClass Class Reference

Inheritance diagram for medpy.tests.filter.AnisotropicDiffusion.TestSurfaceClass:



Collaboration diagram for medpy.tests.filter.AnisotropicDiffusion.TestSurfaceClass:



Public Member Functions

- def **test_diffusion**

3.10.1 Detailed Description

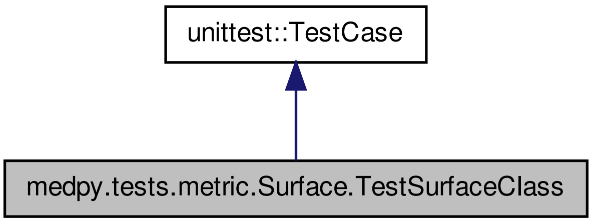
Definition at line 23 of file AnisotropicDiffusion.py.

The documentation for this class was generated from the following file:

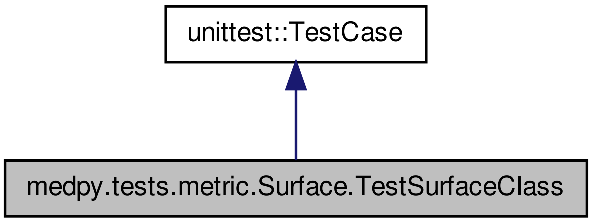
- /home/loli/Programming/Python/MedPy/src/medpy/tests/filter/AnisotropicDiffusion.py

3.11 medpy.tests.metric.Surface.TestSurfaceClass Class Reference

Inheritance diagram for medpy.tests.metric.Surface.TestSurfaceClass:



Collaboration diagram for medpy.tests.metric.Surface.TestSurfaceClass:



Public Member Functions

- def **setUp**
- def **test_Initialization**
- def **test_ComputeContour**
- def **test_GetMaximumSymmetricSurfaceDistance**
- def **test_GetAverageSymmetricSurfaceDistance**
- def **test_GetRootMeanSquareSymmetricSurfaceDistance**

Private Attributes

- **_imaged1**
- **_imaged2**
- **_imaged3**
- **_imaged4**
- **_imagedA**
- **_imagedB**
- **_imagec**

3.11.1 Detailed Description

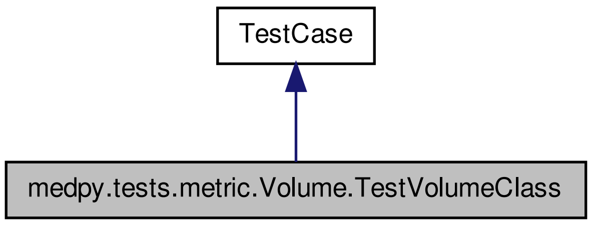
Definition at line 23 of file Surface.py.

The documentation for this class was generated from the following file:

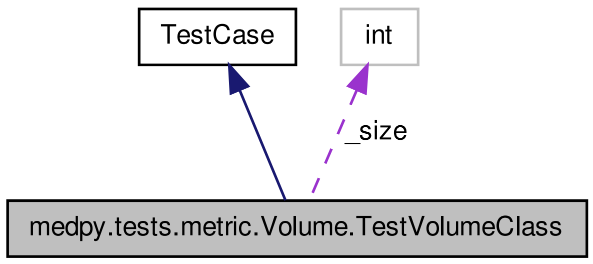
- /home/loli/Programming/Python/MedPy/src/medpy/tests/metric/Surface.py

3.12 medpy.tests.metric.Volume.TestVolumeClass Class Reference

Inheritance diagram for medpy.tests.metric.Volume.TestVolumeClass:



Collaboration diagram for medpy.tests.metric.Volume.TestVolumeClass:



Public Member Functions

- def **test_GetVolumetricOverlapError**
- def **test_GetRelativeVolumeDifference**

Private Member Functions

- def _createTestImages

  Creates some images used in the tests.

Static Private Attributes

- int **_size** = 24

3.12.1 Detailed Description

Definition at line 23 of file Volume.py.

3.12.2 Member Function Documentation

3.12.2.1 def medpy.tests.metric.Volume.TestVolumeClass._createTestImages (*self*) `[private]`

Creates some images used in the tests.

Definition at line 92 of file Volume.py.

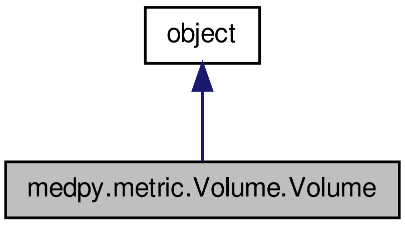
The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/tests/metric/Volume.py

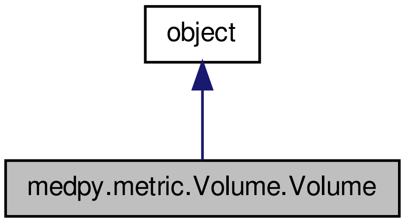
3.13 medpy.metric.Volume.Volume Class Reference

Computes different volume metrics between two 3D-images contain each a binary object.

Inheritance diagram for medpy.metric.Volume.Volume:



Collaboration diagram for medpy.metric.Volume.Volume:



Public Member Functions

- def __init__

  Initialize the class with two binary images whose 0 values are considered to be background voxels and the 1 values foreground.

- def GetRelativeVolumeDifference

  Returns the relative volume difference between two shapes defined by the non-background voxels of in the binary images.
- def GetVolumetricOverlapError

  Returns the volumetric overlap error between two shapes defined by the non-background voxels of in two binary images.
- def GetMaskSize

  Computes the mask objects size if not yet done and returns it.
- def GetReferenceSize

  Computes the reference objects size if not yet done and returns it.
- def GetIntersectionSize

  Computes the two objects intersection size if not yet done and returns it.
- def GetUnionSize

  Computes the two objects union size if not yet done and returns it.

Private Member Functions

- def _ComputeIntersectionWindow

  Computes and sets the intersection windows of the two images.
- def _ComputeUnionAndIntersection

  Computes and sets the two image shapes union and intersection sizes.

Static Private Attributes

- **_mask** = None
- **_reference** = None
- **_mask_offset** = None
- **_reference_offset** = None
- **_intersection_window** = None
- **_mask_size** = None
- **_reference_size** = None
- **_union_size** = None
- **_intersection_size** = None

3.13.1 Detailed Description

Computes different volume metrics between two 3D-images contain each a binary object.

Beside this provides a number of statistics of the two images.

Metrics defined in: Heimann, T.; van Ginneken, B.; Styner, M.A.; Arzhaeva, Y.; Aurich, V.; Bauer, C.; Beck, A.; Becker, C.; Beichel, R.; Bekes, G.; Bello, F.; Binnig, G.; Bischof, H.; Bornik, A.; Cashman, P.; Ying Chi; Cordova, A.; Dawant, B.M.; Fidrich, M.; Furst, J.D.; Furukawa, D.; Grenacher, L.; Hornegger, J.; Kainmuller, D.; Kitney, R.I.; Kobatake, H.; Lamecker, H.; Lange, T.; Jeongjin Lee; Lennon, B.; Rui Li; Senhu Li; Meinzer, H.-P.; Nemeth, G.; Raicu, D.S.; Rau, A.-M.; van Rikxoort, E.M.; Rousson, M.; Rusko, L.;

Saddi, K.A.; Schmidt, G.; Seghers, D.; Shimizu, A.; Slagmolen, P.; Sorantin, E.; Soza, G.; Susomboon, R.; Waite, J.M.; Wimmer, A.; Wolf, I.; , "Comparison and Evaluation of Methods for Liver Segmentation From CT Datasets," Medical Imaging, IEEE Transactions on , vol.28, no.8, pp.1251-1265, Aug. 2009 doi: 10.1109/TMI.2009.2013851

```
The mask image as scipy array.
```

Definition at line 32 of file Volume.py.

3.13.2 Constructor & Destructor Documentation

3.13.2.1 def medpy.metric.Volume.Volume.__init__ (*self*, *mask*, *reference*, *mask_offset* = `[0`, *reference_offset* = `[0`)

Initialize the class with two binary images whose 0 values are considered to be background voxels and the 1 values foreground.

Parameters

mask,:	mask as an scipy array (3D image)
reference,:	reference as an scipy array (3D image)
mask_-offset,:	offset of the mask array to 0,0,0-origin
mask_-offset,:	offset of the reference array to 0,0,0-origin

Definition at line 62 of file Volume.py.

3.13.3 Member Function Documentation

3.13.3.1 def medpy.metric.Volume.Volume._ComputeIntersectionWindow (*self*) `[private]`

Computes and sets the intersection windows of the two images.

The intersection window marks the area in which the two images bounding boxes intersect. The returned rectangle is defined by its lower-left and upper-right corner.

Definition at line 173 of file Volume.py.

3.13.3.2 def medpy.metric.Volume.Volume.GetRelativeVolumeDifference (*self*)

Returns the relative volume difference between two shapes defined by the non-background voxels of in the binary images.

The relative volume distance of 0 means that the volumes are identical. Note that this doesn't apply that the shapes are identical or overlap at all. The result is given as signed number [-100,+100] to indicate over- or under-segmentation. Warning: Can produce values greater than -/+100 if the difference in volume exceeds 50%.

Warning: This measure is not actually a metric, as it is not symmetric.

Metric definition: The relative volume difference between two sets of voxels $S(A)$ and $S(b)$ is given in percent and defined as:

$$100 * \frac{|A - B|}{|B|}$$

Definition at line 92 of file Volume.py.

3.13.3.3 def medpy.metric.Volume.Volume.GetVolumetricOverlapError (*self*)

Returns the volumetric overlap error between two shapes defined by the non-background voxels of in two binary images.

The volumetric overlap error is 0 for a perfect match and 100 if the two shapes don't overlap at all.

Metric definition: The volumetric overlap error between two sets of voxels $S(A)$ and $S(b)$ is given in percent and defined as:

$$100 * \left(1 - \frac{|A \cap B|}{|A \cup B|}\right)$$

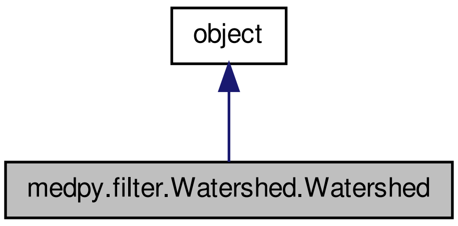
Definition at line 117 of file Volume.py.

The documentation for this class was generated from the following file:

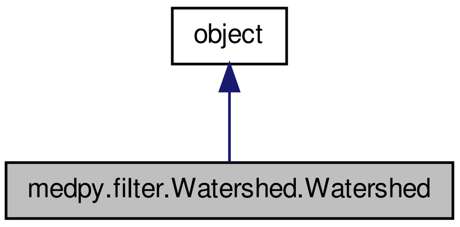
- /home/loli/Programming/Python/MedPy/src/medpy/metric/Volume.py

3.14 medpy.filter.Watershed.Watershed Class Reference

Inheritance diagram for medpy.filter.Watershed.Watershed:



Collaboration diagram for medpy.filter.Watershed.Watershed:



3.14.1 Detailed Description

Definition at line 22 of file Watershed.py.

The documentation for this class was generated from the following file:

- /home/loli/Programming/Python/MedPy/src/medpy/filter/Watershed.py

Index