dicom2nifti Documentation

icometrix

Oct 17, 2018

Contents

Contents:

CHAPTER 1

dicom2nifti

Python library for converting dicom files to nifti

Author Arne Brys

Organization icometrix

Repository https://github.com/icometrix/dicom2nifti

API documentation http://dicom2nifti.readthedocs.io/en/latest

CHAPTER 2

Using dicom2nifti

2.1 Installation

```
pip install dicom2nifti
```

2.2 Updating

```
pip install dicom2nifti --upgrade
```

2.3 Usage

2.3.1 Command line

```
dicom2nifti [-h] [-G] [-r] [-o RESAMPLE_ORDER] [-p RESAMPLE_PADDING] [-M] [-C] [-R]␣
↪input_directory output_directory
```

for more information

```
dicom2nifti -h
```

2.3.2 From python

Converting a directory with dicom files to nifti files

```
import dicom2nifti

dicom2nifti.convert_directory(dicom_directory, output_folder)
```

Converting a directory with only 1 series to 1 nifti file

```
import dicom2nifti

dicom2nifti.dicom_series_to_nifti(original_dicom_directory, output_file, reorient_
↪nifti=True)
```

2.4 Supported data

Most anatomical data for CT and MR should be supported as long as they are in classical dicom files.

Try avoiding "Implicit VR Endian" if possible as this makes converting non anatomical (i.e. DTI, fMRI, ...) much more difficult.

There is some vendor specific support, more specifically for 4D imaging like fMRI and DTI/DKI

2.4.1 Gantry tilted CT

By default support for gantry tilted ct is disabled as we validate image orthogonality. You can explicitly allow gantry tilted data by disabling this validation.

Standard this will result in a nifti file where the gantry tilt is captured by the affine matrix. We also provide the option to resample the data to an orthogonal nifti. For this resampling we use scipy.ndimage.interpolation.affine_transform. You should configure the padding value and spline interpolation order

Command line:

```
dicom2nifti -G -r -o 1 -p -1000 input_directory output_directory
```

Python code:

```
import dicom2nifti
import dicom2nifti.settings as settings

settings.disable_validate_orthogonal()
settings.enable_resampling()
settings.set_resample_spline_interpolation_order(1)
settings.set_resample_padding(-1000)

dicom2nifti.convert_directory(dicom_directory, output_folder)
```

2.4.2 GE MR

Anatomical data should all be support. 4D images like fMRI and DTI/DKI are supported.

2.4.3 Siemens MR

Anatomical data should all be support. 4D images like fMRI and DTI/DKI are supported.

2.4.4 Philips MR

For classic dicom files dicom2nifti support anatomical. For classic dicom files 4D images like fMRI and DTI/DKI are supported.

For "Philips Enhanced Dicom" there is no support for "Implicit VR Endian" transfer syntax. For the others we support anatomical and 4D images like fMRI and DTI/DKI.

2.4.5 Hitachi MR

Anatomical data should all be support. 4D images like fMRI and DTI/DKI are NOT supported. Anyone willing to share DTI and/or fMRI dicom form Hitachi scanners please contact us.

2.5 Unsupported data

If you encounter unsupported data you can help the development of dicom2nifti by providing a dataset. This dataset should be anonymised (but leave as much of the private fields as possible).

CHAPTER 3

dicom2nifti package

3.1 Submodules

3.1.1 dicom2nifti.common module

dicom2nifti

@author: abrys

dicom2nifti.common.**apply_scaling**(*data*, *dicom_headers*)

Rescale the data based on the RescaleSlope and RescaleOffset Based on the scaling from pydicomseries

Parameters

- **dicom_headers** – dicom headers to use to retreive the scaling factors
- **data** – the input data

dicom2nifti.common.**create_affine**(*sorted_dicoms*)

Function to generate the affine matrix for a dicom series This method was based on (http://nipy.org/nibabel/dicom/dicom_orientation.html)

Parameters **sorted_dicoms** – list with sorted dicom files

dicom2nifti.common.**do_scaling**(*data*, *rescale_slope*, *rescale_intercept*, *private_scale_slope=1.0*, *private_scale_intercept=0.0*)

dicom2nifti.common.**get_fd_array_value**(*tag*, *count*)

Getters for data that also work with implicit transfersyntax

Parameters

- **count** – number of items in the array
- **tag** – the tag to read

dicom2nifti.common.**get_fd_value**(*tag*)

Getters for data that also work with implicit transfersyntax

Parameters **tag** – the tag to read

dicom2nifti.common.**get_fl_value**(*tag*)
Getters for data that also work with implicit transfersyntax

Parameters **tag** – the tag to read

dicom2nifti.common.**get_is_value**(*tag*)
Getters for data that also work with implicit transfersyntax

Parameters **tag** – the tag to read

dicom2nifti.common.**get_numpy_type**(*dicom_header*)
Make NumPy format code, e.g. "uint16", "int32" etc from two pieces of info: mosaic.PixelRepresentation – 0 for unsigned, 1 for signed; mosaic.BitsAllocated – 8, 16, or 32

Parameters **dicom_header** – the read dicom file/headers

Returns numpy format string

dicom2nifti.common.**get_ss_value**(*tag*)
Getters for data that also work with implicit transfersyntax

Parameters **tag** – the tag to read

dicom2nifti.common.**get_volume_pixeldata**(*sorted_slices*)
the slice and intercept calculation can cause the slices to have different dtypes we should get the correct dtype that can cover all of them

Parameters **sorted_slices** (*list of slices*) – sliced sored in the correct order to create volume

dicom2nifti.common.**is_ge**(*dicom_input*)
Use this function to detect if a dicom series is a GE dataset

Parameters **dicom_input** – list with dicom objects

dicom2nifti.common.**is_hitachi**(*dicom_input*)
Use this function to detect if a dicom series is a hitachi dataset

Parameters **dicom_input** – directory with dicom files for 1 scan of a dicom_header

dicom2nifti.common.**is_multiframe_dicom**(*dicom_input*)
Use this function to detect if a dicom series is a siemens 4D dataset NOTE: Only the first slice will be checked so you can only provide an already sorted dicom directory (containing one series)

Parameters **dicom_input** – directory with dicom files for 1 scan

dicom2nifti.common.**is_orthogonal**(*dicoms*, *log_details=False*)
Validate that volume is orthonormal

Parameters **dicoms** – check that we have a volume without skewing

dicom2nifti.common.**is_philips**(*dicom_input*)
Use this function to detect if a dicom series is a philips dataset

Parameters **dicom_input** – directory with dicom files for 1 scan of a dicom_header

dicom2nifti.common.**is_siemens**(*dicom_input*)
Use this function to detect if a dicom series is a siemens dataset

Parameters **dicom_input** – directory with dicom files for 1 scan

dicom2nifti.common.**is_valid_imaging_dicom**(*dicom_header*)
Function will do some basic checks to see if this is a valid imaging dicom

dicom2nifti.common.**read_dicom_directory**(*dicom_directory*, *stop_before_pixels=False*)
Read all dicom files in a given directory (stop before pixels)

Parameters

- **stop_before_pixels** (*bool*) – Should we stop reading before the pixeldata (handy if we only want header info)
- **dicom_directory** (*six.string_types*) – Directory with dicom data

Returns List of dicom objects

dicom2nifti.common.**set_fd_value**(*tag*, *value*)
Setters for data that also work with implicit transfersyntax

Parameters

- **value** – the value to set on the tag
- **tag** – the tag to read

dicom2nifti.common.**set_ss_value**(*tag*, *value*)
Setter for data that also work with implicit transfersyntax

Parameters

- **value** – the value to set on the tag
- **tag** – the tag to read

dicom2nifti.common.**set_tr_te**(*nifti_image*, *repetition_time*, *echo_time*)
Set the tr and te in the nifti headers

Parameters

- **echo_time** – echo time
- **repetition_time** – repetition time
- **nifti_image** – nifti image to set the info to

dicom2nifti.common.**sort_dicoms**(*dicoms*)
Sort the dicoms based om the image possition patient

Parameters **dicoms** – list of dicoms

dicom2nifti.common.**validate_orientation**(*dicoms*)
Validate that all dicoms have the same orientation

Parameters **dicoms** – list of dicoms

dicom2nifti.common.**validate_orthogonal**(*dicoms*)
Validate that volume is orthonormal

Parameters **dicoms** – check that we have a volume without skewing

dicom2nifti.common.**validate_slicecount**(*dicoms*)
Validate that volume is big enough to create a meaningfull volume This will also skip localizers and alike

Parameters **dicoms** – list of dicoms

dicom2nifti.common.**validate_sliceincrement**(*dicoms*)
Validate that the distance between all slices is equal (of very close to)

Parameters **dicoms** – list of dicoms

dicom2nifti.common.**write_bval_file**(*bvals*, *bval_file*)
Write an array of bvals to a bval file

Parameters

- **bvals** – array with the values
- **bval_file** – filepath to write to

dicom2nifti.common.**write_bvec_file**(*bvecs*, *bvec_file*)
Write an array of bvecs to a bvec file

Parameters

- **bvecs** – array with the vectors
- **bvec_file** – filepath to write to

3.1.2 dicom2nifti.convert_dicom module

dicom2nifti

@author: abrys

class dicom2nifti.convert_dicom.**Vendor**
Bases: object

Enum with the vendor

GE = 2

GENERIC = 0

HITACHI = 4

PHILIPS = 3

SIEMENS = 1

dicom2nifti.convert_dicom.**are_imaging_dicoms**(*dicom_input*)
This function will check the dicom headers to see which type of series it is Possibilities are fMRI, DTI, Anatomical (if no clear type is found anatomical is used)

Parameters **dicom_input** – directory with dicom files or a list of dicom objects

dicom2nifti.convert_dicom.**dicom_array_to_nifti**(*dicom_list*, *output_file*, *reorient_nifti=True*)
Converts dicom single series (see pydicom) to nifty, mimicking SPM

Examples: See unit test

will return a dictionary containing - the NIFTI under key 'NIFTI' - the NIFTI file path under 'NII_FILE' - the BVAL file path under 'BVAL_FILE' (only for dti) - the BVEC file path under 'BVEC_FILE' (only for dti)

IMPORTANT: If no specific sequence type can be found it will default to anatomical and try to convert. You should check that the data you are trying to convert is supported by this code

Inspired by http://nipy.sourceforge.net/nibabel/dicom/spm_dicom.html Inspired by http://code.google.com/p/pydicom/source/browse/source/dicom/contrib/pydicom_series.py

Parameters

- **reorient_nifti** – if True the nifti affine and data will be updated so the data is stored LAS oriented
- **output_file** – file path to write to
- **dicom_list** – list with uncompressed dicom objects as read by pydicom

dicom2nifti.convert_dicom.**dicom_series_to_nifti**(*original_dicom_directory*, *output_file=None*, *reorient_nifti=True*)

Converts dicom single series (see pydicom) to nifty, mimicking SPM

Examples: See unit test

will return a dictionary containing - the NIFTI under key 'NIFTI' - the NIFTI file path under 'NII_FILE' - the BVAL file path under 'BVAL_FILE' (only for dti) - the BVEC file path under 'BVEC_FILE' (only for dti)

IMPORTANT: If no specific sequence type can be found it will default to anatomical and try to convert. You should check that the data you are trying to convert is supported by this code

Inspired by http://nipy.sourceforge.net/nibabel/dicom/spm_dicom.html Inspired by http://code.google.com/p/pydicom/source/browse/source/dicom/contrib/pydicom_series.py

Parameters

- **reorient_nifti** – if True the nifti affine and data will be updated so the data is stored LAS oriented
- **output_file** – file path to write to if not set to None
- **original_dicom_directory** – directory with the dicom files for a single series/scan

:return nibabel image

3.1.3 dicom2nifti.convert_dir module

this module houses all the code to just convert a directory of random dicom files

@author: abrys

dicom2nifti.convert_dir.**convert_directory**(*dicom_directory*, *output_folder*, *compression=True*, *reorient=True*)

This function will order all dicom files by series and order them one by one

Parameters

- **compression** – enable or disable gzip compression
- **reorient** – reorient the dicoms according to LAS orientation
- **output_folder** – folder to write the nifti files to
- **dicom_directory** – directory with dicom files

3.1.4 dicom2nifti.convert_ge module

dicom2nifti

@author: abrys

dicom2nifti.convert_ge.**dicom_to_nifti**(*dicom_input*, *output_file=None*)

This is the main dicom to nifti conversion fuction for ge images. As input ge images are required. It will then determine the type of images and do the correct conversion

Examples: See unit test

Parameters

- **output_file** – the filepath to the output nifti file
- **dicom_input** – list with dicom objects

3.1.5 dicom2nifti.convert_generic module

dicom2nifti

@author: abrys

dicom2nifti.convert_generic.**dicom_to_nifti**(*dicom_input*, *output_file*)
This function will convert an anatomical dicom series to a nifti

Examples: See unit test

Parameters

- **output_file** – filepath to the output nifti
- **dicom_input** – directory with the dicom files for a single scan, or list of read in dicoms

3.1.6 dicom2nifti.convert_philips module

dicom2nifti

@author: abrys

dicom2nifti.convert_philips.**dicom_to_nifti**(*dicom_input*, *output_file=None*)
This is the main dicom to nifti conversion fuction for philips images. As input philips images are required. It will then determine the type of images and do the correct conversion

Examples: See unit test

Parameters

- **output_file** – file path to the output nifti
- **dicom_input** – directory with dicom files for 1 scan

3.1.7 dicom2nifti.convert_siemens module

dicom2nifti

@author: abrys

class dicom2nifti.convert_siemens.**MosaicType**
Bases: object

Enum for the possible types of mosaic data

ASCENDING = 1

DESCENDING = 2

dicom2nifti.convert_siemens.**dicom_to_nifti**(*dicom_input*, *output_file=None*)
This is the main dicom to nifti conversion function for ge images. As input ge images are required. It will then determine the type of images and do the correct conversion

Parameters

- **output_file** – filepath to the output nifti
- **dicom_input** – directory with dicom files for 1 scan

3.1.8 dicom2nifti.exceptions module

dicom2nifti

@author: abrys

exception dicom2nifti.exceptions.**ConversionError**(*message*)
Bases: Exception

Custom error type to distinguish between know validations and script errors

exception dicom2nifti.exceptions.**ConversionValidationError**(*message*)
Bases: Exception

Custom error type to distinguish between know validations and script errors

3.1.9 dicom2nifti.image_reorientation module

Created on Thu Aug 1 16:10:56 2013

@author: vterzopoulos, abrys

dicom2nifti.image_reorientation.**reorient_image**(*input_image*, *output_image*)
Change the orientation of the Image data in order to be in LAS space x will represent the coronal plane, y the sagittal and z the axial plane. x increases from Right (R) to Left (L), y from Posterior (P) to Anterior (A) and z from Inferior (I) to Superior (S)

Returns The output image in nibabel form

Parameters

- **output_image** – filepath to the nibabel image
- **input_image** – filepath to the nibabel image

3.1.10 dicom2nifti.image_volume module

Created on Fri Jun 7 07:40:20 2013

@author: abrys

class dicom2nifti.image_volume.**ImageVolume**(*nifti_image*)
Bases: object

Class representing an imagevolume. You can provide it with a nifti and can be used to get slices in a certain direction It will take the affine matrix into account to find the correct orientation

get_slice(*slice_type*, *slice_number*, *time_point=0*)
Returns a slice of the dataset. slice.data contains the window/levelled values, in uint8 slice.original_data contains the original data for this slice :param time_point: in case of 4d nifti the 4th dimension :param slice_number: the slice number :param slice_type: tye slice type (AXIAL, SAGITTAL, CORONAL)

class dicom2nifti.image_volume.**Slice**
Bases: object

Class containing all data for a single slice in an image volume

original_data = None

slice_orientation = None

class dicom2nifti.image_volume.**SliceOrientation**
Bases: object

Class containing the orientation of a slice

normal_component = None

x_component = None

x_inverted = False

y_component = None

y_inverted = False

class dicom2nifti.image_volume.**SliceType**
Bases: object

ENUM like container for the slice types

AXIAL = 1

CORONAL = 3

SAGITTAL = 2

dicom2nifti.image_volume.**load**(*nifti_file*)

3.1.11 dicom2nifti.settings module

dicom2nifti.settings.**disable_pydicom_read_force**()
Enable the pydicom read force to try to read non conform dicom data

dicom2nifti.settings.**disable_resampling**()
Disable resampling in case of gantry tilted data (disabled by default)

dicom2nifti.settings.**disable_validate_multiframe_implicit**()
Disable the validation that checks that data is not multiframe implicit This allows to sometimes convert Philips Multiframe with implicit transfer syntax

dicom2nifti.settings.**disable_validate_orientation**()
Disable the validation of the slice orientation. This validation checks that all slices have the same orientation (are parallel). USE WITH CAUTION!

dicom2nifti.settings.**disable_validate_orthogonal**()
Disable the validation whether the volume is orthogonal (so without gantry tilting or alike). This allows for converting gantry tilted data. The gantry tilting will be reflected in the affine matrix and not in the data USE WITH CAUTION!

dicom2nifti.settings.**disable_validate_slicecount**()
Disable the validation of the minimal slice count of 4 slices. This allows for converting data with less slices. Usually less than 4 could be considered localizer or similar thus ignoring these scans by default USE WITH CAUTION!

dicom2nifti.settings.**disable_validate_sliceincrement**()
Disable the validation of the slice increment. This allows for converting data where the slice increment is not consistent. USE WITH CAUTION!

dicom2nifti.settings.**enable_pydicom_read_force**()
Enable the pydicom read force to try to read non conform dicom data

dicom2nifti.settings.**enable_resampling**()
Enable resampling in case of gantry tilted data (disabled by default)

dicom2nifti.settings.**enable_validate_multiframe_implicit**()
Enable the validation that checks that data is not multiframe implicit again (DEFAULT ENABLED)

dicom2nifti.settings.**enable_validate_orientation**()
Enable the slice orientation validation again (DEFAULT ENABLED)

dicom2nifti.settings.**enable_validate_orthogonal**()
Enable the validation whether the volume is orthogonal again (DEFAULT ENABLED)

dicom2nifti.settings.**enable_validate_slicecount**()
Enable the validation of the minimal slice count of 4 slices again (DEFAULT ENABLED)

dicom2nifti.settings.**enable_validate_sliceincrement**()
Enable the slice increment validation again (DEFAULT ENABLED)

dicom2nifti.settings.**set_gdcmconv_path**(*path*)
Set where the filepath to the gdcmconv executable (needed is it is not found in your PATH)

Parameters **path** – the file path to the gdcmconv executable

dicom2nifti.settings.**set_resample_padding**(*padding*)
Set the spline interpolation padding

dicom2nifti.settings.**set_resample_spline_interpolation_order**(*order*)
Set the spline interpolation order used during resampling of gantry tilted data

3.2 Module contents

dicom2nifti

@author: abrys

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

d

Index

H

I

L

M

N

O

P

R

S

V

W

X

Y