

digiholo2D

Generated by Doxygen 1.8.6

Thu Oct 30 2014 15:04:36

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	abstract_fringe_analyser Class Reference	5
3.1.1	Detailed Description	5
3.1.2	Member Function Documentation	5
3.1.2.1	calc_wrapped_phase_map	5
3.2	abstract_smart_tile_unwrapper Class Reference	6
3.2.1	Detailed Description	6
3.2.2	Constructor & Destructor Documentation	6
3.2.2.1	abstract_smart_tile_unwrapper	6
3.2.3	Member Function Documentation	6
3.2.3.1	unwrap	6
3.3	abstract_smart_unwrapper Class Reference	6
3.3.1	Detailed Description	6
3.3.2	Member Function Documentation	7
3.3.2.1	unwrap	7
3.4	abstract_tile_merger Class Reference	8
3.4.1	Detailed Description	8
3.4.2	Member Function Documentation	8
3.4.2.1	merge_tiles	8
3.5	abstract_tile_unwrapper Class Reference	8
3.5.1	Detailed Description	9
3.5.2	Constructor & Destructor Documentation	9
3.5.2.1	abstract_tile_unwrapper	9
3.5.3	Member Function Documentation	9
3.5.3.1	unwrap	9
3.6	abstract_unwrapper Class Reference	9

3.6.1	Detailed Description	10
3.6.2	Member Function Documentation	10
3.6.2.1	unwrap	10
3.7	col_major_float_image Class Reference	10
3.7.1	Constructor & Destructor Documentation	10
3.7.1.1	col_major_float_image	10
3.7.2	Member Function Documentation	11
3.7.2.1	operator()	11
3.8	Ui::digiholoMainGui Class Reference	11
3.9	digiholoMainGui Class Reference	11
3.10	EDGE Struct Reference	12
3.11	float_image Class Reference	12
3.11.1	Detailed Description	12
3.11.2	Constructor & Destructor Documentation	13
3.11.2.1	float_image	13
3.11.2.2	float_image	14
3.11.3	Member Function Documentation	14
3.11.3.1	clear_mem	14
3.11.3.2	copy_data_to	14
3.11.3.3	get_data_pointer	14
3.11.3.4	get_height	14
3.11.3.5	get_pixel	14
3.11.3.6	get_width	15
3.11.3.7	operator()	15
3.11.3.8	set_pixel	15
3.11.3.9	zero_fill	15
3.12	grad_fit_tile_unwrapper Class Reference	15
3.12.1	Detailed Description	16
3.12.2	Member Function Documentation	16
3.12.2.1	unwrap	16
3.13	minimization_tile_unwrapper Class Reference	16
3.13.1	Detailed Description	16
3.13.2	Constructor & Destructor Documentation	17
3.13.2.1	minimization_tile_unwrapper	17
3.13.2.2	~minimization_tile_unwrapper	18
3.13.3	Member Function Documentation	18
3.13.3.1	unwrap	18
3.14	PIXEL Struct Reference	18
3.15	qt_meta_stringdata_digiholoMainGui_t Struct Reference	18
3.16	qt_meta_stringdata_ReconstructionThread_t Struct Reference	19

3.17 ReconstructionThread Class Reference	19
3.17.1 Member Function Documentation	19
3.17.1.1 init	19
3.17.1.2 progressInfo	20
3.17.1.3 progressUpdate	20
3.17.1.4 request_termination	20
3.18 row_major_float_image Class Reference	20
3.18.1 Constructor & Destructor Documentation	21
3.18.1.1 row_major_float_image	21
3.18.2 Member Function Documentation	21
3.18.2.1 operator()	21
3.19 simple1d_tile_merger Class Reference	21
3.19.1 Constructor & Destructor Documentation	21
3.19.1.1 simple1d_tile_merger	21
3.19.2 Member Function Documentation	22
3.19.2.1 merge_tiles	22
3.20 smart_tile Class Reference	22
3.20.1 Constructor & Destructor Documentation	23
3.20.1.1 smart_tile	23
3.20.1.2 smart_tile	23
3.20.1.3 smart_tile	23
3.20.2 Member Function Documentation	23
3.20.2.1 get_tilegroup	23
3.20.2.2 has_tilegroup	23
3.20.2.3 operator=	24
3.20.2.4 rewrap	24
3.20.3 Friends And Related Function Documentation	24
3.20.3.1 add_tile_to_group	24
3.20.3.2 merge_tilegroups	24
3.21 smart_tile_junction Class Reference	25
3.21.1 Detailed Description	25
3.21.2 Member Enumeration Documentation	25
3.21.2.1 rel_position	25
3.21.3 Constructor & Destructor Documentation	25
3.21.3.1 smart_tile_junction	25
3.21.4 Member Function Documentation	25
3.21.4.1 get_relative_position	25
3.22 smart_tiled_image Class Reference	26
3.22.1 Detailed Description	26
3.22.2 Constructor & Destructor Documentation	26

3.22.2.1	smart_tiled_image	26
3.22.2.2	smart_tiled_image	26
3.22.2.3	~smart_tiled_image	27
3.22.3	Member Function Documentation	27
3.22.3.1	convert_to_float_image	27
3.22.3.2	unwrap_tiles	27
3.23	smart_tilegroup Class Reference	27
3.23.1	Detailed Description	28
3.23.2	Member Function Documentation	28
3.23.2.1	add_value	28
3.23.2.2	create_new	28
3.23.2.3	size	28
3.23.3	Friends And Related Function Documentation	28
3.23.3.1	add_tile_to_group	28
3.23.3.2	merge_tilegroups	29
3.24	srncp_tile_merger Class Reference	29
3.24.1	Detailed Description	29
3.24.2	Member Function Documentation	29
3.24.2.1	calc_junction_reliability	29
3.24.2.2	merge_tiles	30
3.25	srncp_unwrapper Class Reference	30
3.25.1	Detailed Description	30
3.25.2	Member Function Documentation	30
3.25.2.1	unwrap	30
3.26	Strand_tile_unwrapper Class Reference	31
3.26.1	Detailed Description	31
3.26.2	Member Function Documentation	31
3.26.2.1	unwrap	31
3.27	Takeda_FFTW_fringe_analyser Class Reference	31
3.27.1	Detailed Description	32
3.27.2	Constructor & Destructor Documentation	32
3.27.2.1	Takeda_FFTW_fringe_analyser	32
3.27.3	Member Function Documentation	32
3.27.3.1	calc_wrapped_phase_map	32
3.28	tesselated_image Class Reference	33
3.28.1	Constructor & Destructor Documentation	33
3.28.1.1	tesselated_image	33
3.28.1.2	tesselated_image	33
3.28.1.3	~tesselated_image	33
3.28.2	Member Function Documentation	33

3.28.2.1	get_image_height	33
3.28.2.2	get_image_width	34
3.28.2.3	get_tile_count_height	34
3.28.2.4	get_tile_count_width	34
3.28.2.5	operator=	34
3.28.2.6	unwrap_tiles	34
3.29	tile Class Reference	34
3.29.1	Detailed Description	35
3.29.2	Constructor & Destructor Documentation	35
3.29.2.1	tile	35
3.29.2.2	~tile	35
3.29.3	Member Function Documentation	35
3.29.3.1	add_value	35
3.29.3.2	calc_mean	35
3.29.3.3	clear_mem	35
3.29.3.4	copy_data	36
3.29.3.5	generate_from_image	36
3.29.3.6	get_height	36
3.29.3.7	get_tilegroup	36
3.29.3.8	get_width	36
3.29.3.9	has_group	36
3.29.3.10	multiply	36
3.29.3.11	rewrap	37
3.29.3.12	wrap	37
3.30	tile_image Class Reference	37
3.30.1	Detailed Description	37
3.31	tile_junction Class Reference	37
3.31.1	Detailed Description	38
3.31.2	Member Enumeration Documentation	38
3.31.2.1	rel_position	38
3.31.3	Constructor & Destructor Documentation	38
3.31.3.1	tile_junction	38
3.31.4	Member Function Documentation	38
3.31.4.1	get_relative_position	38
3.32	tile_merge_unwrapper Class Reference	38
3.32.1	Detailed Description	39
3.32.2	Constructor & Destructor Documentation	39
3.32.2.1	tile_merge_unwrapper	39
3.32.3	Member Function Documentation	39
3.32.3.1	unwrap	39

3.33 tilegroup Class Reference	40
3.33.1 Detailed Description	40
3.33.2 Constructor & Destructor Documentation	40
3.33.2.1 ~tilegroup	40
3.33.3 Member Function Documentation	40
3.33.3.1 add_tile	40
3.33.3.2 add_value	40
3.33.3.3 create_new	41
3.33.3.4 size	41
3.33.4 Friends And Related Function Documentation	41
3.33.4.1 merge_tilegroups	41
3.34 Ui_digiholoMainGui Class Reference	41
Index	43

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

abstract_fringe_analyser	5
Takeda_FFTW_fringe_analyser	31
abstract_smart_tile_unwrapper	6
abstract_smart_unwrapper	6
abstract_tile_merger	8
simple1d_tile_merger	21
srncp_tile_merger	29
abstract_tile_unwrapper	8
grad_fit_tile_unwrapper	15
minimization_tile_unwrapper	16
Strand_tile_unwrapper	31
abstract_unwrapper	9
srncp_unwrapper	30
tile_merge_unwrapper	38
EDGE	12
float_image	12
col_major_float_image	10
row_major_float_image	20
PIXEL	18
QMainWindow	
digiholoMainGui	11
qt_meta_stringdata_digiholoMainGui_t	18
qt_meta_stringdata_ReconstructionThread_t	19
QThread	
ReconstructionThread	19
smart_tile	22
smart_tile_junction	25
smart_tiled_image	26
smart_tilegroup	27
tesselated_image	33
tile	34
tile_image	37
tile_junction	37
tilegroup	40
Ui_digiholoMainGui	41
Ui::digiholoMainGui	11

Chapter 2

Class Index

2.1 Class List

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

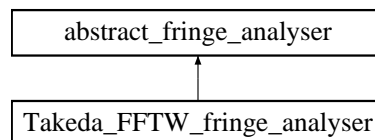
abstract_fringe_analyser	5
abstract_smart_tile_unwrapper	6
abstract_smart_unwrapper	6
abstract_tile_merger	8
abstract_tile_unwrapper	8
abstract_unwrapper	9
col_major_float_image	10
Ui::digiholoMainGui	11
digiholoMainGui	11
EDGE	12
float_image	12
grad_fit_tile_unwrapper	15
minimization_tile_unwrapper	16
PIXEL	18
qt_meta_stringdata_digiholoMainGui_t	18
qt_meta_stringdata_ReconstructionThread_t	19
ReconstructionThread	19
row_major_float_image	20
simple1d_tile_merger	21
smart_tile	22
smart_tile_junction	25
smart_tiled_image	26
smart_tilegroup	27
srncp_tile_merger	29
srncp_unwrapper	30
Strand_tile_unwrapper	31
Takeda_FFTW_fringe_analyser	31
tesselated_image	33
tile	34
tile_image	37
tile_junction	37
tile_merge_unwrapper	38
tilegroup	40
Ui_digiholoMainGui	41

Chapter 3

Class Documentation

3.1 abstract_fringe_analyser Class Reference

Inheritance diagram for abstract_fringe_analyser:



Public Member Functions

- virtual bool [calc_wrapped_phase_map](#) (float_image *fringe_pattern, float_image *wrapped_phase_map)=0

3.1.1 Detailed Description

abstrahiert die Funktionalität aus einem Interferenzbild die (gewrappte!) Phase zu berechnen. Abgeleiteten Klassen sollen alle nötigen Parameter im Konstruktor übergeben werden.

3.1.2 Member Function Documentation

3.1.2.1 virtual bool abstract_fringe_analyser::calc_wrapped_phase_map (float_image * *fringe_pattern*, float_image * *wrapped_phase_map*) [pure virtual]

Berechnet eine wrapped Phase map aus einem Fringe Image aus float Daten.

Parameters

<i>fringe_pattern</i>	Das Interferenzbild.
<i>wrapped_phase-map</i>	Pointer mit Platz für die berechnete wrapped phase map.

Returns

true if everything went well, false if not

Implemented in [Takeda_FFTW_fringe_analyser](#).

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

- `include/abstract_fringe_analyser.h`

3.2 abstract_smart_tile_unwrapper Class Reference

```
#include <abstract_smart_tile_unwrapper.h>
```

Public Member Functions

- [abstract_smart_tile_unwrapper](#) ()
- virtual void [unwrap](#) (boost::shared_ptr< [smart_tile](#) > t)=0

3.2.1 Detailed Description

This pure virtual class provides an abstract interface for an unwrapper that unwraps a single tile. It operates with boost smart pointers.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 `abstract_smart_tile_unwrapper::abstract_smart_tile_unwrapper ()` `[inline]`

Konstruktor

3.2.3 Member Function Documentation

3.2.3.1 `virtual void abstract_smart_tile_unwrapper::unwrap (boost::shared_ptr< smart_tile > t)` `[pure virtual]`

Unwrap a given tile.

Parameters

<i>t</i>	
----------	--

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

- `include/block_srncp/abstract_smart_tile_unwrapper.h`

3.3 abstract_smart_unwrapper Class Reference

```
#include <abstract_smart_unwrapper.h>
```

Public Member Functions

- virtual boost::shared_ptr
 < [float_image](#) > [unwrap](#) (boost::shared_ptr< [float_image](#) > wrapped_phase_image)=0

3.3.1 Detailed Description

This pure virtual class provides an interface for a general unwrapper that operates with boost smart pointers.

3.3.2 Member Function Documentation

3.3.2.1 `virtual boost::shared_ptr<float_image> abstract_smart_unwrapper::unwrap (boost::shared_ptr< float_image > wrapped_phase_image) [pure virtual]`

An abstract method for phase unwrapping.

Parameters

<i>wrapped_phase- _image</i>	This image contains the wrapped phase data.
----------------------------------	---

Returns

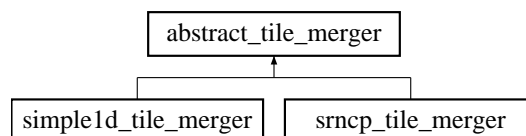
A [float_image](#) with the unwrapped phase data. The image will actually be a [row_major_float_image](#).

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

- include/abstract_smart_unwrapper.h

3.4 abstract_tile_merger Class Reference

Inheritance diagram for abstract_tile_merger:

**Public Member Functions**

- virtual void [merge_tiles](#) ([tesselated_image](#) *t)=0

3.4.1 Detailed Description

provides the abstract interface for merging unwrapped tiles to a single unwrapped phase image.

3.4.2 Member Function Documentation

3.4.2.1 virtual void [abstract_tile_merger::merge_tiles](#) ([tesselated_image](#) * t) [pure virtual]

This method merges tiles from a tessellated wrapped phase image to an unwrapped image. These tiles already need to be unwrapped individually and are then unwrapped with respect to each other to form an unwrapped phase map.

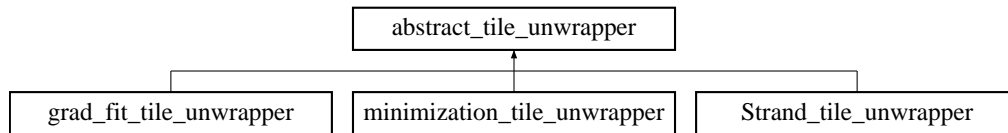
Implemented in [srncp_tile_merger](#), and [simple1d_tile_merger](#).

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

- include/block_srncp/abstract_tile_merger.h

3.5 abstract_tile_unwrapper Class Reference

Inheritance diagram for abstract_tile_unwrapper:



Public Member Functions

- [abstract_tile_unwrapper](#) ()
- virtual void [unwrap](#) ([tile](#) *t)=0

3.5.1 Detailed Description

Funktionalität für das unwrappen von tiles. Ableitungen dieser Klassen müssen die Methode [unwrap\(tile*\)](#) implementieren.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 [abstract_tile_unwrapper::abstract_tile_unwrapper](#) () `[inline]`

Konstruktor

3.5.3 Member Function Documentation

3.5.3.1 `virtual void abstract_tile_unwrapper::unwrap (tile * t) [pure virtual]`

Diese Methode unwrappet eine einzelne tile.

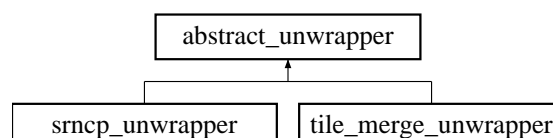
Implemented in [minimization_tile_unwrapper](#), [Strand_tile_unwrapper](#), and [grad_fit_tile_unwrapper](#).

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

- include/block_srncp/abstract_tile_unwrapper.h

3.6 abstract_unwrapper Class Reference

Inheritance diagram for abstract_unwrapper:



Public Member Functions

- virtual bool [unwrap](#) ([float_image](#) *wrapped_phase_image, [float_image](#) *unwrapped_phase_image)=0

3.6.1 Detailed Description

eine abstrakte Klasse, die eine unwrap Methode kapselt. Über den Konstruktor werden alle benötigten Parameter an eine Instanz übergeben. Die wesentliche Methode ist die unwrap Methode.

3.6.2 Member Function Documentation

3.6.2.1 `virtual bool abstract_unwrapper::unwrap (float_image * wrapped_phase_image, float_image * unwrapped_phase_image) [pure virtual]`

Abstrakte Methode für den Phase Unwrap

Parameters

<i>wrapped_phase_image</i>	Das Bild mit der Wrapped Phase Verteilung.
<i>unwrapped_phase_image</i>	Zeiger auf ein Bild mit Platz für die Unwrapped Phase Distribution. Muss gleiche größe wie wrapped phase haben!!

Returns

true, wenn keine Fehler aufgetreten sind, sonst false.

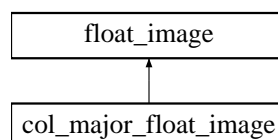
Implemented in [tile_merge_unwrapper](#), and [srncp_unwrapper](#).

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

- include/abstract_unwrapper.h

3.7 col_major_float_image Class Reference

Inheritance diagram for col_major_float_image:



Public Member Functions

- **col_major_float_image** (float *data, long width, long height)
- [col_major_float_image](#) (long width, long height)
- virtual float & [operator\(\)](#) (long w, long h)
- virtual float **operator()** (long w, long h) const

Additional Inherited Members

3.7.1 Constructor & Destructor Documentation

3.7.1.1 `col_major_float_image::col_major_float_image (long width, long height) [inline]`

Generate image that reserves memory

Parameters

<i>width</i>	
<i>height</i>	

3.7.2 Member Function Documentation

3.7.2.1 virtual float& col_major_float_image::operator()(long *w*, long *h*) [inline],[virtual]

Return element at position width = *w*, height = *h*, starting with (0,0) in upper left corner of the image. Implemented in child classes, see e.g. [row_major_float_image](#).

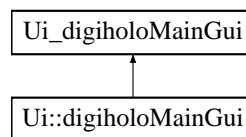
Implements [float_image](#).

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

- include/col_major_image.h

3.8 Ui::digiholoMainGui Class Reference

Inheritance diagram for Ui::digiholoMainGui:



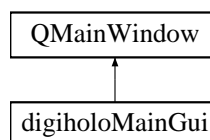
Additional Inherited Members

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

- Qt/digiholo2DGUI/ui_digiholoMainGui.h

3.9 digiholoMainGui Class Reference

Inheritance diagram for digiholoMainGui:



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

- Qt/digiholo2DGUI/include/digiholoMainGui.h
- Qt/digiholo2DGUI/src/digiholoMainGui.cpp

3.10 EDGE Struct Reference

Public Attributes

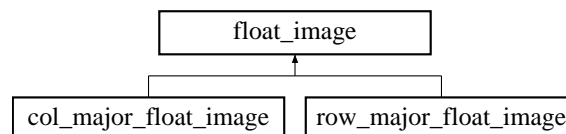
- float **reliab**
- [PIXEL](#) * **pointer_1**
- [PIXEL](#) * **pointer_2**
- int **increment**

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

- include/srncp/srncp_unwrap.h

3.11 float_image Class Reference

Inheritance diagram for float_image:



Public Member Functions

- [float_image](#) (long width, long height)
- [float_image](#) (float *data, long width, long height)
- virtual long [get_width](#) () const
- virtual long [get_height](#) () const
- virtual float * [get_data_pointer](#) ()
- virtual float & [operator\(\)](#) (long w, long h)=0
- virtual float **operator()** (long w, long h) const =0
- virtual void [zero_fill](#) ()
- virtual bool [copy_data_to](#) (float_image *img)
- virtual void [clear_mem](#) ()
- virtual float [get_pixel](#) (long iw, long ih)
- virtual void [set_pixel](#) (long iw, long ih, float val)

Protected Attributes

- float * **data**
- long **width**
- long **height**

3.11.1 Detailed Description

speichert ein 2D Bild, welches in einem 1D Datenvektor gespeichert wird. Die Ordnung des Datenvektors gibt die Klasse nicht vor. Es müssen auch nicht alle Daten hintereinander weg liegen. Die Zugriffsoperatoren sollen überladen werden, um eine bestimmte Art der Speicherung (z.b. row-major / column-major) oder sonstwas zu abstrahieren. Wichtig: Beim Aufruf des Destruktors des Bildes wird das entsprechende Array nicht freigegeben!

3.11.2 Constructor & Destructor Documentation

3.11.2.1 float_image::float_image (long *width*, long *height*)

Create float image with and allocate data array! The data will not be zero filled.

Parameters

<i>width</i>	
<i>height</i>	

3.11.2.2 float_image::float_image (float * *data*, long *width*, long *height*)

Create float image with specified data array. The image will operate on this array.

Parameters

<i>data</i>	
<i>width</i>	
<i>height</i>	

3.11.3 Member Function Documentation

3.11.3.1 void float_image::clear_mem () [virtual]

Free memory associated with the image. This is not done when the destructor is called.

3.11.3.2 bool float_image::copy_data_to (float_image * *img*) [virtual]

Copies the data from this image into the given image. Both images need to have the same dimensions.

Parameters

<i>img</i>	Data from this image is copied into img.
------------	--

Returns

true if successful, false if not.

3.11.3.3 float * float_image::get_data_pointer () [virtual]

Returns

pointer to data array

3.11.3.4 long float_image::get_height () const [virtual]

Returns

image height

3.11.3.5 float float_image::get_pixel (long *iw*, long *ih*) [virtual]

Get Pixel value at specified position.

Parameters

<i>iw</i>	
<i>ih</i>	

Returns

3.11.3.6 `long float_image::get_width () const` [virtual]

Returns

image width

3.11.3.7 `virtual float& float_image::operator()(long w, long h)` [pure virtual]

Return element at position width = w, height = h, starting with (0,0) in upper left corner of the image. Implemented in child classes, see e.g. [row_major_float_image](#).

Implemented in [col_major_float_image](#), and [row_major_float_image](#).

3.11.3.8 `void float_image::set_pixel (long iw, long ih, float val)` [virtual]

Set Pixel value at specified position.

Parameters

<i>iw</i>	
<i>ih</i>	
<i>val</i>	

Returns

3.11.3.9 `void float_image::zero_fill ()` [virtual]

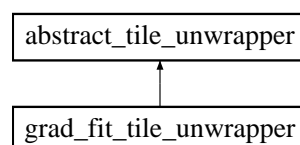
Fill image with zeroes.

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

- include/float_image.h
- src/float_image.cpp

3.12 grad_fit_tile_unwrapper Class Reference

Inheritance diagram for grad_fit_tile_unwrapper:



Public Member Functions

- virtual void [unwrap](#) ([tile](#) *t)

3.12.1 Detailed Description

unwrapper basiert auf der Idee eine ungefähre Fitfunktion für die ungewrappte Phase aus dem Gradienten zu berechnen (siehe Laborbuch 3, Seite 103) und dann mit einem Korrekturterm die Phase zu unwrappen. Welcher Fit genau verwendet wird, siehe unwrap Methode...

3.12.2 Member Function Documentation

3.12.2.1 void grad_fit_tile_unwrapper::unwrap (tile * t) [virtual]

Implementation einer linearen Fitfunktion für Phi_fit innerhalb einer tile $\text{Phi_fit}(x,y) = f(x,y) = a \cdot x + b \cdot y + c$ (dabei $x=iw$, $y=ih$)

Parameters

t	tile
-----	------

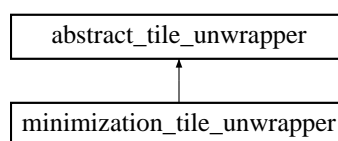
Implements [abstract_tile_unwrapper](#).

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

- include/block_srncp/grad_fit_tile_unwrapper.h
- src/block_srncp/grad_fit_tile_unwrapper.cpp

3.13 minimization_tile_unwrapper Class Reference

Inheritance diagram for minimization_tile_unwrapper:



Public Member Functions

- [minimization_tile_unwrapper](#) (int max_inter, float tol=0.0314)
- virtual [~minimization_tile_unwrapper](#) ()
- virtual void [unwrap](#) ([tile](#) *t)

3.13.1 Detailed Description

!! Diese Klasse soll mit GSL eine effizienteren Tile-Unwrap als der [Strand_tile_unwrapper](#) durchführen. Es ist allerdings nicht so leicht, das Minimierungsproblem so zu verpacken, dass es mit Minimierungsroutinen gelöst werden kann. Man muss erstmal das Minimum eingrenzen... wie soll ich das möglichst effizient machen?

3.13.2 Constructor & Destructor Documentation

3.13.2.1 minimization_tile_unwrapper::minimization_tile_unwrapper (int *max_iter*, float *tol* = 0.0314)

Constructor

Parameters

<i>max_iter</i>	Max number of iterations for minimization
<i>tol</i>	Tolerance parameter as stopping criterion for minimization

Allocate a Brent type minimizer;

3.13.2.2 `minimization_tile_unwrapper::~~minimization_tile_unwrapper ()` [virtual]

Destruktor

3.13.3 Member Function Documentation

3.13.3.1 `void minimization_tile_unwrapper::unwrap (tile * t)` [virtual]

Diese Methode unwrappt eine einzelne tile. HIER TWEAK MÖGLICH FÜR RELATIVES STOPPING KRITERIUM. Ich habe es auf 0.0 gesetzt. Siehe auch GSL Reference Manual.

Implements [abstract_tile_unwrapper](#).

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

- include/block_srncp/minimization_tile_unwrapper.h
- src/block_srncp/minimization_tile_unwrapper.cpp

3.14 PIXEL Struct Reference

Public Attributes

- int **increment**
- int **number_of_pixels_in_group**
- float **value**
- float **reliability**
- int **group**
- int **new_group**
- struct [PIXEL](#) * **head**
- struct [PIXEL](#) * **last**
- struct [PIXEL](#) * **next**

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

- include/srncp/srncp_unwrap.h

3.15 qt_meta_stringdata_digiholoMainGui_t Struct Reference

Public Attributes

- QByteArrayData **data** [9]
- char **stringdata** [134]

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

- Qt/digiholo2DGUI/moc_digiholoMainGui.cpp

3.16 qt_meta_stringdata_ReconstructionThread_t Struct Reference

Public Attributes

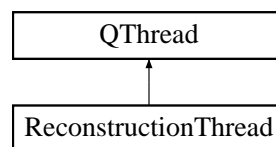
- QByteArrayData **data** [6]
- char **stringdata** [68]

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

- Qt/digiholo2DGUI/moc_ReconstructionThread.cpp

3.17 ReconstructionThread Class Reference

Inheritance diagram for ReconstructionThread:



Signals

- void [progressUpdate](#) (int percent)
- void [progressInfo](#) (QString infotext)

Public Member Functions

- void [init](#) (QString fringe_dir_path, QStringList filename_filters, int width, int height, [abstract_fringe_analyser](#) *fringe_analyser, [abstract_unwrapper](#) *unwrapper, bool output_wrapped, bool output_unwrapped, bool output16bit)
- void [request_termination](#) ()

Protected Member Functions

- virtual void **run** ()

3.17.1 Member Function Documentation

3.17.1.1 void ReconstructionThread::init (QString *fringe_dir_path*, QStringList *filename_filters*, int *width*, int *height*, [abstract_fringe_analyser](#) * *fringe_analyser*, [abstract_unwrapper](#) * *unwrapper*, bool *output_wrapped*, bool *output_unwrapped*, bool *output16bit*)

This method has to be called before starting the reconstruction thread.

Parameters

<i>fringe_dir_path</i>	This is the base dir that contains the fringe image files.
------------------------	--

<i>filename_filters</i>	A list of supported filename filters e.g. "*.raw", ".dat" etc...
<i>width</i>	Width of the images in the folder. Must be same for all images.
<i>height</i>	Height of the images in the folder. Must be same for all images.
<i>fringe_analyser</i>	Pointer to the fringe analyser that generates wrapped phase images from the interferograms. May not be NULL-pointer.
<i>unwrapper</i>	Unwraps the wrapped phase images. This pointer may be NULL. If so, no unwrap is performed.
<i>output_wrapped</i>	Whether a wrapped phase images should be saved to disk (will be in separate folder)
<i>output_unwrapped</i>	Whether unwrapped phase images should be saved to disk (will be in separate folder)
<i>output16bit</i>	Whether data should be converted to 16bit int. (see write_image16bit for important specifics)

3.17.1.2 void ReconstructionThread::progressInfo (QString *infotext*) [signal]

Dies ist eine einfache Methode um während der Laufzeit text aus diesem Thread zurückzugeben...

Parameters

<i>infotext</i>	
-----------------	--

3.17.1.3 void ReconstructionThread::progressUpdate (int *percent*) [signal]

Signal, für den Fortschritt des Threads.

Parameters

<i>percent</i>	
----------------	--

3.17.1.4 void ReconstructionThread::request_termination ()

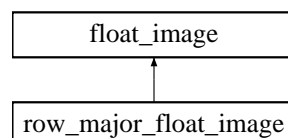
This should be called when Thread execution is to be stopped before it finishes. It basically copies the request-Interruption() functionality from Qt 5.3. which does not exist in QT 5.1.

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

- Qt/digiholo2DGUI/include/ReconstructionThread.h
- Qt/digiholo2DGUI/moc_ReconstructionThread.cpp
- Qt/digiholo2DGUI/src/ReconstructionThread.cpp

3.18 row_major_float_image Class Reference

Inheritance diagram for row_major_float_image:



Public Member Functions

- **row_major_float_image** (float *data, long width, long height)

- [row_major_float_image](#) (long width, long height)
- virtual float & [operator\(\)](#) (long w, long h)
- virtual float **operator()** (long w, long h) const

Additional Inherited Members

3.18.1 Constructor & Destructor Documentation

3.18.1.1 `row_major_float_image::row_major_float_image (long width, long height)` `[inline]`

Generate image that reserves memory

Parameters

<i>width</i>	
<i>height</i>	

3.18.2 Member Function Documentation

3.18.2.1 `float & row_major_float_image::operator() (long w, long h)` `[virtual]`

Return element at position width = w, height = h, starting with (0,0) in upper left corner of the image. Implemented in child classes, see e.g. [row_major_float_image](#).

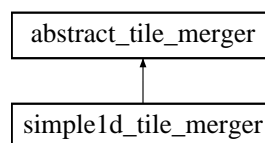
Implements [float_image](#).

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

- include/row_major_float_image.h
- src/row_major_float_image.cpp

3.19 simple1d_tile_merger Class Reference

Inheritance diagram for simple1d_tile_merger:



Public Member Functions

- [simple1d_tile_merger](#) (float unwrap_threshold)
- virtual void [merge_tiles](#) ([tesselated_image](#) *t)

3.19.1 Constructor & Destructor Documentation

3.19.1.1 `simple1d_tile_merger::simple1d_tile_merger (float unwrap_threshold)`

Linear unwrapper based on 1d unwrapping of blocks.

Parameters

<i>unwrap_-threshold</i>	If this threshold is exceeded an unwrap between two blocks is performed. This parameter should be in [0,2PI].
--------------------------	---

3.19.2 Member Function Documentation

3.19.2.1 void simple1d_tile_merger::merge_tiles (tesslated_image * t) [virtual]

Merge tiles of tesslated image. The tiles have to be unwrapped inside themselves. This method unwraps the tiles with respect to each other. The phase discontinuity between each tile must be integer multiples of 2PI.

Parameters

<i>t</i>	
----------	--

Implements [abstract_tile_merger](#).

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

- include/block_srncp/simple1d_tile_merger.h
- src/block_srncp/simple1d_tile_merger.cpp

3.20 smart_tile Class Reference

Public Member Functions

- [smart_tile](#) ()
- [~smart_tile](#) ()
Destructor.
- [smart_tile](#) (const [smart_tile](#) &other)
- [smart_tile](#) & [operator=](#) (const [smart_tile](#) &other)
ACHTUNG HIER WEITERE VARIABLEN KOPIEREN, WENN MEHR DAZUKOMMT.
- [smart_tile](#) (float [image](#) *img, long grid_width, long grid_height, long iw, long ih)
- float [get_value_at](#) (long iw, long ih)
Get value of pixel in tile.
- void [set_value_at](#) (long iw, long ih, float val)
Set value of the specified pixel to val.
- long [get_height](#) () const
Get height number of elements in tile in h-direction.
- long [get_width](#) () const
Get number of elements in tile in w-direction.
- [smart_tile](#) & [add_value](#) (float val)
*Adds a specified value to each pixel in the tile. Returns *this.*
- [smart_tile](#) & [multiply_value](#) (float val)
*Multiplies a specified value to each pixel in the tile. Returns *this.*
- [smart_tile](#) & [wrap](#) ()
Wraps all values in this tile to interval [-Pi, Pi]. Returns pointer to this.*
- [smart_tile](#) & [rewrap](#) (float val)
- bool [has_tilegroup](#) ()
- boost::shared_ptr
< [smart_tilegroup](#) > [get_tilegroup](#) ()

Friends

- void [add_tile_to_group](#) (boost::shared_ptr< [smart_tile](#) > t, boost::shared_ptr< [smart_tilegroup](#) > g)
- void [merge_tilegroups](#) (boost::shared_ptr< [smart_tilegroup](#) > g1, boost::shared_ptr< [smart_tilegroup](#) > g2)

3.20.1 Constructor & Destructor Documentation

3.20.1.1 smart_tile::smart_tile ()

This constructor performs nothing and reserves no memory for data.

3.20.1.2 smart_tile::smart_tile (const smart_tile & other)

Macht eine Kopie der gegebenen tile, die auf denselben Speicher zeigt. Dazu wird der eigene smartpointer umgesetzt und zeigt nun nicht mehr auf den Speicher auf den er zuvor zeigte.

Parameters

<i>other</i>	Die gegebene tile.
--------------	--------------------

Returns

3.20.1.3 smart_tile::smart_tile (float_image * img, long grid_width, long grid_height, long iw, long ih)

Dieser Konstruktor macht erstellt eine neue Tile, die eine lokale Kopie der Daten auf den entsprechenden Bereich aus einem [float_image](#) erstellt. Dabei wird das Bild in ein Gitter mit grid_width und grid_height zerteilt und die Tiles entsprechend indiziert (w:links->rechts, h:oben nach unten, wobei linke obere Ecke iw,ih=0,0 entspricht). Falls das Gridding genau aufgeht sind alle Tiles gleich groß nämlich grid_width*grid_height. Falls nicht, dann werden die untersten Tiles entsprechend kleiner gemacht. Vergleiche Methode

Parameters

<i>img</i>	Das Bild, welches in Tiles zerlegt werden soll
<i>grid_width</i>	Die (maximale) Anzahl der Elemente in dem Tile.
<i>grid_height</i>	
<i>iw</i>	Index der Tile in w-Richtung
<i>ih</i>	

3.20.2 Member Function Documentation

3.20.2.1 boost::shared_ptr< smart_tilegroup > smart_tile::get_tilegroup ()

Returns

Pointer to the tilegroup. If no tilegroup is present a NULL pointer is returned. That may lead to bad things... so check first that the tile has a group.

3.20.2.2 bool smart_tile::has_tilegroup ()

Returns

true if this tile is member in a tilegroup, false otherwise.

3.20.2.3 `smart_tile` & `smart_tile::operator= (const smart_tile & other)`

ACHTUNG HIER WEITERE VARIABLEN KOPIEREN, WENN MEHR DAZUKOMMT.

Siehe Kopierkonstruktor.

Parameters

<i>other</i>	
--------------	--

Returns

Reference to `*this`.

3.20.2.4 `smart_tile` & `smart_tile::rewrap (float val)`

For each pixel `T` sets pixel to `T = wrap(T+val)-val`.

Parameters

<i>val</i>	value to add.
------------	---------------

Returns

pointer to this

3.20.3 Friends And Related Function Documentation

3.20.3.1 `void add_tile_to_group (boost::shared_ptr< smart_tile > t, boost::shared_ptr< smart_tilegroup > g)` [friend]

Sets the tilegroup of the tile `t` to the given group and adds this tile to the vector of tiles in the tilegroup `g`. ACHTUNG! This method should only be used if this tile is not already member in a tilegroup. If it is member of a tilegroup, then the reference to this tile in the old group will still be there. This can lead to errors. For merging tilegroups use the `merge_into` method from [smart_tilegroup](#). These conditions are not checked.

Parameters

<i>t</i>	the tile. It must not be in a group.
<i>g</i>	the tilegroup. It must not already contain the tile <code>t</code> .

3.20.3.2 `void merge_tilegroups (boost::shared_ptr< smart_tilegroup > g1, boost::shared_ptr< smart_tilegroup > g2)` [friend]

Merges [smart_tilegroup](#) `g1` into group `g2`. All `smart_tiles` from `g1` will now point to `g2` as their group and `g1` will be devoid of elements.

Parameters

<i>g1</i>	
<i>g2</i>	

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

- `include/block_srncp/smart_tile.h`
- `src/block_srncp/smart_tile.cpp`

3.21 smart_tile_junction Class Reference

```
#include <smart_tile_junction.h>
```

Public Types

- enum [rel_position](#) { **UP** =123, **DOWN** =-123, **RIGHT** = 321, **LEFT** = -321 }

Public Member Functions

- [smart_tile_junction](#) ()
Pointer to the tiles connected.
- [smart_tile_junction](#) (boost::weak_ptr< [smart_tile](#) > first, boost::weak_ptr< [smart_tile](#) > second, [rel_position](#) second_to_first)
- boost::weak_ptr< [smart_tile](#) > [get_first](#) ()
- boost::weak_ptr< [smart_tile](#) > [get_second](#) ()
- [rel_position](#) [get_relative_position](#) ()

3.21.1 Detailed Description

A little helper that encapsules a connection between two tiles. It stores the two tiles connected as well as the relative position of the second tile to the first. It also provides methods for calculating the differences, variance of the junction.

3.21.2 Member Enumeration Documentation

3.21.2.1 enum smart_tile_junction::rel_position

This enum provides a way of describing the relative position of two tiles. The values are chosen pretty much arbitrary but such that -UP = DOWN and -RIGHT = LEFT. Meaning: right/left = +/- 1 in w direction and down/up = +/- 1 in h direction

The tile junction does not own the tiles, that means it has only weak pointers to the tiles.

3.21.3 Constructor & Destructor Documentation

3.21.3.1 [smart_tile_junction::smart_tile_junction](#) (boost::weak_ptr< [smart_tile](#) > *first*, boost::weak_ptr< [smart_tile](#) > *second*, [rel_position](#) *second_to_first*)

Constructor

Parameters

<i>first</i>	First tile
<i>second</i>	Second tile
<i>second_to_first</i>	Position of the second tile relative to first tile

3.21.4 Member Function Documentation

3.21.4.1 [smart_tile_junction::rel_position](#) [smart_tile_junction::get_relative_position](#) ()

Gives the relative position of second tile to first tile.

Returns

the relative position

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

- include/block_srncp/smart_tile_junction.h
- src/block_srncp/smart_tile_junction.cpp

3.22 smart_tiled_image Class Reference

Public Member Functions

- [smart_tiled_image](#) ()
- [smart_tiled_image](#) ([float_image](#) *img, long tilecount_width_hint, long tilecount_height_hint)
- virtual [~smart_tiled_image](#) ()
- long [get_tilecount_height](#) () const
- long [get_tilecount_width](#) () const
- long [get_pixel_width](#) () const
Returns width of the image in pixels.
- long [get_pixel_height](#) () const
Returns height of the image in pixels.
- boost::weak_ptr< [smart_tile](#) > [get_tile_at](#) (long iw, long ih)
Returns the tile at the specified index.
- boost::shared_ptr
< [row_major_float_image](#) > [convert_to_float_image](#) ()
- void [unwrap_tiles](#) (boost::shared_ptr< [abstract_smart_tile_unwrapper](#) > uw)

3.22.1 Detailed Description

ersetzt tessellated image.

Sie verwaltet ein 2D Array von tiles, die zusammengesetzt ein Bild ergeben. Dabei erfolgt die Verwaltung des 2D Arrays intern über eine Indizierung eines 1D Vektors aus strong pointers auf die tiles. Damit kann die dynamische Speicherverwaltung von boost mit dynamischen Arrays kombiniert werden.

Ein [smart_tiled_image](#) wird mit den Daten aus einem [float_image](#) initialisiert, außerdem werdend die Anzahl der tiles in w und h Richtung übergeben in die das Bild zerlegt werden soll. Weil jede [smart_tile](#) sich eine lokale Kopie der betreffenden Daten erstellt ist das [smart_tiled_image](#) in diesem Sinne auch eine Kopie des gegebenen [float_image](#). Die Operationen auf dem [smart_tiled_image](#) verändern das Originalbild also nicht.

3.22.2 Constructor & Destructor Documentation

3.22.2.1 smart_tiled_image::smart_tiled_image ()

Standard constructor creates an empty tessellated image with tilecount_width und tilecount_height equals zero.

3.22.2.2 smart_tiled_image::smart_tiled_image (float_image * img, long tilecount_width_hint, long tilecount_height_hint)

Create a [smart_tiled_image](#) from an existing [float_image](#). The tiled image create a copy and thus not operate on the same memory as the original [float_image](#).

Parameters

<i>img</i>	The image to be split into tiles
<i>tilecount_width_hint</i>	Number if tiles in width direction. This number is not exactly adhered to, but the real number of tiles will be close.
<i>tilecount_height_hint</i>	Number of tiles in height direction. This number is not exactly adhered to, but the real number of tiles will be close.

3.22.2.3 smart_tiled_image::~smart_tiled_image () [virtual]

Frees the space associated with this image, if there are no more strong pointers to the tiles within this image. Tiles will be destroyed as soon as no strong pointers to them exist anymore.

3.22.3 Member Function Documentation

3.22.3.1 boost::shared_ptr< row_major_float_image > smart_tiled_image::convert_to_float_image ()

Generates a float image that contains a copy of the [smart_tiled_image](#).

Returns

A shared pointer to that image. This shared pointer has a custom deleter with delete_float_image from [float_image.h](#) so that it can deal with the memory-management without further need to address it in the code.

Shared pointer mit custom deleter übergeben, damit das Memory Management übernommen werden kann.

3.22.3.2 void smart_tiled_image::unwrap_tiles (boost::shared_ptr< abstract_smart_tile_unwrapper > uw)

This method will apply the unwrap method of the tile unwrapper instance to every tile of the image.

Parameters

<i>uw</i>	An instance of the unwrapper.
-----------	-------------------------------

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

- include/block_srncp/smart_tiled_image.h
- src/block_srncp/smart_tiled_image.cpp

3.23 smart_tilegroup Class Reference

Public Member Functions

- [~smart_tilegroup](#) ()
Destruktor.
- int [size](#) ()
- void [add_value](#) (float val)

Static Public Member Functions

- static boost::shared_ptr
 < [smart_tilegroup](#) > [create_new](#) ()

Friends

- void [add_tile_to_group](#) (boost::shared_ptr< [smart_tile](#) > t, boost::shared_ptr< [smart_tilegroup](#) > g)
- void [merge_tilegroups](#) (boost::shared_ptr< [smart_tilegroup](#) > g1, boost::shared_ptr< [smart_tilegroup](#) > g2)

3.23.1 Detailed Description

, die eine Gruppe von Smart-Tiles verwaltet. Die tiles haben strong pointer auf ihre tilegroup, wohingegen diese tilegroup nur schwache pointer auf die tiles haben. Die starken pointer auf die tiles sitzen im [smart_tiled_image](#). Die [smart_tilegroup](#) Objekte müssen demnach nirgendwo zwischengespeichert werden. Sie können über die entsprechenden smart_tiles angesprochen werden. Wenn keine [smart_tile](#) mehr auf die tilegroup zeigt, wird diese automatisch zerstört.

Die Funktion zum hinzufügen von smart_tiles zu einer Gruppe befindet sich in der [smart_tile](#) Klasse.

3.23.2 Member Function Documentation

3.23.2.1 void smart_tilegroup::add_value (float val)

Add value to all tiles in the group.

Parameters

<i>val</i>	The value to add.
------------	-------------------

3.23.2.2 static boost::shared_ptr<smart_tilegroup> smart_tilegroup::create_new () [inline],[static]

Static creator method, so that new tilegroups can only be constructed like this

Returns

A strong pointer to a new tilegroup.

3.23.2.3 int smart_tilegroup::size ()

Returns

The number of elements in the tilegroup.

3.23.3 Friends And Related Function Documentation

3.23.3.1 void add_tile_to_group (boost::shared_ptr< smart_tile > t, boost::shared_ptr< smart_tilegroup > g) [friend]

Sets the tilegroup of the tile t to the given group and adds this tile to the vector of tiles in the tilegroup g. ACHTUNG! This method should only be used if this tile is not already member in a tilegroup. If it is member of a tilegroup, then the reference to this tile in the old group will still be there. This can lead to errors. For merging tilegroups use the merge_into method from [smart_tilegroup](#). These conditions are not checked.

Parameters

<i>t</i>	the tile. It must not be in a group.
----------	--------------------------------------

<i>g</i>	the tilegroup. It must not already contain the tile <i>t</i> .
----------	--

3.23.3.2 void merge_tilegroups (boost::shared_ptr< smart_tilegroup > *g1*, boost::shared_ptr< smart_tilegroup > *g2*) [friend]

Merges [smart_tilegroup](#) *g1* into group *g2*. All smart_tiles from *g1* will now point to *g2* as their group and *g1* will be devoid of elements.

Parameters

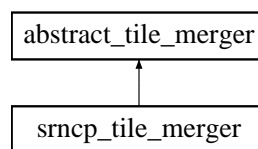
<i>g1</i>	
<i>g2</i>	

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

- include/block_srncp/smart_tile.h
- src/block_srncp/smart_tilegroup.cpp

3.24 srncp_tile_merger Class Reference

Inheritance diagram for srncp_tile_merger:



Public Member Functions

- virtual void [merge_tiles](#) (tesselated_image *t)
- float [calc_junction_reliability](#) (tile_junction &tj)

Static Public Attributes

- static const float [MAX_RELIABILITY](#) = 1e10
Maximalwert für die zulässige Reliability einer [tile_junction](#).
- static const float [MIN_RELIABILITY](#) = -1.f
Dies ist der Minimalwert für die Reliability einer [tile_junction](#).

3.24.1 Detailed Description

merges individually unwrapped tiles on the SRNCP algorithm from the Harraez et al Paper. The reliability measure for tiles is very different from the measure for single pixels though.

3.24.2 Member Function Documentation

3.24.2.1 float srncp_tile_merger::calc_junction_reliability (tile_junction & tj)

Calculate the reliability of a given tile junction

3.24.2.2 void srncp_tile_merger::merge_tiles (tesslated_image * t) [virtual]

This method merges tiles from a tesslated wrapped phase image to an unwrapped image. These tiles already need to be unwrapped individually and are then unwrapped with respect to each other to form an unwrapped phase map.

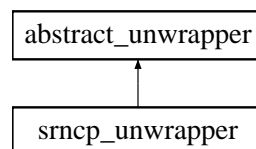
Implements [abstract_tile_merger](#).

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

- include/block_srncp/srncp_tile_merger.h
- src/block_srncp/srncp_tile_merger.cpp

3.25 srncp_unwrapper Class Reference

Inheritance diagram for srncp_unwrapper:



Public Member Functions

- virtual bool [unwrap](#) (float_image *wrapped, float_image *unwrapped)

3.25.1 Detailed Description

provides the functionality of the srncp unwrapper using the abstract unwrapper interface.

Author

g.antonopoulos

3.25.2 Member Function Documentation

3.25.2.1 bool srncp_unwrapper::unwrap (float_image * wrapped_phase_image, float_image * unwrapped_phase_image) [virtual]

Abstrakte Methode für den Phase Unwrap

Parameters

<i>wrapped_phase_image</i>	Das Bild mit der Wrapped Phase Verteilung.
<i>unwrapped_phase_image</i>	Zeiger auf ein Bild mit Platz für die Unwrapped Phase Distribution. Muss gleiche gröÙe wie wrapped phase haben!!

Returns

true, wenn keine Fehler aufgetreten sind, sonst false.

Implements [abstract_unwrapper](#).

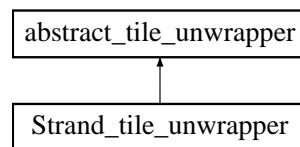
The documentation for this class was generated from the following files:

- include/srncp/srncp_unwrap.h
- src/srncp/srncp_unwrap.cpp

3.26 Strand_tile_unwrapper Class Reference

```
#include <Strand_tile_unwrapper.h>
```

Inheritance diagram for Strand_tile_unwrapper:



Public Member Functions

- **Strand_tile_unwrapper** (int N_rho)
- virtual void **unwrap** (tile *t)

3.26.1 Detailed Description

Unwrap single tile with the single block method from Strand et al.: "Two-Dimensional Phase Unwrapping Using a Block Least-Squares Method", equations (4),(5) insbesondere

3.26.2 Member Function Documentation

3.26.2.1 void Strand_tile_unwrapper::unwrap (tile * t) [virtual]

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	5
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	---

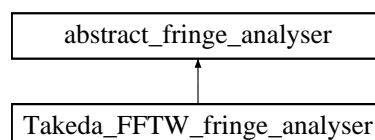
Implements `abstract_tile_unwrapper`.

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

- include/block_srncp/Strand_tile_unwrapper.h
- src/block_srncp/Strand_tile_unwrapper.cpp

3.27 Takeda_FFTW_fringe_analyser Class Reference

Inheritance diagram for Takeda_FFTW_fringe_analyser:



Public Member Functions

- `Takeda_FFTW_fringe_analyser` (int width, int height, unsigned FLAGS=FFTW_ESTIMATE)
- virtual bool `calc_wrapped_phase_map(float image *fringe img, float image *wrapped phase map)`

Static Public Attributes

- static const float [CLIPPING_FRACTION](#) = 0.8f

Gibt an, wieviel Prozent des Abstand vom der Nullfrequenz als clipping-radius verwendet werden.

3.27.1 Detailed Description

Wrapped Phase Map aus einem Interferenzbild mit einer beliebig orientierten Streifenmodulation nach Takeda et. al "Fourier-transform method of fringe-pattern analysis for computer-based topography and interferometry". Dabei wird die FFTW verwendet, um die Fourier-Transformation durchzuführen. Die Bilder, welche übergeben werden, MÜSSEN mit `fftw_malloc` allociert worden sein.

3.27.2 Constructor & Destructor Documentation

3.27.2.1 `Takeda_FFTW_fringe_analyser::Takeda_FFTW_fringe_analyser (int width, int height, unsigned FLAGS = FFTW_ESTIMATE)`

In dem Konstruktor wird ein Plan für ein Array mit width und height erstellt. Damit können dann alle weiteren Pläne schnell erstellt werden ausgeführt werden. Breite und Höhe der Bilder, die gerechnet werden sollen muss angegeben werden. Dies wird dazu benutzt, einmal einen Plan zu erstellen. Falls die Bilder in der `calc_wrapped_phase_map` Methode nicht dieselbe Größe haben, wird eine exception geschmissen.

Parameters

<i>width</i>	Breite
<i>height</i>	Höhe
<i>flags</i>	

Arrays für FFTW initialisieren

3.27.3 Member Function Documentation

3.27.3.1 `bool Takeda_FFTW_fringe_analyser::calc_wrapped_phase_map (float_image * fringe_img, float_image * wrapped_phase_map) [virtual]`

Berechnung der Wrapped Phase Map aus einem Interferenzbild mit einer beliebig orientierten Streifenmodulation nach Takeda et. al "Fourier-transform method of fringe-pattern analysis for computer-based topography and interferometry".

Parameters

<i>fringe_pattern</i>	Das Interferenzbild.
<i>wrapped_phase_map</i>	Pointer mit Platz für die berechnete wrapped phase map.

Returns

true, wenn keine Fehler aufgetreten sind, sonst false

Implements [abstract_fringe_analyser](#).

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

- include/Takeda_FFTW_fringe_analyser.h
- Takeda_FFTW_fringe_analyser.cpp

3.28 tessellated_image Class Reference

Public Member Functions

- [tessellated_image](#) ()
- [tessellated_image](#) ([float_image](#) *img, long tilecount_width, long tilecount_height)
- [tessellated_image](#) & [operator=](#) (const [tessellated_image](#) &t)
- virtual [~tessellated_image](#) ()
- [tile](#) * [get_tile_ptr](#) (long w, long h)
- long [get_tile_count_width](#) () const
- long [get_tile_count_height](#) () const
- long [get_image_width](#) () const
- long [get_image_height](#) () const
- void [unwrap_tiles](#) ([abstract_tile_unwrapper](#) *u)

Public Attributes

- [tile](#) ** [image_tiles](#)

3.28.1 Constructor & Destructor Documentation

3.28.1.1 tessellated_image::tessellated_image ()

Standard constructor creates an empty tessellated images with all pointers set to NULL and tilecount_width and tilecount_height of 0.

3.28.1.2 tessellated_image::tessellated_image ([float_image](#) * *img*, long *tilecount_width*, long *tilecount_height*)

Create a tiled image from an existing image. The tessellated image operates on the same memory as the original image.

Parameters

<i>img</i>	The image to be tessellated into tiles
<i>tilecount_width</i>	Number if tiles in width direction. This number is not exactly adhered to, but the real number of tiles will be close.
<i>tilecount_height</i>	Number of tiles in height direction. This number is not exactly adhered to, but the real number of tiles will be close.

3.28.1.3 tessellated_image::~tessellated_image () [virtual]

Frees the memory associated with this image's tiles but does destroy the [float_image](#) that this image operates on.

Free the memory occupied by this object but not the underlying image.

3.28.2 Member Function Documentation

3.28.2.1 long tessellated_image::get_image_height () const

Returns

Image height in pixels

3.28.2.2 `long tesslated_image::get_image_width () const`

Returns

Image width in pixels

3.28.2.3 `long tesslated_image::get_tile_count_height () const`

Returns

Number of tiles in height direction.

3.28.2.4 `long tesslated_image::get_tile_count_width () const`

Returns

Number of tiles in width-direction.

3.28.2.5 `tesslated_image & tesslated_image::operator= (const tesslated_image & t)`

Makes the L-value point to the same [float_image](#) as the R-value but generates a new set of tiles for this image that are independent of the R-value's tiles.

Parameters

<i>t</i>	
----------	--

Returns

3.28.2.6 `void tesslated_image::unwrap_tiles (abstract_tile_unwrapper * u)`

Unwraps all tiles using the unwrap method given by the instance of the [abstract_tile_unwrapper](#)

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

- `include/block_srncp/tesslated_image.h`
- `src/block_srncp/tesslated_image.cpp`

3.29 tile Class Reference

Public Member Functions

- void [clear_mem](#) ()
- [tile](#) (long max_width, long max_height)
- [~tile](#) ()
- float & [operator\(\)](#) (long w, long h)
Access element of this tile at (w,h)
- float [operator\(\)](#) (long w, long h) const
Access value of the element of this tile at (w,h)
- [tile](#) & [add_value](#) (float value)
- [tile](#) & [multiply](#) (float value)

- `tile & wrap ()`
Only works if pixelvalue is in $[-PI,PI]$ and val is in $[-2PI,2PI]$.
- `tile & rewrap (float val)`
- `float * copy_data () const`
- `long get_width () const`
- `long get_height () const`
- `void generate_from_image (float_image *img, long tile_max_width, long tile_max_height, long iw, long ih)`
- `float calc_mean () const`
- `bool has_group ()`
- `tilegroup * get_tilegroup ()`

Friends

- class `tilegroup`

3.29.1 Detailed Description

Unterteilung eines Bildes in Kacheln (tiles). Die Daten der Tiles sind nicht hintereinander im Speicher abgelegt, darum kann sie auch nicht von `float_image` abgeleitet sein.

3.29.2 Constructor & Destructor Documentation

3.29.2.1 `tile::tile (long max_width, long max_height)`

Create a tile that reserves memory for pointers to (max_width)X(max_height) elements

3.29.2.2 `tile::~tile ()`

Does not do anything. Call free to free mem.

3.29.3 Member Function Documentation

3.29.3.1 `tile & tile::add_value (float value)`

Add a specified number to all pixels in the tile.

Parameters

<i>value</i>	The value to add.
--------------	-------------------

3.29.3.2 `float tile::calc_mean () const`

Calculate mean value of pixel values in tile.

Returns

mean.

3.29.3.3 `void tile::clear_mem ()`

Hier wird nur der blockierte Speicher und nicht die Bildinformation auf die gezeigt wird gelöscht.

3.29.3.4 `float * tile::copy_data () const`

This methods allocates memory for a 1D Array and copies the data that this tile points to into this array in a row (width) major fashion. The pointer to the copy of the data has to be deleted to avoid memory leaks. This is a nice little helper method.

Returns

3.29.3.5 `void tile::generate_from_image (float_image * img, long tile_max_width, long tile_max_height, long iw, long ih)`

Fill the tile with pointers to the data of the given image.

Parameters

<i>img</i>	Image
<i>tile_max_width</i>	The width of the tile. If the width is larger than what is left of the image, then only the remainder of the pixels of the image are linked to the tile and the width of the tile is set accordingly.
<i>tile_max_height</i>	The number of tiles in height direction
<i>iw</i>	Index of the tile in width direction, beginning with 0
<i>ih</i>	Index of the tile in height direction, beginning with 0

3.29.3.6 `long tile::get_height () const`

Returns

The actual height of the tile.

3.29.3.7 `tilegroup * tile::get_tilegroup ()`

Returns

Pointer to the tilegroup this element belongs to. Zero if it does not belong.

3.29.3.8 `long tile::get_width () const`

Returns

The actual width of the tile.

3.29.3.9 `bool tile::has_group ()`

Returns

True if this element belongs to a tilegroup, false if not

3.29.3.10 `tile & tile::multiply (float value)`

Multiply this value to all pixels in the tile.

Parameters

<i>value</i>	
--------------	--

3.29.3.11 tile & tile::rewrap (float *val*)

For each pixel *T* sets pixel to $T = \text{wrap}(T + \text{val}) - \text{val}$.

Parameters

<i>val</i>	value to add.
------------	---------------

Returns

pointer to this

3.29.3.12 tile & tile::wrap ()

Only works if pixelvalue is in $[-\pi, \pi]$ and *val* is in $[-2\pi, 2\pi]$.

Wraps a block to the interval $[-\pi, \pi]$

Returns

reference to this

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

- include/block_srncp/tile.h
- src/block_srncp/tile.cpp

3.30 tile_image Class Reference

3.30.1 Detailed Description

that provides a framework to divide an image into rectangular tiles. The tile width and height are specified. If the width/height of the whole image is not a multiple of the tile width/height, the tiles in the last column/row get the remainder of the pixels.

Diese Klasse ist nicht von `abstract_image` abgeleitet, da hier nicht auf die einzelnen Pixel Zugriff bestehen soll, sondern nur auf die Tiles.

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

- include/block_srncp/tesselated_image.h

3.31 tile_junction Class Reference

Public Types

- enum `rel_position` { **UP**, **DOWN**, **LEFT**, **RIGHT** }

Public Member Functions

- [tile_junction](#) ()
Pointer to the tiles connected.
- [tile_junction](#) ([tile](#) *first, [tile](#) *second, [rel_position](#) second_to_first)
- [tile_junction](#) (const [tile_junction](#) &t)
- [tile_junction](#) & [operator=](#) (const [tile_junction](#) &)
- [tile](#) * [get_first](#) ()
- [tile](#) * [get_second](#) ()
- [rel_position](#) [get_relative_position](#) ()

3.31.1 Detailed Description

helper that encapsules a connection between two tiles. It stores the two tiles connected as well as the relative position of the second tile to the first. It also provides methods for calculating the variance of a

3.31.2 Member Enumeration Documentation

3.31.2.1 enum [tile_junction::rel_position](#)

Dieser enum gibt an, welche position die tile second relativ zur tile first hat.

3.31.3 Constructor & Destructor Documentation

3.31.3.1 [tile_junction::tile_junction](#) ([tile](#) * first, [tile](#) * second, [rel_position](#) second_to_first)

Konstruktor

Parameters

<i>first</i>	First tile
<i>second</i>	Second tile
<i>second_to_first</i>	Position of the second tile relative to first tile (left/right = w +/- 1 and up/down = h +/- 1)

3.31.4 Member Function Documentation

3.31.4.1 [tile_junction::rel_position](#) [tile_junction::get_relative_position](#) ()

Gives the relative position of second tile to first tile.

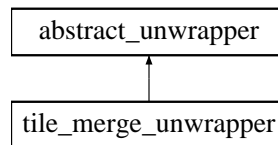
Returns

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

- include/block_srncp/tile_junction.h
- src/block_srncp/tile_junction.cpp

3.32 [tile_merge_unwrapper](#) Class Reference

Inheritance diagram for [tile_merge_unwrapper](#):



Public Member Functions

- [tile_merge_unwrapper](#) (int *N_width_hint*, int *N_height_hint*, [abstract_tile_unwrapper](#) **unwrapper*, [abstract_tile_merger](#) **merger*)
- virtual bool [unwrap](#) ([float_image](#) **wrapped_phase_image*, [float_image](#) **unwrapped_phase_image*)

3.32.1 Detailed Description

provides a comfortable way of handling the tile unwrapping and merging process to unwrap an image. The class provides with an instant of an [abstract_tile_unwrapper](#) and an [abstract_tile_merger](#) and will unwrap a given image using the two elements. The tiles will be first unwrapped individually using the `tile_unwrapper` and then merged using the specified merger.

3.32.2 Constructor & Destructor Documentation

3.32.2.1 `tile_merge_unwrapper::tile_merge_unwrapper (int N_width_hint, int N_height_hint, abstract_tile_unwrapper *unwrapper, abstract_tile_merger *merger)`

Initialize the unwrapper.

Parameters

<i>tile_N_width_hint</i>	Number of tiles in width direction. The actual number used will be close but is probably not identical (see tesselated_image).
<i>tile_N_height_hint</i>	Number of tiles in height direction. The actual number used will be close but is probably not identical (see tesselated_image).
<i>unwrapper</i>	This unwrapper will perform the unwrap process of the individual tiles.
<i>merger</i>	This is the merger that merges the unwrapped tiles.

3.32.3 Member Function Documentation

3.32.3.1 `bool tile_merge_unwrapper::unwrap (float_image * wrapped_phase_image, float_image * unwrapped_phase_image) [virtual]`

This object will copy the input image since the tile-based unwrap procedures will alternate the input image.

Parameters

<i>wrapped_phase_image</i>	
<i>unwrapped_phase_image</i>	

Returns

Implements [abstract_unwrapper](#).

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

- include/block_srncp/tile_merge_unwrapper.h
- src/block_srncp/tile_merge_unwrapper.cpp

3.33 tilegroup Class Reference

Public Member Functions

- [~tilegroup](#) ()
Destruktor.
- void [add_tile](#) ([tile](#) *t)
- int [size](#) ()
- void [add_value](#) (float val)

Static Public Member Functions

- static [tilegroup](#) * [create_new](#) ()

Friends

- void [merge_tilegroups](#) ([tilegroup](#) *g1, [tilegroup](#) *g2)

3.33.1 Detailed Description

a helper class that groups tiles into a single group It allows easy access to all elements of a tile group.

3.33.2 Constructor & Destructor Documentation

3.33.2.1 [tilegroup::~tilegroup](#) ()

Destruktor.

SEGFAULT?

<http://stackoverflow.com/questions/10464992/c-delete-vector-objects-free-memory>

3.33.3 Member Function Documentation

3.33.3.1 [void tilegroup::add_tile](#) ([tile](#) * *t*)

Add an Element to this group. The pointer to the tilegroup is updated in for the given tile t and ONLY FOR THIS TILE and not any other tiles in any other group it might have been in.

Parameters

<i>t</i>	Tile to add to this group.
----------	----------------------------

3.33.3.2 [void tilegroup::add_value](#) ([float](#) *val*)

Add value to all tiles.

Parameters

<i>val</i>	The value to add.
------------	-------------------

3.33.3.3 static tilegroup* tilegroup::create_new () [inline],[static]

Creates a new tilegroup. This is so new groups can only be created on the heap;

Returns

3.33.3.4 int tilegroup::size ()

Returns

The number of elements in the tilegroup.

3.33.4 Friends And Related Function Documentation

3.33.4.1 void merge_tilegroups (tilegroup * *g1*, tilegroup * *g2*) [friend]

Merges the tiles from 2 groups into one group. Group *g2* is merged into *g1*.

Parameters

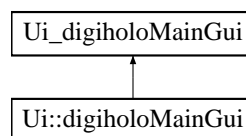
<i>g1</i>	First group
<i>g2</i>	Second group.

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

- include/block_srncp/tile.h
- src/block_srncp/tilegroup.cpp

3.34 Ui_digiholoMainGui Class Reference

Inheritance diagram for Ui_digiholoMainGui:



Public Member Functions

- void **setupUi** (QMainWindow *[digiholoMainGui](#))
- void **retranslateUi** (QMainWindow *[digiholoMainGui](#))

Public Attributes

- QWidget * **centralwidget**
- QVBoxLayout * **verticalLayout_2**

- QGridLayout * **gridLayout_3**
- QLineEdit * **qleFringeFolder**
- QLabel * **label**
- QToolButton * **qtbChooseFolder**
- QGridLayout * **gridLayout_6**
- QGroupBox * **qgbInput**
- QVBoxLayout * **verticalLayout**
- QGridLayout * **gridLayout_5**
- QComboBox * **qcbResolution**
- QLabel * **label_4**
- QGridLayout * **gridLayout_4**
- QLabel * **label_6**
- QLabel * **label_5**
- QLineEdit * **qleWidth**
- QLineEdit * **qleHeight**
- QGridLayout * **gridLayout**
- QComboBox * **qcbInputEncoding**
- QLabel * **label_2**
- QGroupBox * **qgbOutput**
- QHBoxLayout * **horizontalLayout**
- QGridLayout * **gridLayout_2**
- QCheckBox * **qcpGenerateWrapped**
- QCheckBox * **qcpGenerateUnwrapped**
- QCheckBox * **qcb16bitOutput**
- QPushButton * **qpbStart**
- QTextEdit * **qteOutput**
- QProgressBar * **qpbProgress**
- QMenuBar * **menubar**
- QStatusBar * **statusbar**

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

- Qt/digiholo2DGUI/ui_digiholoMainGui.h

Index

- ~minimization_tile_unwrapper
 - minimization_tile_unwrapper, 18
- ~smart_tiled_image
 - smart_tiled_image, 27
- ~tesselated_image
 - tesselated_image, 33
- ~tile
 - tile, 35
- ~tilegroup
 - tilegroup, 40
- abstract_fringe_analyser, 5
 - calc_wrapped_phase_map, 5
- abstract_smart_tile_unwrapper, 6
 - abstract_smart_tile_unwrapper, 6
 - abstract_smart_tile_unwrapper, 6
 - unwrap, 6
- abstract_smart_unwrapper, 6
 - unwrap, 7
- abstract_tile_merger, 8
 - merge_tiles, 8
- abstract_tile_unwrapper, 8
 - abstract_tile_unwrapper, 9
 - abstract_tile_unwrapper, 9
 - unwrap, 9
- abstract_unwrapper, 9
 - unwrap, 10
- add_tile
 - tilegroup, 40
- add_tile_to_group
 - smart_tile, 24
 - smart_tilegroup, 28
- add_value
 - smart_tilegroup, 28
 - tile, 35
 - tilegroup, 40
- calc_junction_reliability
 - srncp_tile_merger, 29
- calc_mean
 - tile, 35
- calc_wrapped_phase_map
 - abstract_fringe_analyser, 5
 - Takeda_FFTW_fringe_analyser, 32
- clear_mem
 - float_image, 14
 - tile, 35
- col_major_float_image, 10
 - col_major_float_image, 10
 - col_major_float_image, 10
- operator(), 11
- convert_to_float_image
 - smart_tiled_image, 27
- copy_data
 - tile, 35
- copy_data_to
 - float_image, 14
- create_new
 - smart_tilegroup, 28
 - tilegroup, 41
- digiholoMainGui, 11
- EDGE, 12
- float_image, 12
 - clear_mem, 14
 - copy_data_to, 14
 - float_image, 13, 14
 - float_image, 13, 14
 - get_data_pointer, 14
 - get_height, 14
 - get_pixel, 14
 - get_width, 15
 - operator(), 15
 - set_pixel, 15
 - zero_fill, 15
- generate_from_image
 - tile, 36
- get_data_pointer
 - float_image, 14
- get_height
 - float_image, 14
 - tile, 36
- get_image_height
 - tesselated_image, 33
- get_image_width
 - tesselated_image, 33
- get_pixel
 - float_image, 14
- get_relative_position
 - smart_tile_junction, 25
 - tile_junction, 38
- get_tile_count_height
 - tesselated_image, 34
- get_tile_count_width
 - tesselated_image, 34
- get_tilegroup
 - smart_tile, 23

- tile, 36
- get_width
 - float_image, 15
 - tile, 36
- grad_fit_tile_unwrapper, 15
 - unwrap, 16
- has_group
 - tile, 36
- has_tilegroup
 - smart_tile, 23
- init
 - ReconstructionThread, 19
- merge_tilegroups
 - smart_tile, 24
 - smart_tilegroup, 29
 - tilegroup, 41
- merge_tiles
 - abstract_tile_merger, 8
 - simple1d_tile_merger, 22
 - srncp_tile_merger, 29
- minimization_tile_unwrapper, 16
 - ~minimization_tile_unwrapper, 18
 - minimization_tile_unwrapper, 17
 - minimization_tile_unwrapper, 17
 - unwrap, 18
- multiply
 - tile, 36
- operator()
 - col_major_float_image, 11
 - float_image, 15
 - row_major_float_image, 21
- operator=
 - smart_tile, 23
 - tesselated_image, 34
- PIXEL, 18
- progressInfo
 - ReconstructionThread, 20
- progressUpdate
 - ReconstructionThread, 20
- qt_meta_stringdata_ReconstructionThread_t, 19
- qt_meta_stringdata_digiholoMainGui_t, 18
- ReconstructionThread, 19
 - init, 19
 - progressInfo, 20
 - progressUpdate, 20
 - request_termination, 20
- rel_position
 - smart_tile_junction, 25
 - tile_junction, 38
- request_termination
 - ReconstructionThread, 20
- rewrap
 - smart_tile, 24
- tile, 37
- row_major_float_image, 20
 - operator(), 21
 - row_major_float_image, 21
 - row_major_float_image, 21
- set_pixel
 - float_image, 15
- simple1d_tile_merger, 21
 - merge_tiles, 22
 - simple1d_tile_merger, 21
 - simple1d_tile_merger, 21
- size
 - smart_tilegroup, 28
 - tilegroup, 41
- smart_tile, 22
 - add_tile_to_group, 24
 - get_tilegroup, 23
 - has_tilegroup, 23
 - merge_tilegroups, 24
 - operator=, 23
 - rewrap, 24
 - smart_tile, 23
 - smart_tile, 23
- smart_tile_junction, 25
 - get_relative_position, 25
 - rel_position, 25
 - smart_tile_junction, 25
 - smart_tile_junction, 25
- smart_tiled_image, 26
 - ~smart_tiled_image, 27
 - convert_to_float_image, 27
 - smart_tiled_image, 26
 - smart_tiled_image, 26
 - unwrap_tiles, 27
- smart_tilegroup, 27
 - add_tile_to_group, 28
 - add_value, 28
 - create_new, 28
 - merge_tilegroups, 29
 - size, 28
- srncp_tile_merger, 29
 - calc_junction_reliability, 29
 - merge_tiles, 29
- srncp_unwrapper, 30
 - unwrap, 30
- Strand_tile_unwrapper, 31
 - unwrap, 31
- Takeda_FFTW_fringe_analyser, 31
 - calc_wrapped_phase_map, 32
 - Takeda_FFTW_fringe_analyser, 32
 - Takeda_FFTW_fringe_analyser, 32
- tesselated_image, 33
 - ~tesselated_image, 33
 - get_image_height, 33
 - get_image_width, 33
 - get_tile_count_height, 34
 - get_tile_count_width, 34

- operator=, [34](#)
- tesselated_image, [33](#)
- tesselated_image, [33](#)
- unwrap_tiles, [34](#)
- tile, [34](#)
 - ~tile, [35](#)
 - add_value, [35](#)
 - calc_mean, [35](#)
 - clear_mem, [35](#)
 - copy_data, [35](#)
 - generate_from_image, [36](#)
 - get_height, [36](#)
 - get_tilegroup, [36](#)
 - get_width, [36](#)
 - has_group, [36](#)
 - multiply, [36](#)
 - rewrap, [37](#)
 - tile, [35](#)
 - wrap, [37](#)
- tile_image, [37](#)
- tile_junction, [37](#)
 - get_relative_position, [38](#)
 - rel_position, [38](#)
 - tile_junction, [38](#)
 - tile_junction, [38](#)
- tile_merge_unwrapper, [38](#)
 - tile_merge_unwrapper, [39](#)
 - tile_merge_unwrapper, [39](#)
 - unwrap, [39](#)
- tilegroup, [40](#)
 - ~tilegroup, [40](#)
 - add_tile, [40](#)
 - add_value, [40](#)
 - create_new, [41](#)
 - merge_tilegroups, [41](#)
 - size, [41](#)
- Ui::digiholoMainGui, [11](#)
- Ui_digiholoMainGui, [41](#)
- unwrap
 - abstract_smart_tile_unwrapper, [6](#)
 - abstract_smart_unwrapper, [7](#)
 - abstract_tile_unwrapper, [9](#)
 - abstract_unwrapper, [10](#)
 - grad_fit_tile_unwrapper, [16](#)
 - minimization_tile_unwrapper, [18](#)
 - srncp_unwrapper, [30](#)
 - Strand_tile_unwrapper, [31](#)
 - tile_merge_unwrapper, [39](#)
- unwrap_tiles
 - smart_tiled_image, [27](#)
 - tesselated_image, [34](#)
- wrap
 - tile, [37](#)
- zero_fill
 - float_image, [15](#)