digiholo2D

Generated by Doxygen 1.8.6

Thu Oct 30 2014 15:04:36

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

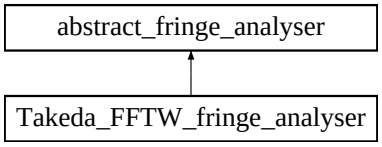
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 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

fringe_pattern	Das Interferenzbild.
wrapped_phase_map	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

t	

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 asbtract method for phase unwrapping.

Parameters

wrapped_phase-_image	This image contains the wrapped phase data.

Returns

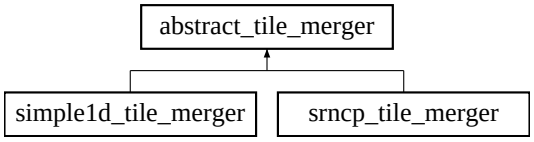
A float_image with the unwrapped phase data. The image will actually be a row_major_float_image.

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 tesselated wrapped phase image to an unwrapped image. These tiles alreasy need to be unwrapped individually and are then unwrapped with respect to each other to form an unwrapped phase map.

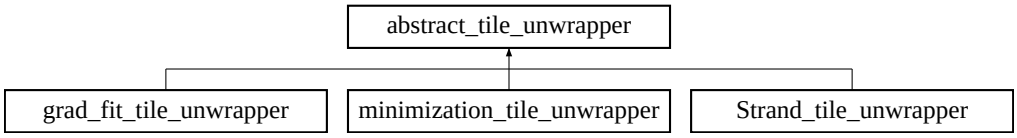
Implemented in srncp_tile_merger, and simple1d_tile_merger.

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 unwrappt eine einzelne tile.

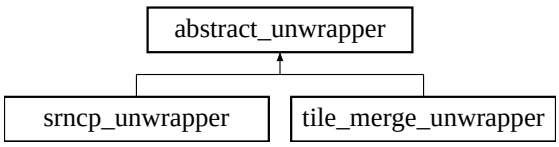
Implemented in minimization_tile_unwrapper, Strand_tile_unwrapper, and grad_fit_tile_unwrapper.

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

wrapped_phase_image	Das Bild mit der Wrapped Phase Verteilung.
unwrapped_phase_image	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.

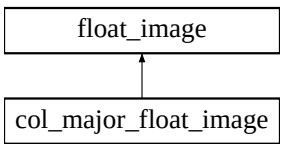
Implemented in tile_merge_unwrapper, and srncp_unwrapper.

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

width	
height	

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.

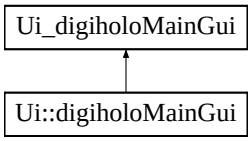
Implements float_image.

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:



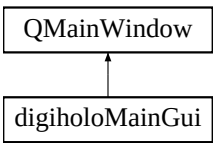
Additional Inherited Members

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

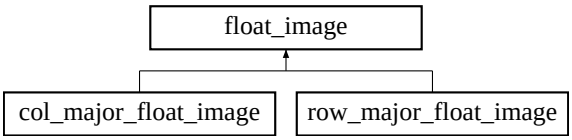
- float **reliab**
- PIXEL * **pointer_1**
- PIXEL * **pointer_2**
- int **increment**

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

width	
height	

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

data	
width	
height	

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

img	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

iw	
ih	

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

iw	
ih	
val	

Returns

3.11.3.9 void float_image::zero_fill ( ) [virtual]

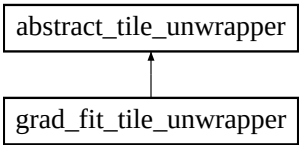
Fill image with zeroes.

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 Phi_fit(x,y) = f(x,y) = a∗x+b∗y+c (dabei x=iw, y=ih)

Parameters

t	tile

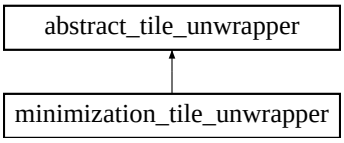
Implements abstract_tile_unwrapper.

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_inter,* float *tol* = 0.0314)

Constructor

Parameters

max_inter	Max number of iterations for minimization
tol	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

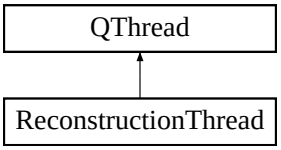
- QByteArrayData **data** [6]
- char **stringdata** [68]

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

fringe_dir_path	This is the base dir that contains the fringe image files.

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

infotext	

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

Signal, für den Fortschritt des Threads.

Parameters

percent	

3.17.1.4 void ReconstructionThread::request_termination ( )

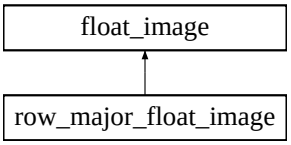
This should be called when Thread execution is to be stopped before it finishes. It basically copies the requestInterruption() functionality from Qt 5.3. which does not exist in QT 5.1.

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

width	
height	

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.

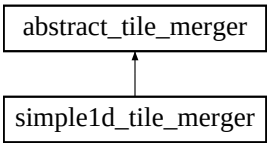
Implements float_image.

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

unwrap_threshold	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 (tesselated_image * t) [virtual]

Merge tiles of tesselated 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

t	

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

other	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

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

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

other	

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

val	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 alreay 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

t	the tile. It must not be in a group.
g	the tilegroup. It must not already contain the tile t.

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

g1	
g2	

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*)

Contructor

Parameters

first	First tile
second	Second tile
second_to_first	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 tesselated 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 tesselated 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

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

uw	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

val	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 tilegoups 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 alreay 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

t	the tile. It must not be in a group.

g	the tilegroup. It must not already contain the tile t.

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

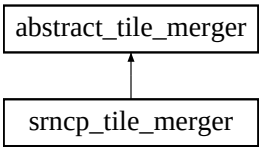
g1	
g2	

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 if 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 (tesselated_image * *t*) [virtual]

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

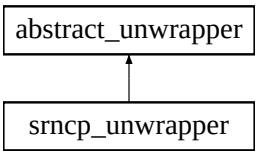
Implements abstract_tile_merger.

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

wrapped_phase_image	Das Bild mit der Wrapped Phase Verteilung.
unwrapped_phase_image	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.

Implements abstract_unwrapper.

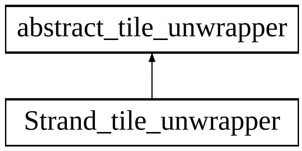
The documentation for this class was generated from the following files:

- 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]`

!!! !!! !!! !!!

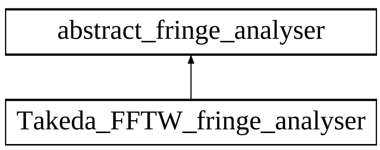
Implements abstract_tile_unwrapper.

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 vonm 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

width	Breite
height	Höhe
flags	

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

fringe_pattern	Das Interferenzbild.
wrapped_phase_map	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 tesselated_image Class Reference

Public Member Functions

- tesselated_image ()
- tesselated_image (float_image *img, long tilecount_width, long tilecount_height)
- tesselated_image & operator= (const tesselated_image &t)
- virtual ~tesselated_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 tesselated_image::tesselated_image ( )

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

3.28.1.2 tesselated_image::tesselated_image (float_image * *img*, long *tilecount_width*, long *tilecount_height*)

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

Parameters

img	The image to be tesselated into tiles
tilecount_width	Number if tiles in width direction. This number is not exactly adhered to, but the real number of tiles will be close.
tilecount_height	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 tesselated_image::~tesselated_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 opccupied by this object but not the underlying image.

3.28.2 Member Function Documentation

3.28.2.1 long tesselated_image::get_image_height ( ) const

Returns

Image height in pixels

3.28.2.2 long tesselated_image::get_image_width ( ) const

Returns

Image width in pixels

3.28.2.3 long tesselated_image::get_tile_count_height ( ) const

Returns

Number of tiles in height direction.

3.28.2.4 long tesselated_image::get_tile_count_width ( ) const

Returns

Number of tiles in width-direction.

3.28.2.5 tesselated_image & tesselated_image::operator= (const tesselated_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

t	

Returns

3.28.2.6 void tesselated_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/tesselated_image.h
- src/block_srncp/tesselated_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

value	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 allcocates 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

img	Image
tile_max_width	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.
tile_max_height	The number of tiles in height direction
iw	Index of the tile in width direction, beginning with 0
ih	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

value	

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

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

Parameters

val	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 [-2PI,2PI].

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

first	First tile
second	Second tile
second_to_first	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.

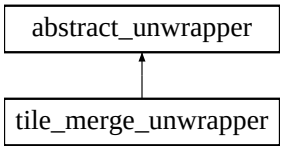
Returns

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 is 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

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

wrapped_phase_image	
unwrapped_phase_image	

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

t	Tile to add to this group.

3.33.3.2 void tilegroup::add_value (float *val*)

Add value to all tiles.

Parameters

val	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

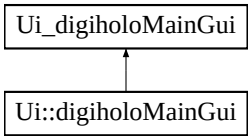
g1	First group
g2	Second group.

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