Code Documentation

Generated by Doxygen 1.8.10

Thu Sep 3 2015 14:18:20
# Contents

1 Code documentation ................................. 99

2 File Index ........................................... 101
   2.1 File List ........................................ 101

3 Class Index .......................................... 103
   3.1 Class List ....................................... 103

4 Class Documentation ................................. 105
   4.1 patch Struct Reference ........................... 105
      4.1.1 Detailed Description ......................... 105
      4.1.2 Member Data Documentation ................. 105
         4.1.2.1 labels .................................... 105
         4.1.2.2 mu ........................................ 105
         4.1.2.3 mu_l ...................................... 105
         4.1.2.4 p ......................................... 106
         4.1.2.5 std ....................................... 106
         4.1.2.6 std_l .................................... 106
         4.1.2.7 var ....................................... 106
         4.1.2.8 var_l .................................... 106

5 File Documentation ................................... 107
   5.1 src/parDistLabelMask.h File Reference .......... 107
      5.1.1 Detailed Description ......................... 107
      5.1.2 Function Documentation ....................... 108
         5.1.2.1 clamp(int x, int a, int b) ................ 108
         5.1.2.2 distLabelMask(unsigned int numPixels, unsigned int *n, std::vector< unsigned int > &vX, std::vector< unsigned int > &vY, std::vector< unsigned int > &vZ, unsigned int numAtlases, patch *patches, patch *patchTarget, std::vector< unsigned int > i, unsigned int long_vector, double thresh, unsigned int numAllPatches, float *dist_labels, unsigned int f, std::vector< float > dist, std::vector< float > dist_lb, std::vector< std::vector< bool > > labels) .................. 108
         5.1.2.3 min(std::vector< float > v) ................ 109
5.1.2.4 parMEX(unsigned int numPixels, unsigned int *X, unsigned int *Y, unsigned int *Z, unsigned int *n, unsigned int numAtlases, patch *patches, patch *patch← Target, unsigned int long_vector, double thresh, unsigned int numAllPatches, int *q, float *dist_labels) ........................................ 109

5.2 parDistLabelMask.h ............................................................... 109
Chapter 1

Code documentation

This appendix contains the documentation of the C++ function developed as an optimization of the patch-labeling method proposed by Platero. The aim of this function is to translate part of the original MATLAB code to C++ in order to optimize and reduce execution time.

See also

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

src/parDistLabelMask.h
MEX function that applies non-local means method for automatic brain MRI segmentation . . . 107
Chapter 3

Class Index

3.1 Class List

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

patch

This structure contains an atlas and its mean and standard deviation  

105
Chapter 4

Class Documentation

4.1 patch Struct Reference

This structure contains an atlas and its mean and standard deviation.

Public Attributes

- float * mu
- float * mu_l
- float * std
- float * std_l
- float * var
- float * var_l
- float * p
- bool * labels

4.1.1 Detailed Description

This structure contains an atlas and its mean and standard deviation.
Definition at line 24 of file parDistLabelMask.h.

4.1.2 Member Data Documentation

4.1.2.1 bool* patch::labels

Label

4.1.2.2 float* patch::mu

Mean of the patch

4.1.2.3 float* patch::mu_l

Mean of the patch label
4.1.2.4 float patch::p
Image

4.1.2.5 float patch::std
Standard deviation of the patch

4.1.2.6 float patch::std_l
Standard deviation of the patch label

4.1.2.7 float patch::var
Variance of the patch

4.1.2.8 float patch::var_l
Variance of the patch label

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

- src/parDistLabelMask.h
Chapter 5

File Documentation

5.1 src/parDistLabelMask.h File Reference

MEX function that applies non-local means method for automatic brain MRI segmentation.

```c
#include "mex.h"
#include <vector>
```

Classes

- struct patch
  
  This structure contains an atlas and its mean and standard deviation.

Functions

- void parMEX (unsigned int numPixels, unsigned int *X, unsigned int *Y, unsigned int *Z, unsigned int *n, unsigned int numAtlases, patch *patches, patch *patchTarget, unsigned int long_vector, double thresh, unsigned int numAllPatches, int *q, float *dist_labels)
  
  This function gathers all the input data and prepares the output data.

- void distLabelMask (unsigned int numPixels, unsigned int *n, std::vector<unsigned int> &vX, std::vector<unsigned int> &vY, std::vector<unsigned int> &vZ, unsigned int numAtlases, patch *patches, patch *patchTarget, std::vector<unsigned int> i, unsigned int long_vector, double thresh, unsigned int numAllPatches, float *dist_labels, unsigned int f, std::vector<float> dist, std::vector<float> dist_lb, std::vector<std::vector<bool>> labels)
  
  This function applies the non-local means method.

- int clamp (int x, int a, int b)
  
  This function clamps a float variable between the specified limits.

- float min (std::vector<float> v)
  
  This function obtains the minimum value of a vector of floats.

5.1.1 Detailed Description

MEX function that applies non-local means method for automatic brain MRI segmentation.

Author

Jose María Sanz Sanz
5.1.2 Function Documentation

5.1.2.1 int clamp ( int x, int a, int b )

This function clamps a float variable between the specified limits.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>Variable to clamp</td>
</tr>
<tr>
<td>a</td>
<td>Lower limit</td>
</tr>
<tr>
<td>b</td>
<td>Upper limit</td>
</tr>
</tbody>
</table>

Returns

Clamped value

5.1.2.2 void distLabelMask ( unsigned int numPixels, unsigned int * n, std::vector< unsigned int > & vX, std::vector< unsigned int > & vY, std::vector< unsigned int > & vZ, unsigned int numAtlases, patch * patches, patch * patchTarget, std::vector< unsigned int > i, unsigned int long_vector, double thresh, unsigned int numAllPatches, float * dist_labels, unsigned int f, std::vector< float > dist, std::vector< float > dist_lb, std::vector< std::vector< bool > > labels )

This function applies the non-local means method.

This function computes the similarity measure in order to choose which patches will be selected.

It also applies the non-local means label fusion.

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numPixels</td>
<td>Number of voxels selected</td>
</tr>
<tr>
<td>*n</td>
<td>Image dimensions</td>
</tr>
<tr>
<td>&amp;vX</td>
<td>X coordinates of the selected voxels</td>
</tr>
<tr>
<td>&amp;vY</td>
<td>Y coordinates of the selected voxels</td>
</tr>
<tr>
<td>&amp;vZ</td>
<td>Z coordinates of the selected voxels</td>
</tr>
<tr>
<td>numAtlases</td>
<td>Number of atlases</td>
</tr>
<tr>
<td>*patchTarget</td>
<td>Array containing the reference atlas</td>
</tr>
<tr>
<td>i</td>
<td>Coordinates of the centre of the current patch</td>
</tr>
<tr>
<td>long_vector</td>
<td>Number of voxels in the patch</td>
</tr>
<tr>
<td>thresh</td>
<td>Threshold to select patch</td>
</tr>
<tr>
<td>numAllPatches</td>
<td>Number of voxels in a patch times the number of reference atlas</td>
</tr>
<tr>
<td>*dist_labels</td>
<td>Result of non-local means</td>
</tr>
<tr>
<td>f</td>
<td>Voxel index</td>
</tr>
<tr>
<td>dist</td>
<td>(empty) Auxiliar variable for non-local means</td>
</tr>
<tr>
<td>dist_lb</td>
<td>(empty) Auxiliar variable for non-local means</td>
</tr>
</tbody>
</table>
5.1.2.3 float min ( std::vector<float> v )

This function obtains the minimum value of a vector of floats.

Parameters

| v | Input vector |

Returns

Minimum value

5.1.2.4 void parMEX ( unsigned int numPixels, unsigned int * X, unsigned int * Y, unsigned int * Z, unsigned int * n, unsigned int numAtlases, patch * patches, patch * patchTarget, unsigned int long_vector, double thresh, unsigned int numAllPatches, int * q, float * dist_labels )

This function gathers all the input data and prepares the output data.

This function gets all the input values using MEX-specific functions and prepares the output variable. The functions used appear in the library mex.h:

mxGetData
mxGetPr
mxGetScalar
mxGetLogicals
mxGetNumberOfElements
mxCreateNumericMatrix

Parameters

<table>
<thead>
<tr>
<th>numPixels</th>
<th>Number of voxels selected</th>
</tr>
</thead>
<tbody>
<tr>
<td>*X</td>
<td>X coordinates of the selected voxels</td>
</tr>
<tr>
<td>*Y</td>
<td>Y coordinates of the selected voxels</td>
</tr>
<tr>
<td>*Z</td>
<td>Z coordinates of the selected voxels</td>
</tr>
<tr>
<td>*n</td>
<td>Image dimensions</td>
</tr>
<tr>
<td>numAtlases</td>
<td>Number of atlases (dimension of *patches)</td>
</tr>
<tr>
<td>*patches</td>
<td>Vector containing the reference atlas</td>
</tr>
<tr>
<td>*patchTarget</td>
<td>Structure containing the target image</td>
</tr>
<tr>
<td>long_vector</td>
<td>Number of voxels in the patch</td>
</tr>
<tr>
<td>thresh</td>
<td>Threshold to select patch</td>
</tr>
<tr>
<td>numAllPatches</td>
<td>Number of voxels in a patch times the number of reference atlas</td>
</tr>
<tr>
<td>*q</td>
<td>Patch radius (in voxels) in the three dimensions</td>
</tr>
</tbody>
</table>

*dist_labels | (MEX output) Result of non-local means |

---

5.2 parDistLabelMask.h

```c
labels | (empty) Auxiliary variable for non-local means

```

```c
// Include header files
#include "mex.h"
#include <vector>

// Structure patches
struct patch{
  float *mu;
  float *mu_l;
  float *std;
  float *std_l;
  float *var;
```

Generated on Thu Sep 3 2015 14:18:20 for Code Documentation by Doxygen
float *var_l;
float *p;
bool *labels;

// Declare function prototypes
void parMEX(unsigned int numPixels, unsigned int *X, unsigned int *Y, unsigned int *Z, unsigned int *n, unsigned int numAtlases, patch *patches, patch *patchTarget, unsigned int long_vector, double thresh, unsigned int numAllPatches, int *q, float *dist_labels);

// Declare function prototypes
void distLabelMask(unsigned int numPixels, unsigned int *n, std::vector<unsigned int> &vX, std::vector<unsigned int> &vY, std::vector<unsigned int> &vZ, unsigned int numAtlases, patch *patches, patch *patchTarget, std::vector<unsigned int> i, unsigned int long_vector, double thresh, unsigned int numAllPatches, float *dist_labels, unsigned int f, std::vector<float> dist, std::vector<float> dist_lb, std::vector<std::vector<bool> > labels);

// Declare function prototypes
int clamp(int x, int a, int b);
float min(std::vector<float> v);