**Summary of the 1st week activities**

**Overview of the co-registration method & lab activities**

* Refer to the 1st week slides in the Dropbox folder
  + “FIRST-WEEK-MATERIAL/PRESENTATIONS”

**Lab Session**

* **From 2D images to 2D triangle meshes**
  + - Load a 2D image & (eventually) make the input image a square image
    - Extract the boundary components (black-white adjacent pixels) or representative level-sets
    - Considering pixels as 2D points (with integer coordinates), generate a Delaunay triangulation of the input point set
    - Visualise the triangulation
  + **From 3D (MRI) images to tetrahedral meshes**
    - Load a 3D MR image, single slices, read metadata, and convert MRI slices to JPG images (for co-registration)
    - For each slice, extract representative level-sets and generate a tetrahedral mesh by applying the Delaunay triangulation
    - Visualise the tetrahedralization
  + **Triangle meshes**
    - Load triangle meshes in PLY format
    - Visualize triangle meshes (e.g., segmented 3D districts from MR images)
    - Mesh data structures for mesh representation and visualisation
      * Discussion on different mesh representations (OFF & related data structure, GMSH tool)
    - Change the sampling density of the Delaunay triangulation through subdivision (simple implementation) or simplification (main ideas). Some hints on tools for mesh processing and the generation of tetrahedral meshes (e.g., Meshlab, TET-GEN)
    - Compute and plot the main properties of the triangle mesh
    - Edge length (*isometry-preserving*)
    - Triangle area (*area-preserving*)
    - Triangle angle (*conformal*)
  + **Transformation**
    - Classes of transformations: affine, “algebraic”, etc
    - Classes of transformation in Matlab (affine transformation) and in “our code” (trasforma.m)
    - Evaluation of the transformation distortion in terms of edge length, angles, and triangle areas
    - Overview on the co-registration code and on the structure of deforma.m
    - Selection of preliminary test cased on the code and preliminary results
    - Identification of
  + Interesting test cases (see FIRST-WEEK-MATERIAL/ CO-REGISTRATION-TESTS)
  + possible improvements (e.g., include the class of affine transformation (?) for the co-registration)
  + **Contact**: for questions, I’ll be available by
  + Email: [patane@ge.imati.cnr.it](mailto:patane@ge.imati.cnr.it)
  + Skype: giuseppe-imati

**Summary of results during the first week**

### Meshing

We implement 2D and 3D visualization of medial images by using OCTAVE via the image processing packages : IMAGE and DICOM. The octave routines IMREAD and IMWRITE are used to load and write images. A grayscale image which is composed by a set of pixels can be considered as a matrix whose components indicate the corresponding gray intensity. We then extract from level sets (using the CONTOURC routine) the gray intensity of the sample points from which we generate triangle (2D) and tetrahedral (3D) meshes representing the approximated geometry of segmented regions of the image (see Fig. 1 in 2D case and Fig. 2 in 3D case). To this end, we use an available useful method called Delaunay algorithm (‘delaunay.m’ as an octave routine). Relying on the criteria of length, area and angle, we then naively refined the meshe to obtain a better, more regular meshe (see Fig. 3) by means of taking midpoints of triangles whose larger edge was above a given threshold.

|  |
| --- |
|  |
| Fig. 1 : 2D MRI image (slice 57) and corresponding 2D triangle mesh |

|  |
| --- |
|  |
| Fig. 2 : 3D tetrahedral meshes in 2D view and 3D view. |

|  |
| --- |
|  |

|  |
| --- |
| Fig. 3 : Naive refinement of 2D mesh from medical1128.jpg with isovalue=200 and suppression of smallest lines |

We also performed mesh visualization of .ply format files.



Fig. 11 : Visualisation of 3D .ply file ; phalanxes and metacarpals.

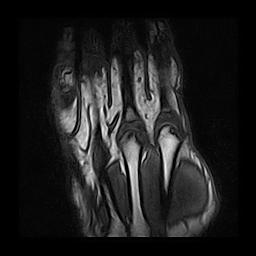
|  |
| --- |
|  |

### Reading MRI

Within the Dicom package, two functions were mainly used. The dicomread function allowed us to read MRI files and store these informations into grayscale matrices. The dicominfo functions yields many metadata about each slice, like these ones of particular interest :

* BodyPartExamined: 'HAND'
* MRAcquisitionType: '3D'
* SpacingBetweenSlices: 0.8000
* InPlanePhaseEncodingDirection: 'ROW'
* FlipAngle: 65
* Rows: 256
* Columns: 256

They were extracted from the file PT\_105/17508601 :



### 

### 

### Towards MRI coregistration

We used the Octave code (SSIM) provided by our instructor to deal with the following problem. Given two images (supposedly of the same kind of object, e.g. reference picture of a district and the corresponding MRI from a patient), find the geometric transformation mapping onto the other achieving the minimum distance in the sense of least squares or WNRMSE (see [2]).

We implemented a routine under MatLab and Octave to load a whole stack of MRI images with the Dicom package. The following pictures are yielded by this process.

Multiple benchmarks were addressed, namely :

|  |
| --- |
|  |
| Fig. 4 : Comparison of two adjacent MRI slices from the same patient at the same time |

|  |
| --- |
|  |
| Fig. 5 : Comparison of two distant (9 skipped frames) MRI slices of the same patient at the same time |

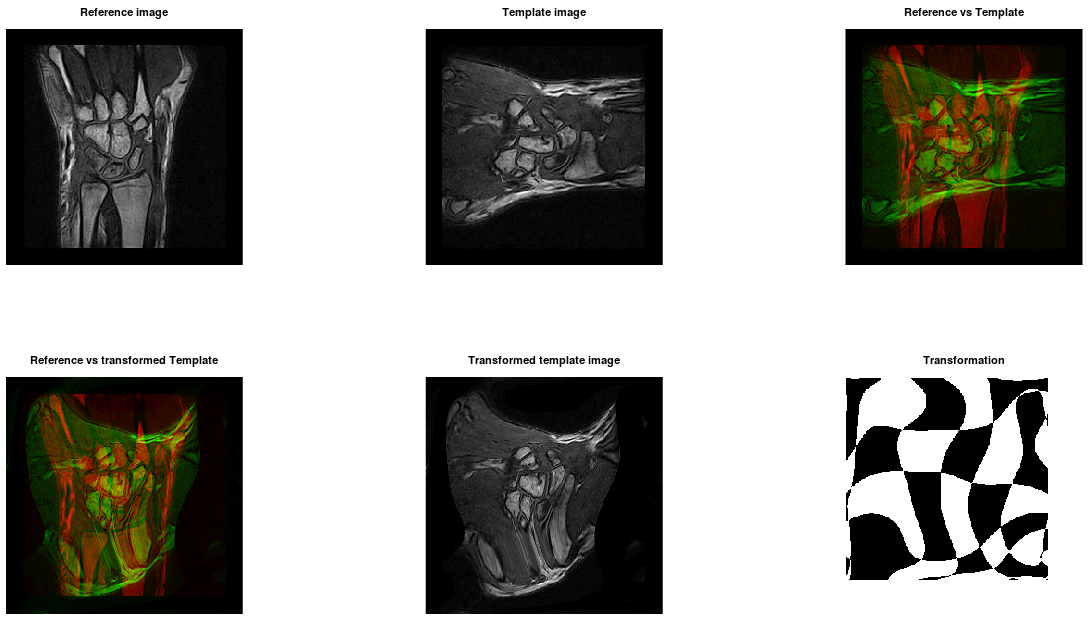


Fig. 6 : Comparison of two distant (9 skipped frames) MRI slices of the same patient at the same time with transposition of one of them

|  |
| --- |
|  |

|  |
| --- |
| Fig. 7 : Comparison of non similar MRI slices from two different patients    Fig. 8 : Comparison of similar MRI slices from two different patients  We found out the script lacks some features. In particular, the third example above (Fig. 6) and the following example (Fig. 9, 10) shows neither rotations nor translations and scalings are available. Moreover, the program crashes whenever the two input picture are not squares, the same size, which is a power of two given in advance ; this is a reliability issue not too complicated to avoid. We are considering implementing these during the following weeks. |
| Fig. 9 : Translating and scaling issues    Fig. 10 : Corresponding convergence plot |